Oracle® Beehive

Administrator's Guide

Release 2 (2.0.1.8)

E16648-07

August 2013

Documentation for administrators whose task is the installation, deployment, configuration, administration, and maintenance of Oracle Beehive.

Oracle Beehive Administrator's Guide Release 2 (2.0.1.8)

E16648-07

Copyright © 2008, 2013, Oracle and/or its affiliates. All rights reserved.

Primary Author: Sonia Nagar

Contributing Authors: Jason Davis, Manon Delisle, Paul Nock, Jamie Rancourt

Contributors: Henrik Blixt, Pradeep Chulliyan, Vikas Dhamija, Ray Dutcher, Richard Hall, Duane Jensen, Ravi Jupudy, Rodrigo Lima, Tait McCarthy, Joe Paradise, Mark Paterson, Rajesh Parakkal, Gregory Pekofsky, François Perrault, Alain Petit, Mark Preston, Jay Rajiva, Sudip Roy, Costa Siourbas, Ridwan Tan, Mike Zhou

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

Contents

5 Managing Oracle Beehive Services

6 Managing Oracle Beehive Workspaces

7 Managing Oracle Beehive Mobility Services

8 Managing Oracle Beehive E-mail

12 Oracle Beehive Client Customization

13 Managing Oracle Beehive Access Control

14 Managing Oracle Beehive Auditing

15 Backing Up and Recovering Oracle Beehive

A Oracle Beehive Ports Reference

Index

List of Figures

List of Tables

List of Examples

Preface

The *Oracle Beehive Administrator's Guide* describes administration tasks associated with Oracle Beehive.

Audience

The *Oracle Beehive Administrator's Guide* is intended for administrators whose task is the installation, deployment, configuration, administration, and maintenance of Oracle Beehive.

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 Oracle Beehive Release 2 (2.0) documentation library:

Administration Guides

- *Oracle® Beehive Administrator's Reference Guide*
- *Oracle® Beehive Beekeeper Online Help (Integrated UA)*
- *Oracle® Beehive Integration Guide*

Application Development

- *Oracle® Beehive Application Developer's Guide*
- *Oracle® Beehive Business Views*
- *Oracle® Beehive Java Content Repository Java API Reference*
- *Oracle® Beehive RESTful Web Services API Reference*
- *Oracle® Beehive SOAP Web Services API Reference*

Installation Guides

- *Oracle® Beehive Installation Guide for Linux*
- *Oracle® Beehive Installation Guide for Microsoft Windows*
- *Oracle® Beehive Installation Guide for Oracle Solaris on SPARC (64-Bit)*

- *Oracle® Beehive Installation Help (Integrated UA)*

Online Helps

- *Oracle® Beehive Central*
- *Oracle® Beehive Webmail*
- *Oracle® Beehive Standards-based Clients*
- *Oracle® Beehive Team Collaboration*
- *Oracle® Beehive Conferencing*
- *Oracle® Beehive Extensions for Explorer Supplemental Help and Release Notes*
- *Oracle® Beehive Extensions for Outlook Supplemental Help and Release Notes*
- *Oracle® Beehive Extensions for Explorer (OBEE) (Integrated UA)*
- *Oracle® Beehive Extensions for Outlook (OBEO) (Integrated UA)*

Mobile Devices

- *Oracle® Beehive Using Windows Mobile Device*
- *Oracle® Beehive Using BlackBerry*
- *Oracle® Beehive Using iPhone or iPad*
- *Oracle® Beehive Registering and Configuring Mobile Devices*

Planning Guides

- *Oracle® Beehive Concepts*
- *Oracle® Beehive Deployment Guide*
- *Oracle® Beehive Licensing Information*

Release Notes

- *Oracle® Beehive 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

Overview of Oracle Beehive Administration

Oracle Beehive is a new, powerful, and unified platform and application for enterprise collaboration. With an architecture that is built on Java 2 Platform Enterprise Edition (J2EE), Oracle Beehive offers a new paradigm for enterprise collaboration: a unified offering for in-context, team-based collaboration.

With Oracle Beehive, users can seamlessly collaborate in teams or individually. Oracle Beehive enables all users to easily save, organize, find, and share the content that they create during the course of their collaborative projects and day-to-day activities. Enterprise colleagues and key partners can leverage the platform to seamlessly interact in a variety of convenient and effective ways.

Oracle Beehive provides familiar collaborative features such as time management, instant messaging, content management, and e-mail, among others, and unifies them in a cohesive platform and application. The Oracle Beehive platform is also built on proven, cost effective, and secure Oracle technologies, such as Oracle Database and Oracle Application Server, for reliability, manageability, and performance.

Getting Started with Oracle Beehive

To get started administering Oracle Beehive, you must first ensure that your installation is complete, and then perform any necessary post-install tasks.

Post-install tasks may include any or all of the following:

- Changing port numbers for various services to new values
- Configuring Oracle Beehive to synchronize with a third-party, LDAP-based user directory
- Installing or configuring a security certificate to enable secure communications protocols such as SSL
- Creating a backup of your deployment in its fresh, successfully-installed state

For a complete list of post-installation tasks, see the "Oracle Beehive Post-Installation Procedures" chapter in the *Oracle Beehive Installation Guide* relevant to your platform.

If you have finished performing all post-install tasks, you can begin working with Oracle Beehive. You should begin by familiarizing yourself with the Oracle Beehive `beectl` command-line utility.

Managing Oracle Beehive Using `beectl`

Access the Oracle Beehive `beectl` command-line utility from the following directory on any Oracle Beehive Application tier:

```
$ORACLE_HOME/beehive/bin
```

The Oracle Beehive `beectl` utility is always used in conjunction with a qualifying command. When running an `beectl` command, use the following syntax:

```
beectl command --option <argument>
```

Where:

- `command` represents the beectl command.
- *--option* represents an available option(s) to use with the command.
- *<argument>* represents a valid argument passed with an option.

For complete documentation of the various `beectl` commands and options, see "Oracle Beehive Command-Line Utility" in Module 2 of the *Oracle Beehive Administrator's Reference Guide*.

Managing Oracle Beehive using Oracle Beekeeper

If you have installed the Oracle Beekeeper administrative user interface, you can perform many common administration tasks using a Web interface, rather than the command-line utility.

In this guide, in most cases administration procedures are described using the command-line interface only. See the online help integrated with Oracle Beekeeper for assistance with performing administration procedures using the Web interface.

Access Oracle Beekeeper by pointing a Web browser to the URL that was displayed at the end of the Oracle Beekeeper installation. By default, the URL is `http://<server_name>:7778/bkpr`. The URL for your installation might be different if you specified a different port number as one of the post-install steps.

See Chapter 27, Oracle Beehive Beekeeper Post-Installation Procedures in the *Oracle Beehive Installation Guide* for your platform for more information.

Managing Oracle Beehive Using Clients

Some administration tasks may be accomplished using Oracle Beehive clients. These tasks are typically oriented towards creating or managing Oracle Beehive entities, such as workspaces, groups, event subscriptions, and resources. The exact functionality exposed varies between the possible clients. For example, using Oracle Beehive Extensions for Outlook exposes workspaces, calendar events, tasks, and resources, while a pure e-mail client such as Mozilla Thunderbird exposes a more limited subset of functionality.

Your First Administration Tasks in Oracle Beehive

The following are some suggestions for first administration tasks using Oracle Beehive. The next sections in this guide describe these tasks in detail.

- Create a structure of organizations to model your user population
- Create and provision some users and groups

  See "Managing and Provisioning Oracle Beehive Users" for details.
- Create some team workspaces

  See "Managing Oracle Beehive Workspaces" for details.
- Create some resources

See "Managing Oracle Beehive Resources" for details.

- Configure Oracle Beehive Mobility Services

  See "Managing Oracle Beehive Mobility Services" for details.

Although the above list is not exhaustive, it is sufficient to expose some of the basic functionality of Oracle Beehive.

2 Starting and Stopping Oracle Beehive

This chapter describes how to start and stop Oracle Beehive, managed components, and processes using the `beectl` command-line utility.

It includes the following topics:

- Starting and Stopping Oracle Beehive
- Starting and Stopping Oracle Beekeeper

Starting and Stopping Oracle Beehive

This section includes the following topics:

- Overview of Starting and Stopping Oracle Beehive
- Getting Started
- Starting and Stopping Oracle Beehive Using the `beectl` Command-Line Utility
- Starting and Stopping Oracle Beehive Managed Components Individually Using the `beectl` Command-Line Utility

Overview of Starting and Stopping Oracle Beehive

Oracle Beehive is a flexible product that you can start and stop in different ways, depending on your requirements. You can start, stop, or restart an Oracle Beehive managed component, or the entire deployment.

The Oracle Database must be running to start Oracle Beehive successfully. If the Oracle Database instance is not running, start it before using the instructions in this section.

The Oracle Install Wizard will attempt to start Oracle Beehive when it completes installation. Occasionally, you will need to stop, start, or restart various managed components of the system or the entire deployment.

Note: The Oracle Install Wizard will attempt to start Oracle Beehive after installation only when the Install and Configure option is selected during installation.

After successfully running, the Configuration Wizard will also attempt to start Oracle Beehive components.

Getting Started

To perform the administration tasks described in this section, you must be logged into the system as the user that installed Oracle Beehive. You can invoke the `beectl` utility every time you run a command, or you can use it in the shell mode. This section explains different methods of invoking the `beectl` shell mode, and includes the following topic:

Using `beectl` Commands in Shell Mode Without Authentication

> **Note:** The `beectl` shell expires if inactive for more than 30 minutes.

Using `beectl` Commands in Shell Mode Without Authentication

To use the `beectl` shell, run the `beectl` command-line utility with no commands or options from the Oracle Beehive `$ORACLE_HOME/beehive/bin` directory. After running the command, the following `beectl` prompt will appear on the command-line:

```
beectl>
```

Once this prompt appears on the command-line, there is no need to specify `beectl` before a command.

Starting and Stopping Oracle Beehive Using the `beectl` Command-Line Utility

This section describes how to stop, start, and restart Oracle Beehive using the `beectl` command-line utility. The instructions in this section assume that the `beectl` shell is being used.

> **See Also:** For more information about the `beectl` commands used in this section, see "Oracle Beehive Command-Line Utility" in Module 2 of the *Oracle Beehive Administrator's Reference Guide*.

Starting Oracle Beehive

To start all Oracle Beehive managed components using the `beectl` command-line utility, use the `start` command with the **--all** option.

The following example illustrates the command with the **--all** option, as well as the resulting output:

```
beectl> start --all
Starting all the beehive components ...
Successfully started all the beehive components.
Operation completed in <time>.
```

Stopping Oracle Beehive

To stop all Oracle Beehive managed components using the `beectl` command-line utility, use the `stop` command with the **--all** option.

The following example illustrates the command with the **--all** option, as well as the resulting output:

```
beectl> stop --all
Stopping all the beehive components ...
Successfully stopped all the beehive components.
Operation completed in <time>.
```

Restarting Oracle Beehive

To restart all Oracle Beehive managed components using the `beectl` command-line utility, use the `restart` command with the **--all** option.

The following example illustrates the command with the **--all** option, as well as the resulting output:

```
beectl> restart --all
Stopping all the beehive components ...
Successfully stopped all the beehive components.
Operation completed in <time>.

Starting all the beehive components ...
Successfully started all the beehive components.
Operation completed in <time>.
```

Starting and Stopping Oracle Beehive Managed Components Individually Using the `beectl` Command-Line Utility

This section describes how to stop, start, and restart Oracle Beehive managed components individually using the `beectl` command-line utility. The instructions in this section assume that the `beectl` shell is being used.

> **See Also:** For more information about the `beectl` commands used in this section, see "Oracle Beehive Command-Line Utility" in Module 2 of the *Oracle Beehive Administrator's Reference Guide*.

Determining the Managed Component Identifier

Use the following instructions to determine the component identifier of a specific Oracle Beehive managed component using the `beectl` command-line utility:

1. Determine the component identifiers of all managed components by running the `status` command:

   ```
    beectl> status
   --------------------------------------------+----------------+---------------
   Component identifier                        | Component type | Status
   --------------------------------------------+----------------+---------------
   BTI_redirector_instance1.host.domain.com    | Bti            | RUNNING
   --------------------------------------------+----------------+---------------
   BEECLIENT_instance1.host.domain.com         | ManagedOc4j    | RUNNING
   --------------------------------------------+----------------+---------------
   BEEAPP_instance1.host.domain.com            | ManagedOc4j    | RUNNING
   --------------------------------------------+----------------+---------------
   BEEMGMT_instance1.host.domain.com           | ManagedOc4j    | RUNNING
   --------------------------------------------+----------------+---------------
   BEECORE_instance1.host.domain.com           | ManagedOc4j    | RUNNING
   --------------------------------------------+----------------+---------------
   oc4j_soa_instance1.host.domain.com          | ManagedOc4j    | RUNNING
   --------------------------------------------+----------------+---------------
   ohs_instance1.host.domain.com               | HttpServer     | RUNNING
   --------------------------------------------+----------------+---------------
   ```

   The component identifier is a string of characters, including the fully qualified host name preceded by a descriptive prefix. For a list of services included in each managed component, see "Managed Component Services".

2. Take note of the managed component identifier.

For example, from the output returned by the `status` command in step 1, we can determine that the component identifier of the Oracle Beehive Applications is: `BEEAPP_instance1.host.domain.com`.

Note: Component identifiers differ from one deployment to another.

Managed Component Services

Table 2–1 describes the services that are associated with different managed components. Stopping, starting, or restarting a managed component will affect all of the associated services listed in the Oracle Beehive Services column of Table 2–1.

Table 2–1 Managed Component Services

Managed Component Prefix	Oracle Beehive Services
`BTI`	Oracle Beehive Transport Infrastructure
`oc4j_soa`	Oracle Container for Java Service Oriented Architecture
`BEEAPP`	CalDAV Service Coexistence Service Conference Service Discussions Service Device Management Service E-mail Service Event Services Fax Service FTP Service Instant Message Services Message Delivery Service Mobile Data Sync Service Mobile Device Management Service Mobile Mail Service Mobile Push Service Notification Service Platform Service Platform Web Service Presence Service Records Management Service Search Service Time Management Service Voice Message Service WebDAV Service XMPP Service
`BEEMGMT`	Management Service

Table 2–1 (Cont.) Managed Component Services

Managed Component Prefix	Oracle Beehive Services
BEECORE	Access Control Service
	Alarm Service
	Audit Service
	Authentication Services
	Policy Service
	Resource Directory Service
	Time Zone Service
	User Directory Service
	Workspace Service
ohs	Oracle HTTP Server

Starting a Specific Managed Component

Use the following instructions to start a specific Oracle Beehive managed component using the `beectl` command-line utility:

1. Determine the component identifier of the managed component to start. For more information about obtaining the component identifier, see "Determining the Managed Component Identifier".

2. Start a specific Oracle Beehive managed component using the `beectl` command-line utility, use the `start` command with the **--component** option and argument.

   The following example illustrates the command and option, including the component identifier noted from the output in Step 2 of "Determining the Managed Component Identifier":

```
 beectl> start --component  BEEAPP_instance1.host.domain.com
Starting beehive component "BEEAPP_instance1.host.domain.com" ...
Successfully started beehive component "BEEAPP_instance1.host.domain.com".
Operation completed in <time>.
```

> **Note:** To start more than one managed component, specify the **--component** option multiple times: once before each argument. For example: `start --component` <*componentID1*> `--component` <*componentID2*>... `--component` <*componentIDn*>

Stopping a Specific Managed Component

Use the following instructions to stop a specific Oracle Beehive managed component using the `beectl` command-line utility:

1. Determine the ID of the managed component to stop. For more information about obtaining the component identifier, see "Determining the Managed Component Identifier".

2. Stop a specific Oracle Beehive managed component using the `beectl` command-line utility, use the `stop` command with the **--component** option and argument.

   The following example illustrates the command and option, including the component identifier noted from the output in Step 2 of "Determining the Managed Component Identifier":

```
beectl> stop --component  BEEAPP_instance1.host.domain.com
```

```
Stopping beehive component "BEEAPP_instance1.host.domain.com" ...
Successfully stopped beehive component "BEEAPP_instance1.host.domain.com".
Operation completed in <time>.

1 of 1 component(s) stopped successfully.
```

Note: To stop more than one managed component, specify the **--component** option multiple times: once before each argument. For example: `stop --component` *<componentID1>* `--component` *<componentID2>*... `--component` *<componenIDn>*

Restarting a Specific Managed Component

Use the following instructions to restart a specific Oracle Beehive managed component using the `beectl` command-line utility:

1. Determine the ID of the managed component to restart. For more information about obtaining the component identifier, see "Determining the Managed Component Identifier".
2. Restart a specific Oracle Beehive managed component using the `beectl` command-line utility, use the `restart` command with the **--component** option and argument.

   The following example illustrates the command and option, including the component identifier, as it appears in the output within Step 1 of "Determining the Managed Component Identifier":

   ```
   beectl> restart --component  BEEAPP_instance1.host.domain.com
   Stopping beehive component "BEEAPP_instance1.host.domain.com" ...
   Successfully stopped beehive component "BEEAPP_instance1.host.domain.com".
   Operation completed in <time>.

   Starting beehive component "BEEAPP_instance1.host.domain.com" ...
   Successfully started beehive component "BEEAPP_instance1.host.domain.com".
   Operation completed in <time>.

   1 of 1 component(s) restarted successfully.
   ```

Note: To restart more than one managed component, specify the **--component** option multiple times: once before each argument. For example: `restart --component` *<componentID1>* `--component` *<componentID2>*... `--component` *<componentIDn>*

Starting and Stopping Oracle Beekeeper

Oracle Beekeeper is installed as a separate, stand-alone OC4J process. To start or stop Oracle Beekeeper, you must start or stop the OC4J component using the `opmnctl` utility.

Ensure your environment is set with the following variables. Adjust the variables to match the install path and version numbers appropriate to your particular installation:

```
export ORACLE_BASE=/home/oracle/oracle/product
export ORACLE_HOME=$ORACLE_BASE/1.5.1.0.0/beekeeper_1
export PATH=$PATH:$ORACLE_HOME/bin:$ORACLE_HOME/opmn/bin
```

To start Oracle Beekeeper, perform the following steps:

1. From the command line on the machine where Oracle Beekeeper is installed, start OPMN using the `opmnctl` utility:

   ```
   > opmnctl start
   opmnctl: opmn started.
   ```

2. Check the status of the OC4J process:

   ```
   > opmnctl status
   ```

   You should see output similar to the following:

   ```
   Processes in Instance: BeehiveControl
   ---------------+-------------+-----+---------
   ias-component | process-type  | pid   | status
   ---------------+-------------+-----+---------
   bkpr                | bkpr              | N/A | Down
   ```

3. Start Oracle Beekeeper.

   - Using Oracle Beekeeper 1.5.1.0 or later:

     ```
     > opmnctl startproc process-type=BEEKEEPER
     opmnctl: starting opmn managed processes...
     ```

   - Using Oracle Beekeeper 1.4.3 or earlier:

     ```
     > opmnctl startproc process-type=bkpr
     opmnctl: starting opmn managed processes...
     ```

   Note: You can also start both OPMN, and Oracle Beekeeper, using the `opmnctl startall` command:

   ```
   > opmnctl startall
   ```

To stop Oracle Beekeeper, perform the following steps:

1. From the command line on the machine where Oracle Beekeeper is installed, run the following command:

   - In Oracle Beekeeper 1.5.1.0 or later:

     ```
     > opmnctl stopproc process-type=BEEKEEPER
     opmnctl: stopping opmn managed processes...
     ```

   - In Oracle Beekeeper 1.4.3 or earlier:

     ```
     > opmnctl stopproc process-type=bkpr
     opmnctl: stopping opmn managed processes...
     ```

2. If you want to, you can also stop OPMN:

   ```
   > opmnctl stopall
   opmnctl: stopping opmn and all managed processes...
   ```

   Note: You can also stop both OPMN, and Oracle Beekeeper, using the `opmnctl stopall` command:

   ```
   > opmnctl stopall
   ```

3

Managing and Provisioning Oracle Beehive Users

This chapter describes how to provision and manage user accounts and user groups in Oracle Beehive.

This chapter contains the following topics:

Introduction to Managing and Provisioning Users With Oracle Beehive

Oracle Beehive provides a flexible user account management and provisioning structure. You can manage user accounts from the command line, or by using a third-party user management product such as an external LDAP directory.

Every unique user account in Oracle Beehive has a corresponding record in the Oracle Beehive database, even when users are managed (mastered) by an external user management product. The Oracle Beehive user account stores settings and parameters necessary to establish permissions and access to the various Oracle Beehive services and user functions. Oracle Beehive automatically synchronizes information when it is duplicated between an external user management product and the Oracle Beehive account profile.

Oracle Beehive user accounts can be mastered in either the Oracle Beehive database, or an external LDAP-based directory. **Mastered** means that source is used as the point of reference to determine the correct value for some user account attributes, and the master source is used for making changes to those account details. When user accounts are mastered in an external directory, Oracle Beehive automatically updates the Oracle Beehive database entry whenever account information in the external directory changes.

When a user account is mastered in an external LDAP-based directory:

- Some attributes which are not in the LDAP-based directory may still be mastered in Oracle Beehive.

- LDAP-mastered attributes cannot be managed in Oracle Beehive. Changes to those attributes must be made in the LDAP-based directory.
- Once LDAP-based directory synchronization is enabled, you cannot add or remove users from Oracle Beehive. You may only add or remove users via the external LDAP-based directory. These user accounts will then be created or removed in Oracle Beehive during the automated synchronization process.

For more information about setting up and managing Oracle Beehive with an external user management product, see Chapter 4, "Integrating an External User Directory with Oracle Beehive" in the *Oracle Beehive Integration Guide*.

This section includes the following topics:

- About the User Directory Service (UDS)
- About Organizations
- About Personal Workspaces
- About Provisioning and Deprovisioning Policies
- About Using `beectl` to Manage User Accounts

About the User Directory Service (UDS)

The User Directory Service (UDS) is the Oracle Beehive service responsible for storing and retrieving user and group information, as well as contact lists (address books). All other Oracle Beehive services query UDS whenever they need to look up user or group data.

The Authentication Service interfaces with UDS during any authentication event to query the user login ID. Passwords are encrypted and stored by the Authentication Service. When users are mastered in an external, LDAP-based directory, the Authentication Service queries both UDS and the external directory for login information, but only the external directory stores the user password information.

About Organizations

All Oracle Beehive installations have an organizational structure rooted at the enterprise level. Each installation may only have one enterprise. However you can further structure your users into organizations. An enterprise may have any number of organizations, and each organization may have sub-organizations. You may create users within the root enterprise, or within an organization.

Note: Adding a user to one organization does not prevent that user from interacting with members of a different organization.

About Personal Workspaces

Every Oracle Beehive user automatically has one personal workspace, created at account creation.

When an account is created, a personal workspace template is used during personal workspace creation. The personal workspace template contains details of the users' default folders, default seeded content, and seeded sensitivities. Service-specific folders are always created in the personal workspace. You can modify the default personal workspace template, and create custom personal workspace templates, to suit your requirements.

For more information about creating and modifying workspace templates, see "Managing Oracle Beehive Workspaces".

About Provisioning and Deprovisioning Policies

Policies are collections of *if ... then* logical statements which define how the Oracle Beehive system should behave, given certain inputs or circumstances. When you create or delete user accounts, the user provisioning or deprovisioning policies are triggered.

Oracle Beehive is shipped with a default provisioning policy. The default provisioning policy determines which personal workspace template should be used. By default there is only one personal workspace template. The default personal workspace template enables all Oracle Beehive services for all users.

You can modify the default provisioning policy, to customize and extend Beehive's automation with regards to user account management.

For example, if you create additional personal workspace templates, the policy can select which template to use, based on the value of a user attribute such as job title. If the new user is a vice president, one personal workspace template is used. If the new user is a software engineer, a different custom personal workspace template is used, otherwise, if the user has neither of these job titles, the default personal workspace template is used.

For more information about creating and managing policies, see "Managing Oracle Beehive Events and Policies".

About Using `beectl` to Manage User Accounts

User account management functionality is exposed in the `beectl` command line tool. You can create user accounts, modify them, assign and change privileges, and delete accounts from the command line. The `beectl` commands for managing user accounts are provided, with full syntax reference in "Oracle Beehive Command-Line Utility", in Module 2 of the *Oracle Beehive Administrator's Reference Guide*.

There is a category of `beectl` commands, called *users*. All of the `beectl` commands related to user and group management are in the users category. You can list these commands using the `beectl list_commands` command:

```
beectl> list_commands --category users
```

Note: Beginning in Oracle Beehive 1.5, the user identifier has changed to the form `user=<login_id>`. The previous form, `loginid=<login_id>` will still work, but is deprecated. In versions of Oracle Beehive previous to version 1.5, you must use the form `loginid=<login_id>`.

About User Accounts

Every user of Oracle Beehive must have a user account. Accounts have varying, highly-granular levels of access to Oracle Beehive services, clients, and stored artifacts. These features allow you to use Oracle Beehive user accounts to accommodate a wide variety of casual, limited, or regular users that will interact with Oracle Beehive in some way.

It is not necessary to create or use shared administration accounts in Oracle Beehive. Instead, each person may be assigned highly granular sets of permissions for performing administrative actions, as appropriate, up to and including total system access. Likewise, you can create user accounts with very limited access, such as for customers, contractors, or partners.

There are three types of users defined in Oracle Beehive:

- Enterprise users

  These users by default have access to all services and objects of the Oracle Beehive system, based on the user's access privileges. Most Oracle Beehive users are Enterprise users.

- Extended-enterprise users

  These users are typically business partners or contractors, with limited access to the Oracle Beehive services based on access privileges.

- External contacts

  External contacts cannot access any part of the Oracle Beehive system. The only purpose of this user type is to have their contact information to be accessible to users of the system in the enterprise global contacts list. External contacts do not have Oracle Beehive user accounts. You can add, modify, and delete external contacts in a manner similar to user accounts.

This section includes the following topics:

- About User Account Fields
- About Unique User Identifiers
- About User `display_name` Values
- About User Memberships
- About Roles
- About User Account Status
- About Special and System-Reserved Accounts

About User Account Fields

Oracle Beehive stores information about user accounts using five different types of account fields, referred to on the command-line as `attribute_type`:

- **Attribute**: A general account field type. Attributes are usually text strings.
- **Principal**: An attribute used for authentication of the user. There must be one primary principal field for every Oracle Beehive user account, but there may be additional non-primary principal fields, including a voice principal for use with voice mail systems and a protocol principal for use with protocols that only support 7-bit character sets.
- **Credential**: A value that must be provided by a user in order to log in to Oracle Beehive, such as the password field. Each principal has a corresponding credential field.
- **Address**: A special attribute which combines a type (business, home, or other), a URI scheme, and a value. Address fields are special because Oracle Beehive can use them as destinations for messages, and can interpret the scheme to determine the correct transportation method for such a message.

- **Property**: This attribute is for custom properties that you create to suit your business needs.

Each of these account fields is described in greater detail later in this section.

Most user account fields are optional, but a few are required, meaning that every user account in Oracle Beehive must have a non-null value for the required account fields. Table 3–1, " Default User Account Fields" shows the default user account fields.

Note:

- If the user will be using any Oracle Beehive e-mail functionality, you must enter an e-mail address (using the `--address` option). It must be unique within the enterprise, and it must conform to the basic e-mail address format (username@domain.xyz).
- If the user will be using any Oracle Beehive instant messaging functionality, you must enter an IM address (using the `--address` option). The IM address serves as a login credential. It must be unique within the enterprise, and it must conform to the XMPP standard. (The XMPP standard is similar to a basic e-mail address format (username@example.com), but is less restrictive of special characters. See: RFC 3920, section 3, available at `http://xmpp.org/rfcs/rfc3920.html`.)
- When creating user accounts from the command line, in addition to providing user account fields, the `beectl add_user` command requires you to specify a container for the user using the `--scope` option: the Enterprise, or optionally, an Organization within the enterprise. These are not user account fields; rather, they specify a **Parent Identifier** to which the user account belongs. The user will automatically have a membership to this parent identifier, but you can add additional memberships using the `--organization` option.
- For many account attributes, you can couple a value with a locale; this allows you to create different values for different locales. For example, you could create a family name for the `en_US` locale using common English spelling, and a different family name for the `fr_FR` locale using common French spelling. The following attributes support multiple locale values: `family_name`, `given_name`, `display_name`, `middle_name`, `job_title`, `prefix`, `suffix`, and `nick_name`.
- Beginning in Oracle Beehive 1.5, the `display_name` attribute must always have a value. If you do not enter a value for `display_name`, one will be automatically generated according to the current rule. See the section "About User `display_name` Values" on page 3-13.

Table 3–1 Default User Account Fields

Attribute	Required or Optional	Field Type	Details
`--family_name`	Required	Attribute	All users must have a family name (surname, or last name). This value does not need to be unique. You can use any combination of alphabetic (including multi-byte) characters. If the familyname contains a space or any special characters, you must surround it with double quotes.
`--given_name`	Optional	Attribute	User accounts may optionally have a given (first) name. This value does not need to be unique. You can use any combination of alphanumeric characters. If the given name contains a space or any special characters, you must surround it with double quotes.
`--display_name`	Optional	Attribute	An alternative name to be used for display in various clients. This value does not need to be unique. You can use any combination of alphanumeric characters. If the display name contains a space or any special characters, you must surround it with double quotes.
`--middle_name`	Optional	Attribute	An optional middle name. This value does not need to be unique. You can use any combination of alphanumeric characters. If the middle name contains a space or any special characters, you must surround it with double quotes.
`--prefix`	Optional	Attribute	A name prefix such as Mr., Mrs., or Dr.
`--suffix`	Optional	Attribute	A name suffix such as Jr. or Sr.
`--nick_name`	Optional	Attribute	A nickname, such as Bob (for Robert)
`--login_id`	Required	Principal (Primary)	The value of this attribute is the exact string the user will normally enter when logging in to the system. It must be unique among all users in the enterprise. Although inclusion of this attribute at account creation is not enforced, if it is null (not created), it is not possible to log in to Oracle Beehive with this account. Consequently, for practical purposes the `login_id` should be considered as mandatory for most accounts.

Table 3–1 (Cont.) Default User Account Fields

Attribute	Required or Optional	Field Type	Details
`--login_password`	Required	Credential	The password to be used for the first login to Oracle Beehive. After the first login the user can create a new password. User account creation will fail if the password value violates the password policy. By default, the Oracle Beehive password policy dictates that passwords must be at least 6 characters long, contain at least one capital letter, and contain at least one non-alphabetic character. If you are using `beectl` shell mode, you may enter the password in plaintext. If you are using single-command mode, you must obfuscate the password using the `beectl obfuscate` feature. Although this attribute is required, it can be null (not created) if the Oracle Beehive password policy is modified to allow a zero-length password. By default, a password must be created for an account in accordance with the password policy.
`--primary_address`	Optional	Address	A primary address designates an address that should be the default address for the user account. See "About User Account Addresses" on page 3-10 for details on how to format an address.
`--address`	Optional	Address	An address is a composite attribute consisting of a type, a scheme, and a value. Addresses include phone numbers, e-mail addresses, street addresses, fax numbers, instant message IDs, and so forth. You can specify many addresses. See "About User Account Addresses" on page 3-10 for details on how to format an address.
`--default_address_for_type`	Optional	Address	Addresses consist of a type, a scheme, and a value. This option lets you set an address that will be the default address for that type, for this user. See "About User Account Addresses" on page 3-10 for details on how to format an address.
`--default_address_for_scheme`	Optional	Address	Addresses consist of a type, a scheme, and a value. This option lets you set an address that will be the default address for that scheme, for this user. See "About User Account Addresses" on page 3-10 for details on how to format an address.
`--voice_principal`	Optional	Address (Principal)	If the user will be using any Oracle Beehive voicemail functionality, you must enter a phone number. The phone number serves as a login to the voice mail feature under certain conditions, in combination with the PIN. It must be unique within this enterprise.

Table 3–1 (Cont.) Default User Account Fields

Attribute	Required or Optional	Field Type	Details
`--voice_pin`	Optional	Credential	If you enter a phone number, and you will be using any Oracle Beehive voicemail functionality, you must enter an initial PIN. The user must enter the PIN when logging in to the voicemail feature. The user can change the PIN after the initial login. The PIN must conform to any PIN "password" policy currently active.
`--protocol_principal`	Optional	Principal	Because the primary principal may contain characters that are illegal with certain protocols (such as 7-bit e-mail protocols), you can use this field to enter a protocol-friendly alternate principal.
`--protocol_password`	Optional	Credential	A password for the protocol principal. It must be compliant with the default password policy, and additionally should only contain 7-bit characters, to maintain compatibility with various protocols.
`--timezone`	Optional	Attribute	You may enter a time zone identifier corresponding to the user's home time zone. If you do not enter a value for --timezone, under most conditions Oracle Beehive will assume the user is in the time zone Etc/GMT. To get a list of valid time zone identifiers, use the `beectl list_timezones` command. **Note**: Users can update their own timezone attribute from various clients and preference pages.
`--locale`	Optional	Attribute	You may enter a locale identifier corresponding to the user's home locale. The locales should be in the format `x[_y]` where `x` is per per ISO639 and `y` is per ISO3166. For example, the United States English locale identifier is `en_US`.
`--job_title`	Optional	Attribute	A text field for specifying a job title.
`--office_location`	Optional	Attribute	A text field for specifying an office location.
`--company`	Optional	Attribute	A text field for specifying a company.
`--profession`	Optional	Attribute	A text field for specifying a profession.
`--department`	Optional	Attribute	A text field for specifying a department.
`--assistant`	Optional	Attribute	You can enter the identifier of another Oracle Beehive user account, as this user's assistant.
`--manager`	Optional	Attribute	You can enter the identifier of another Oracle Beehive user account, as this user's manager.
`--external_inbox`	Optional	Attribute	You can indicate that this user should have all e-mail messages routed to an external inbox, outside of Oracle Beehive.If you set this attribute to true, all e-mail messages for this user will be sent to the outbound VMS for delivery. This attribute is not in versions of Oracle Beehive older than 1.3.

Table 3–1 (Cont.) Default User Account Fields

Attribute	Required or Optional	Field Type	Details
`--organization`	Optional	Attribute	You can specify membership in additional organizations by entering the identifier of an Organization. You can specify multiple such memberships. **Note**: to specify that the user should be created at a given organization's scope, use the `--scope` option with the organization's identifier instead.
`--extended_enterprise_user`	Optional	Attribute	You can specify that this user is an extended enterprise user using this attribute. This attribute is not available in versions of Oracle Beehive older than 1.3.
`--property`	Optional	Property	A special field for entering custom user account attributes, notes, and a user's encryption certificate. See "About User Account Properties" on page 3-12 for details.

This section contains the following topics:

- About User Account Principals and Credentials
- About User Account Addresses
- About User Account Attributes
- About User Account Properties
- About the Timezone Attribute with LDAP Synchronization

About User Account Principals and Credentials

In Oracle Beehive, *principals* are globally-unique user names or identifiers, and *credentials* are passwords. You specify principals and credentials when creating or modifying user accounts, but while credentials are exposed for adding or modifying through the User Directory Service, they are handled and stored in a secure manner by the Authentication Service.

Principal is a field type, which is assigned to a user attribute. For example, by default (primary) principal is assigned to the login ID field.

Oracle Beehive user accounts must have at least one principal field (the primary principal), but it is possible to include additional principals in a user account.

The reason for supporting multiple principals is to allow a full range of acceptable characters for normal use in the primary principal, but include secondary (or protocol) principals for protocols that only support a limited character set.

For example, if a user's name contains multi-byte characters, a secondary principal composed only of 7-bit characters allows the user to authenticate over the IMAP protocol. Another secondary principal, called the voice principal, is composed only of numbers allows the user to authenticate with a voice mail system using a touch tone telephone.

When you create a user using `beectl`, you use the `--login_id` option to set the value for the primary principal.

A credential is a password for a given principal. The value of a credential field must conform to the password policy.

About User Account Addresses

In Oracle Beehive, *addresses* are special user account fields designed to contain information that designates a destination for a message — an e-mail, a written letter, a telephone call, or a transfer of data via some other protocol.

Each address field is labeled according to a scheme that contains two parts:

- **Type** – A label indicating the type of address (business, personal, other, or proxy), and a numeric value (1 through 5 for business, personal, and other and 1 through 25 for proxy).
- **Scheme** – A value indicating a specific address (URI) scheme.

An Oracle Beehive account may contain up to forty addresses of each scheme: five each of `business`, `personal` , and `other` addresses, and twenty five `proxy` addresses.

Table 3–2, " Oracle Beehive User Account Address Field Schemes" lists all of the valid schemes for an Oracle Beehive user account address field.

Table 3–2 Oracle Beehive User Account Address Field Schemes

Scheme	Description
FAX	Fax
FTP	File Transfer Protocol
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IM	Instant Messaging
ORAISDN	reserved
IMAP	Instant Message Access Protocol
LDAP	Lightweight Directory Access Protocol
MAILTO	Electronic mail address (e-mail)
NEWS	USENET news
NNTP	USENET news using NNTP access
ORAALERT	reserved
ORAASSISTANTPHONE	reserved
ORACALLBACK	reserved
ORACARPHONE	reserved
ORAMOBILE	reserved
ORAPUSH	reserved
ORAPAGER	reserved
ORAPOSTAL	Postal addresses, containing fields in the following format: `l1=<address-line-1>?l2=<address-line-2>?box=<post-box-number>?cy=<city>?st=<state>?code=<postal-code>?c=<country>` The fields `l1`, `l2`, `box`, and `cy` are optional, but at least one of them should be present.
ORARADIO	reserved

Table 3–2 (Cont.) Oracle Beehive User Account Address Field Schemes

Scheme	Description
ORASMS	reserved
ORATELEX	reserved
ORATTYTTD	reserved
ORAVMAIL	reserved
PRES	Presence
SIP	Session Initiation Protocol
TEL	Telephone number
URN	Uniform Resource Names
XMPP	Extensible Messaging and Presence Protocol

> **Note:** Schemes beginning with *ora* are reserved for Oracle Beehive internal messaging functionality. For example, the `ORAALERT` scheme is used by Oracle Beehive for addressing alerts.

The following example demonstrates how to specify an address and its value using the `beectl modify_user` command. In this example, a fax number is added to a preexisting user account:

```
beectl> modify_user --user user=user1 --address +business_1:FAX:6505551212
```

> **Note:** You can use the `--address` operator, the `--primary_address` operator, the `--default_address_for_type` operator, or the `--default_address_for_scheme` operator.

In this case, a new business fax field is created, but is not designated as a primary address or default address for type or scheme. The type section specifies that it is `business_1`; there could be up to five business fax fields, using types `business_1` through `business_5`.

The following example specifies the new attribute as the default address for type:

```
beectl> modify_user --user user=user1 --default_address_for_type +business_
1:FAX:6505551212
```

By using the `--default_address_for_type operator`, this value becomes the default `business_1` attribute, among all attribute types (phone numbers, e-mail addresses, and so forth).

About User Account Attributes

The user account field type, *attribute*, is the general type. While all user account fields may be commonly referred to as attributes, those which are not properties, addresses, principals, or credentials, are *attributes*.

Table 3–1, " Default User Account Fields" shows which user account fields have the *attribute* type, in the **Type** column.

About User Account Properties

Properties refers to custom user properties which you have created.

A property field has a name, value, and description. To add a property to a user account, use the `beectl modify_user` command with the `--property` option:

```
beectl> modify_user --user user=user1 --property +name=value=description
```

Use the = (equal) character to separate the property name, value, and description.

You can create new, custom user properties. This allows you to extend user accounts with your own attributes.

Use the `beectl add_custom_user_property` command to create a new custom user property:

```
beectl> add_custom_user_property --name <attribute_name> --type <property_type> [
--description <description> ]
```

The `--type` is the field type that will be used for this custom property, such as integer, string, boolean, and so forth.

You can list all custom user properties using the `beectl list_custom_user_properties` command:

```
beectl> list_custom_user_properties
```

About the Timezone Attribute with LDAP Synchronization

Generally, all user attributes are controlled by the administrator and not by the user. For example, a user is unable to change their `given_name` or `job_title` attributes. Only an administrator can do that.

When you are making use of LDAP-based directory synchronization, any attribute mapped from the LDAP server to an Oracle Beehive attribute can only be changed in the LDAP server; you cannot change those attribute values for users in Oracle Beehive, but only in the LDAP server.

The `timezone` attribute is an exception. Because users will sometimes need to change their own time zones, such as when traveling, it would be unwieldy to require users to change their time zones in the LDAP directory, or to have to ask administrators to change it for them. For this reason, the `timezone` attribute is actually modelled in two parts:

- `DatasourceTimezone`: the value mapped from the LDAP server
- `timezone`: The time zone selected by the user

If `timezone` is null (not set) then the `DatasourceTimezone` is used for the users' time zone value.

If the user makes a change to their time zone such as by logging in to Oracle Beehive using Oracle Beehive Extension for Outlook (OBEO), where OBEO is set to a different time zone than their DatasourceTimezone, or by making the change using the Oracle Beehive Central user configuration page, then the `timezone` value will override the DatasourceTimezone. Any future changes to the DatasourceTimezone, such as from LDAP value changes, will update the `DatasourceTimezone` but will *not* update the user's `timezone` attribute.

Thus, the DatasourceTimezone is mainly useful for initial bootstrap purposes. For example, so that users get notifications for their correct time zone.

About Unique User Identifiers

A user account is a logical object in Oracle Beehive, essentially composed of a collection of required and optional data fields which are associated with each other. All entities, including user accounts, have a unique identifier, which is not normally displayed. However, user accounts also have a unique login ID. For most purposes in Oracle Beehive, whenever you need to identify a user account, you can use either the login ID or the identifier. You can use the `beectl list_users` command to find the unique login identifier of any user account:

```
beectl> list_users
```

This returns a list of users similar to the following:

```
Display Record: 1
===========================================
User Identifier: user=Bob.Smith
Family Name: Smith
Given Name: Bob
Parent Identifier: enpr=MyEnterprise

Display Record: 2
===========================================
User Identifier: user=Sarah.Jones
Family Name: Jones
Given Name: Sarah
Parent Identifier: enpr=MyEnterprise

Display Record: 3
===========================================
User Identifier: user=beeadmin
Family Name: BEEAdmin
Given Name: BEEAdmin
Parent Identifier: enpr=MyEnterprise
```

Each user is listed by first name, along with the unique user identifier, in the format `user=<login_id>`.

If you delete a user account and then re-create it, even using identical information and user identifier as before, it will have a new internal (non-displayed) object identifier, and will not be accidentally or automatically associated with settings or data objects from the previous user account.

About User `display_name` Values

In Oracle Beehive versions 1.4.3 and earlier, the user account `display_name` attribute is optional. However, beginning in Oracle Beehive version 1.5, `display_name` should always have a value (although the attribute is not mandatory because you can create an account without specifying a value). When creating new user accounts in Oracle Beehive 1.5 or later, if you do not supply a value for `--display_name`, the system will generate a value and populate the attribute automatically, according to the current rule. Additionally, when you edit an account that has a null value for `--display_name`, the system also will generate and populate the attribute. Such an account might exist if you upgraded from a previous version of Oracle Beehive to version 1.5 or later.

Note: If you are synchronizing your user directory with an LDAP-based directory, and you map display_name to a value in your LDAP directory, whenever the LDAP directory's value is null, Oracle Beehive will generate a display_name. When you then run the beectl validate_directory_entry command, a difference will be detected. Even if you run the command with the --commit option, Oracle Beehive will still re-generate the value, so the data inconsistency will continue to exist.

If the majority of your LDAP-based user accounts do not have a value for display_name, consider not including this attribute in your attribute map.

A display_name is generated according to the formula stored in the display_name_format attribute of the locale preference property on an Organization or the Enterprise. The attribute is set according to the nearest parent scope to the user account being created or edited.

Note: By default, if you have not set this attribute anywhere in a user's scope, the formula $G $F will be used (see Table 3–3, " display_name Generation Tokens" on page 3-14.).

The display_name_format must contain a string composed of any combination of the following codes, each of which is a token that will be substituted for the corresponding value for that user account. You can also include literal characters in the string, including commas, periods, and so forth.

Table 3–3 display_name ***Generation Tokens***

Token	Substituted value in generated display_name
$G	given_name
$M	middle_name
$F	family_name
$g	First character of given_name
$m	First character of middle_name
$f	First character of family_name
$P	prefix
$S	suffix
$J	job_title
$N	nickname

In addition, when generating the display_name, the following rules are applied:

- Two or more consecutive spaces are replaced with a single space.
- Leading and trailing spaces are removed.
- Empty parentheses characters ( ) containing zero or more spaces are removed.
- Leading and trailing commas are removed.

- Trailing space-dot sequences are removed (for example: " .").

To set or modify a `display_name_format`, use the `beectl add_preference_property` command. For example:

```
beectl> add_preference_property --set prfs=Locale,enpr=my_enterprise --name
display_name_format --type string --value "$g. $F"
```

> **Note:** Be sure to enclose the value in quotes if it contains spaces or special characters.

In this example, the preference property is set on the enterprise, and its value is set to `$g. $F`. If a user's `given_name` is `Abraham` and his `family_name` is `Smith`, then the system would generate a display_name of `A. Smith`.

> **Note:**
>
> - If there is an existing value for `display_name_format` on that preference set, it will be overwritten with your new value.
> - If the token string you specify evaluates to a null value for a given user, Oracle Beehive will instead use the default, `$G $F`. The default always evaluates to a non-null string because `--family_name` is a required user account attribute, so `$F` will always produce a non-null string.

To list existing settings on the Locale preference property for a given scope (Enterprise or Organization), use the `beectl list_preference_properties` command:

```
beectl> list_preference_properties --set prfs=Locale,enpr=my_enterprise
```

where `--set` is the identifier of the Locale preference property for a given scope.

> **Note:** If you have never manually set a `display_name_format` for a given scope, none will be listed for that scope. That is, there is no pre-seeded value on install.

About User Memberships

> **Note:** The organization memberships feature of user accounts is deprecated in Oracle Beehive 2.0. Oracle plans to remove this feature in a future release.

All users have at least one membership, to the enterprise or organization in which the user account was created. This is called the user account's context, and the containing enterprise or organization is called the Parent Identifier.

You can add additional memberships by using the `--organization` option with the `beectl add_user` or `modify_user` commands.

For example, a user named Fred Jones was created in the organization `orgn=Dev_QA,orgn=Dev,enpr=MyEnterprise`. Using the `beectl list_users --show ALL` command shows the following user account record:

```
Display Record: 3
===========================================
User Identifier: user=fred.jones
Family Name: Jones
Given Name: Fred
Display Name: Fred Jones
Parent Identifier: orgn=Dev_QA,orgn=Dev,enpr=MyEnterprise
Middle Name:
Job Title: Director
Department: Development
Company: Example.com
Suffix:
Prefix:
Nickname:
Profession:
Office Location: 101
Status: ENABLED
External Inbox: false
Effective External Inbox: false
Extended Enterprise User: false
Timezone: tmzn=Etc/GMT
Principals
===============
Principal Identifier: 13B2:6CCD:pcpl:F4AE2024BD5F4D1D9A7AC00C92AB964C000000000011
Principal Name: fred.jones
Type: PRIMARY=true
Addresses
=============
Type: BUSINESS_1
Value: mailto:fred.jones@example.com
Type: BUSINESS_1
Value: tel:16505551212
Memberships
===================
Member Of: orgn=Dev_QA,orgn=Dev,enpr=MyEnterprise
Member Of: orgn=Dev,enpr=MyEnterprise
Member Of: enpr=MyEnterprise
```

Note: This example shows results from Oracle Beehive version 1.5. Earlier versions of Oracle Beehive produce similar output, but with some different or missing attributes.

Since the user was created in a sub-organization called Dev_QA, which was contained by the Dev organization, which is in the MyEnterprise enterprise, the user has three memberships, one to each of these parent levels of scope.

Using the `beectl modify_user` command with the `--organization` option, an additional membership (to the organization `Install_QA`) is added:

```
beectl> modify_user --user user=fred.jones --organization orgn=Install_
QA,enpr=MyEnterprise
```

Now, listing the user shows that the memberships have been updated:

```
Display Record: 3
===========================================
User Identifier: user=fred.jones
Family Name: Jones
```

```
Given Name: Fred
Display Name: Fred Jones
Parent Identifier: orgn=Dev_QA,orgn=Dev,enpr=MyEnterprise
Middle Name:
Job Title: Director
Department: Development
Company: Example.com
Suffix:
Prefix:
Nickname:
Profession:
Office Location:
Status: ENABLED
External Inbox: false
Effective External Inbox: false
Extended Enterprise User: false
Timezone: tmzn=Etc/GMT
Principals
===============
Principal Identifier: 13B2:6CCD:pcpl:F4AE2024BD5F4D1D9A7AC00C92AB964C000000000011
Principal Name: fred.jones
Type: PRIMARY=true
Addresses
=============
Type: BUSINESS_1
Value: mailto:fred.jones@example.com
Type: BUSINESS_1
Value: tel:16505551212
Memberships
===================
Member Of: orgn=Dev_QA,orgn=Dev,enpr=MyEnterprise
Member Of: orgn=Install_QA,enpr=MyEnterprise
Member Of: orgn=Dev,enpr=MyEnterprise
Member Of: enpr=MyEnterprise
```

> **Note:** This example shows results from Oracle Beehive version 1.5. Earlier versions of Oracle Beehive produce similar output, but with some different or missing attributes.

Similarly, you can remove memberships by using the `beectl modify_user` command with the `--remove_membership` option:

```
beectl> modify_user --user user=fred.jones --remove_organization orgn=Install_
QA,enpr=MyEnterprise
```

The membership that was indicated is removed from the user account.

About Roles

To facilitate the potentially highly-complex structure of user privileges in a large Oracle Beehive deployment, you may make use of roles. A *role* is a collection of privileges and access types designed to fit commonly-used user responsibilities or positions.

For example, a Business Administrator role might grant privileges to provision user accounts, manage various types of artifacts such as archived messages, conferences, and discussions, create and modify user groups, and so forth. This role might not provide access to functions such as shutting down or restarting services, changing memory allocation, or viewing system log files.

You may assign one or more roles to any user account. Roles grant privileges, but do not (by default) revoke them. In other words, if a user account is granted a privilege not granted by a role, assigning that account a role does not revoke the privilege. Roles do not interact with each other: a user has a privilege if any role grants it, and does not have a privilege only if no roles grant it (and it has not been granted directly to that user).

Likewise, you may later grant or revoke any privilege, regardless of whether that privilege is granted to a user account by any assigned role.

Oracle Beehive is shipped with a selection of pre-created roles designed to fit many commonly-used user and management functions. You can modify the supplied roles, use them as templates to design your own roles, or create new roles from scratch, to suit the requirements of your organization.

For more information about managing roles, see "Managing Roles" on page 3-29, and "Managing Oracle Beehive Access Control".

About User Account Status

By default, all user accounts are set to *Enabled* and *Unlocked*. You may disable, lock, or delete a user account. Each of these states is treated specially by Oracle Beehive. When you delete an account, the system behaves according to the appropriate deprovisioning policy.

If a user account fails authentication (the password entered is incorrect) more than the maximum number of attempts defined in the password policy, the affected principal of the account is automatically locked.

About Special and System-Reserved Accounts

When Oracle Beehive is installed and configured, a special user account, called the BEEadmin account, is created automatically. You can use this account to perform configuration tasks prior to creating your own user accounts. Once you have created at least one account with system administration privileges, you may remove the BEEadmin account.

In addition to the BEEadmin account, when you act from the command line as the super-user, such as when you invoke `beectl`, objects may be indicated as owned by the *system actor*. There is no user account related to the system actor, but it is a valid actor for the purpose of evaluating privileges. In this case, the system actor has the *bypass* privilege, allowing total access to all entities, objects, and functions of Oracle Beehive.

Provisioning User Accounts

This section describes how to go about provisioning access to Oracle Beehive to your users. In the context of user accounts, provisioning means both creating an account, and the process by which the system enables users to access Oracle Beehive client functions. By default, a created account is automatically provisioned for access to all Oracle Beehive client functions.

The following sections describe user account creation and provisioning for both types of user accounts (enterprise and extended-enterprise). External users do not have user accounts; they are merely entries in various contacts lists.

The process for creating a user account depends on the nature of your deployment:

- An Oracle Beehive only deployment – making no use of external user directory or coexistence.
- A deployment where all user accounts are mastered in an external directory only – UDS synchronizes with the third-party LDAP-based directory server.

You may provision user accounts using the `beectl` command line tool. Using `beectl`, you may create user accounts one at a time, or in a batch process. To create user accounts in this manner, follow the instructions in "Provisioning User Accounts Using `beectl`" on page 3-19, or in "Bulk Provisioning User Accounts" on page 3-21.

You can also provision user accounts (one at a time) using Oracle Beekeeper.

If you are going to use an external third-party LDAP-based directory server, you should set up synchronization before you create or provision any users. Follow the instructions in Chapter 4, "Integrating an External User Directory with Oracle Beehive" in the *Oracle Beehive Integration Guide*.

This section contains the following topics:

- Provisioning User Accounts in Coexistence Environments
- Provisioning User Accounts Using `beectl`
- Bulk Provisioning User Accounts

Provisioning User Accounts in Coexistence Environments

In a coexistence environment, such as with Oracle Beehive coexisting with Microsoft Exchange, you may grant access to Oracle Beehive services to Exchange users, provision users in both systems, or provision users only in Oracle Beehive but synchronize them with Microsoft Exchange. Before you follow any of the user account creation and provisioning instructions in this module, see Chapter 3, "Integrating Microsoft Exchange Server 2003 or 2007 with Oracle Beehive" in the *Oracle Beehive Integration Guide*.

Provisioning User Accounts Using `beectl`

> **Note:** This section describes how to create an Oracle Beehive user account from the command line. Such a user is always mastered in UDS. If you are going to use an external, LDAP-based directory to master users, you should not create users in this manner. For instructions, see "Integrating and Synchronizing LDAP with Oracle Beehive," in the *Oracle Beehive Installation Guide* for your platform.

You can create a single user account, and provision it for Oracle Beehive, by using the `beectl add_user` command. You can list the syntax by entering the following command:

```
beectl> add_user --help
```

To create a user account for directory data mastered in UDS only, compose a `beectl add_user` command containing the required and optional fields described in "About User Account Fields" on page 3-4.

You must also specify a level of scope, such as the enterprise or an organization. You can find the enterprise identifier by using the `beectl list_enterprises` command:

```
beectl> list_enterprises
```

This will produce output similar to the following:

```
-----------------------------------------------
| Enterprise Name       | Identifier           |
-----------------------------------------------
| mycompany             | enpr=mycompany       |
-----------------------------------------------
```

The identifier is the complete string `enpr=mycompany`. You may find it convenient to copy this value to a text file for easy reference later.

You may also specify an organization as a user's scope. You can list organizations by using the `beectl list_organizations` command:

```
beectl> list_organizations --scope <parent enterprise or organization> [--recurse
TRUE|FALSE]
```

This will produce output similar to the following:

```
Organization name:                        ST
Description:                              Unknown
Identifier:                               orgn=ST,enpr=MyEnterprise
Allocated Quota:                          0
Hard quota in megabytes (MB):             Unlimited quota
Default sub organization hard quota in megabytes (MB):Unlimited quota
Default team workspace hard quota in megabytes (MB):Unlimited quota
Default team workspace soft quota in megabytes (MB):Unlimited quota
Default personal workspace hard quota in megabytes (MB):Unlimited quota
Default personal workspace soft quota in megabytes (MB):Unlimited quota
Active preference profile:
prfp=ActivePreferenceProfile,orgn=ST,enpr=MyEnterprise
```

> **Note:** This example shows results from Oracle Beehive version 1.3. Earlier versions of Oracle Beehive produce similar output, but with some different or missing attributes.

In this example, an organization called ST is created under the parent enterprise; it has no limits set on quota.

Since organizations may be nested (an organization can contain organizations), you may use the `--recurse TRUE` option to recursively list all sub-organizations within the scope you specify.

In addition to a scope, you must also (at a minimum) specify the user's `family_name`, `login_id`, and `login_password` attributes. Optionally you may specify many additional user account attributes.

> **Note:** Beginning in Oracle Beehive version 1.5, if you do not specify a display_name value, one will be generated. See "About User `display_name` Values" on page 3-13 for details.

Example 3–1 illustrates the syntax for a typical `beectl add_user` command to add a single user to UDS. The example shows all of the mandatory attributes, along with a selection of optional attributes. In this example, the password is obfuscated (instead of in plain text) because this command is being issued from `beectl` command-line mode (instead of shell mode).

Example 3–1 Creating a single user using `beectl`

```
./beectl add_user
--given_name User1GivenName
--family_name User1FamilyName
--login_id user@example.com
--login_password hioquery731419==
--scope enpr=example
--address BUSINESS_1:mailto:example1@example.com
--address BUSINESS_2:fax:121345222
--address BUSINESS_1:im:example2@example.com
--voice_principal 8881234567
--voice_pin 1234
--timezone tmzn=America/Denver
--locale "en_us"
--obfuscated
```

After you submit the command, the user account will be created. Submitting the command also triggers the provisioning policy, which determines how the user will be provisioned for Oracle Beehive. For more information about the provisioning policy, see "Managing Oracle Beehive Events and Policies".

The user should be able to log in to the system within a few minutes.

Bulk Provisioning User Accounts

Rather than create user accounts one at a time, you can create multiple accounts at once, by passing an XML-formatted document to the `beectl add_user` command. The XML standard format describes one or more user accounts, including, at a minimum, the required fields for each user. The file may also contain values for the optional fields. Creating many users at once using this method is referred to as bulk provisioning.

Certain elements in the XML file are order-sensitive and must appear in the correct order:

- Specify all extended enterprise users first, and then enterprise users.
- Within a `<user>` element, specify all `<givenname>` elements before any `<familyname>` elements.
- If you include a password element for a given principal, you must order the elements as shown in the following order: first the `<name>` element, then the `<password>` element, then the `<type>` element.

The provisioning policy is triggered after the creation of each user account.

To bulk provision user accounts:

1. Create an XML-formatted file containing the user account information.
2. Issue the `beectl add_user` command, using the `--file` option to specify the XML-formatted file:

   ```
   beectl> add_user --file <yourfile.xml>
   ```

If the XML file contains one or more invalid user definitions, those user accounts will not be created. However, all valid user accounts will be created. You will see a completion message similar to the following:

```
user23 failed: invalid initial_password attribute
user54 failed: duplicate email address

2 users failed
554 users added succesfully.
```

You can fix the invalid user entries and submit the XML file again. User accounts which were already created will simply fail to be re-created (because duplicate entries already exist in UDS), and the user entries which you fixed will be created.

For your convenience, an example user definition XML file is included at the end of this chapter. See "Example Bulk User Provisioning XML File" on page 3-42.

Using Gather Stats after Bulk Provisioning User Accounts

The first time you bulk load users, during the upload, the database statistics will quickly become out of date. This can lead to poor system performance, both during and after the upload process. If you are loading a large number of users, consider using the **gather stats** function in the Oracle Beehive database.

To improve performance, you can use the following SQL*plus command (as either the SYS or BEE_DATA user) to force the database to refresh statistics:

```
exec dbms_stats.gather_schema_stats('BEE_DATA',GRANULARITY=>'ALL');
```

Consider gathering statistics in any one of the following ways:

- Split the load into two files, with one small batch and one large. Run the small batch, then gather stats, and then do the larger batch.
- Run the gather stats while doing the bulk load all at once. This option will slow performance of the bulk load while gather stats is running.

In either case, if you are loading a large amount of users, this should provide a significant performance improvement after the database command completes.

Managing External Contacts

External contacts allow you to populate the enterprise Oracle Beehive shared address book with entries which are not user accounts. External contacts cannot log in to Oracle Beehive. You can create, modify, and delete external contacts using `beectl` commands.

> **Note:** Unlike user accounts, you can only add external contacts at the Enterprise level of scope. You cannot add external contacts to organizations.

List all existing external contacts using the `beectl list_external_contact` command:

```
beectl> list_external_contacts --show <show attributes(ALL|MORE)>
```

The `--show` option allows you to specify the level of detail of each listed external contact will be displayed.

You can also list a specific contact using the `--contact <identifier of external contact>` option.

You can locate a set of contacts with a common attribute value, by specifying one or more contact attributes. For example, list all contacts with a given department attribute:

```
beectl> list_external_contacts --department <department of the contact>
```

Create an external contact using the `beectl add_external_contact` command:

```
beectl> add_external_contact --family_name <[locale:]family name>
```

As with user accounts, the family name field is required.

> **Note:** As with user accounts, there are many optional fields you can use with external contacts. You can also preface most fields with a locale, used for display when an Oracle Beehive user is using a given locale setting for a client application. If you do not specify a locale, the default locale of `en_US` is used.

Modify an existing external contact using the `beectl modify_external_contact` command:

```
beectl> modify_external_contact { --contact <identifier of external contact> |
--email <Email address of the contact> }
```

You can specify the external contact to be modified using either its identifier or one of its e-mail addresses.

Delete an external contact using the `beectl delete_external_contact` command:

```
beectl> delete_external_contact { --contact <identifier of external contact > |
--email <Email address of the contact> }
```

You can specify the external contact to be deleted using either its identifier or one of its e-mail addresses.

> **Note:** You can send the `--purge` command to purge a deleted external contact, in the same manner as purging user accounts. See Deleting User Accounts for more details about the `--purge` command.

Managing User Accounts

You may need to perform a number of one-time, periodic, and day-to-day tasks related to user accounts. This includes changing or resetting status, creating or modifying provisioning policies, modifying individual user accounts, managing large numbers of accounts at once, and creating, modifying, and deleting roles.

> **Note:** In this section, the command line is used to demonstrate all procedures. However, you can also perform most of these activities using Oracle Beekeeper.

This section includes the following topics:

- Listing Users
- Changing Status
- Creating Custom User Properties
- Modifying User Accounts
- Managing Roles

Listing Users

Oracle Beehive provides `beectl` commands to list and search user accounts in your enterprise.

To list all users, use the `beectl list_users` command:

```
beectl> list_users
```

All users in the enterprise are listed, along with a summary of a few of their most commonly-referenced attributes.

You can list a specific user by specifying it with either the `--user` option or the `--email` options:

```
beectl> list_users --user <user identifier>

beectl> list_users --email <email address>
```

You can display a more detailed list of attributes by using the `--show` option:

```
beectl> list_users --show [ALL|MORE]
```

Searching for Users

> **Note:** Search parameters for users are case-insensitive.

You can search for users matching a certain pattern, using the `--match` (ANY | ALL) attribute with the `beectl list_users` command, combined with one or more user attributes to match, from the following list:

- --family_name
- --given_name
- --display_name
- --middle_name
- --job_title
- --prefix
- --suffix
- --nick_name
- --office_location
- --company
- --profession

- --department
- --manager
- --extended_enterprise_user
- --organization
- --property
- --address
- --status
- --is_deleted
- --created_by
- --modified_by

You may also indicate how much detail about each matching user to display, by using the --show (ALL | MORE) option.

For example, to search for all users which were created at the level of a particular organization, and to display all attributes of each such user:

```
beectl> list_users --match ALL --organization <organization identifier> --show ALL
```

Changing Status

You can change the status of one or more user accounts manually, or the status of accounts may change automatically. For example, if a user makes repeated unsuccessful attempts to type in a password, the user account may be locked (depending on the password policy).

In this section, the behavior set by the default policy is described, but you may modify such behavior by creating or modifying the applicable policy.

For more information about user account policies, see "Managing Oracle Beehive Events and Policies".

When an account is locked (or unlocked), the locked status applies to the primary principal. For example, if a user exceeds the maximum number of failed login attempts using an IMAP client, then the IMAP Protocol Principal will be locked, but the user can still login to a voicemail system using their VOICE Principal.

To manually set a user account to enabled or disabled, use the `beectl modify_user` command to modify the principal, setting the desired status in the `--attribute_value` parameter:

```
beectl> modify_user --status DISABLED --user user=user1

beectl> modify_user --status ENABLED --user user=user1
```

To lock or unlock (enable) a user account, use the following commands:

```
beectl> modify_user --user user=user1 --lock <principal to be
locked(PRIMARY|PROTOCOL|VOICE|ALL)>

beectl> modify_user --user user=user1 --unlock <principal to be
unlocked(PRIMARY|PROTOCOL|VOICE|ALL)>
```

To delete a user account, see the section "Deleting User Accounts" on page 3-30. You cannot set the MARKED_FOR_DELETE status; this status is set by the system when you send a user account delete command, or when a user mastered in an external LDAP-based directory is deleted from the directory (or no longer matches the query that marks a user for synchronization with Oracle Beehive).

The following account statuses are possible:

- Created Status
- Enabled Status
- Locked Status
- Disabled Status
- Marked for Delete and Deleted Statuses

Created Status

As soon as valid account creation values are passed in to Oracle Beehive and recorded in the database, the new account is set to created status. Oracle Beehive may still be performing provisioning functions, such as creating the personal workspace, adding the user account to other workspaces, and so forth.

In most cases you should not see the created status, and you cannot manually set an account to created status.

When the process of provisioning the new account is completed, the user account status is set to enabled (by default), locked, or disabled status.

Enabled Status

Most of the time, user accounts are set to enabled status. This means the user can log in to Oracle Beehive and make use of all provisioned services. By default, all user accounts are set to enabled when account creation and provisioning is completed. There is no explicit flag on an account called enabled. Instead, a lack of other status flags means the account is enabled.

Locked Status

The lock/unlock mechanism is primarily intended to aid with enforcing authentication rules, such as a maximum number of failed login attempts rule. This prevents brute-force login attacks (password guessing) from succeeding.

When a user account becomes locked, all of the user’s principals are locked, so that the user can no longer log in to Oracle Beehive, even if the correct password is entered. In all other ways, the account continues to be treated normally; messages sent to the user will continue to be delivered to the user’s inbox, the account name will continue to show up in address books, and so forth.

By default, a user account is set to locked status when the maximum number of failed login attempts is exceeded. This may occur if the user enters an invalid password repeatedly. The password policy sets the maximum number of failed login attempts. An account that is locked permanently when the maximum number of failed login attempts is exceeded can be unlocked by an administrator. For information about changing the password policy, see the chapter "Managing Oracle Beehive Events and Policies".

Administrators with sufficient privileges may also manually set a user account to locked status.

By default, manual action by an administrator is required to unlock a user account (by modifying the account and passing the `--unlock` option).

Disabled Status

When a user account is set to disabled status, the user cannot log in to Oracle Beehive. Additionally, the user account becomes unavailable to other users in the enterprise; it will no longer be listed in the enterprise contact list, messages sent to the user will be returned as undeliverable, and so on. Any artifacts owned by the user account continue to be owned.

The disabled status is useful for users who are only periodically allowed access to Oracle Beehive. You may also use the disabled status instead of deleting user accounts, such as when employees leave the company. This allows you to restore the account should the employee return to the company at a later date, with all previous stored artifacts and owned objects in place.

You may wish to create a custom policy that is triggered whenever a user account is set to disabled status, to determine the disposition of owned public workspaces, resources, and groups. For more information about creating custom policies, see the chapter "Managing Oracle Beehive Events and Policies".

Marked for Delete and Deleted Statuses

The marked for delete and deleted statuses are set by the system when deleting user accounts.

Caution: The deletion of an account is unrecoverable, unless you resort to restoring the Oracle Beehive system from backup. For this reason, Oracle recommends that you use the disabled status when you want to remove a user from the system, only deleting the account after a verification process. Effectively the only difference between a disabled and marked for delete user account, is that you can easily set a disabled user account back to enabled status.

When you initially begin the process of deleting an account (by issuing the `beectl delete_user` command), the user account is set to marked for delete status. At this point, the user deletion (deprovisioning) policy is triggered.

Once an account is set to marked for delete status, it cannot be recovered. The user cannot be added to workspaces or calendar events, cannot receive messages, and will not show up in contact lists.

The removal of account data is resource-intensive. You can periodically run the `beectl delete_user` command with the `--purge` option to purge accounts set to **marked for delete** status, and all their data, from the system. Oracle recommends running this command during minimum system usage periods, to avoid an impact on system performance during peak usage times. Once an account has been purged, it is set to **deleted** status, although you will not see the account (or its status) from most commands or displays.

Unlike a disabled account, when a user account is set to marked for delete, its unique user login identifier and any other unique attribute values are released, so these values can be re-used by a new account. Any new account created with the same attributes will not be associated with data from the deleted account.

Note: When user accounts are mastered in an external, LDAP-based directory, a delete action is triggered by a corresponding account deletion in the LDAP directory. Whenever UDS reads a user account deletion from the LDAP directory, it will automatically set that account to **marked for delete** status and begin the deprovisioning process. As with UDS mastered accounts, accounts mastered in an external LDAP directory that are marked for delete cannot be undeleted, and new accounts created with the same unique user attributes will not be associated with data from the previous, deleted account.

If you accidentally delete a user account, the only way to recover that account is to perform a system restore using a recent system backup archive.

For more information on deleting accounts, see "Deleting User Accounts" on page 3-30.

Creating Custom User Properties

In addition to the required and optional user account attributes provided by default in Oracle Beehive, you may create new attributes to suit the needs of your organization. You may create new optional or required attributes, or delete optional attributes.

Note: The `beectl` command `modify_user` is used to modify the values of custom properties. You use the `add_custom_user_property` and `delete_custom_user_property` commands for defining the properties themselves (that is, the metadata).

When you create a new user attribute, it is always of the type Property. Addresses and Principals are not considered custom attributes, even though by default, any specific address or principal field of an account might not be set.

To see a list of custom attributes, use the `beectl list_custom_user_properties` command:

```
beectl> list_custom_user_properties
```

To create a new custom field, use the `beectl add_custom_user_property` command:

```
beectl> add_custom_user_property --name <attribute_name> --type <property_type> [
--description <description> ]
```

For `<property_type>`, valid options are BOOLEAN, COLLABID, DATETIME, DOUBLE, and STRING.

To delete a custom user property, use the `beectl delete_custom_user_property` command:

```
beectl> delete_custom_user_property --name <name>
```

Modifying User Accounts

Whenever you change a user account’s attributes, you are modifying that user’s account. When you make changes to group membership, you are actually modifying the group; the same is true of resources, workspaces, and so forth.

If a user account is mastered in UDS only, you can modify all of the user's attributes stored in Oracle Beehive.

For user account management for directory data mastered in an external directory (a third party directory server is synchronized with UDS), you should modify the account attributes directly in the external third party directory. When a user's attributes are modified in the third party directory server, a synchronization process is initiated with UDS and the user's attributes in the UDS directory will reflect the same modified attributes.

When a user account directory data is mastered in an external directory, but the attributes you want to modify are not stored by the external directory, you should modify those Oracle Beehive-specific attributes using Oracle Beehive.

To modify a user account in Oracle Beehive, use the `beectl modify_user` command:

```
beectl> modify_user { --user <User Identifier> | --email <Email address of the
user> }
```

You can provide the user identifier, or any of the user's e-mail addresses (addresses of scheme `mailto`) to identify the user account you want to modify.

There are many user account attributes. See Table 3–1, " Default User Account Fields" for a complete list. See the `modify_user` command reference for details and syntax.

For many user account attributes, multiple values are permitted. For example, a user account can have multiple given names (for various different locales). If you want to remove or modify the value of a user account attribute which already has a value, you can use the + (plus) or - (minus) signs. These signifiers work for all of the following attributes: family_name, given_name, display_name, middle_name, prefix, suffix, nick_name, property, address, login_id, voice_principal, and protocol_principal.

If you do not specify a + or - option, the add action is assumed, and any existing value of the same attribute will be overwritten.

Note: The `beectl` interface will not allow an option value to begin with the - (minus sign) character. As described in the `beectl --help`, you can use an alternate format to provide an option beginning with the -; by prepending `ESCAPE:` to the option text. For example:

```
beectl> modify_user --user user=user1 --nick_name ESCAPE:-Bob
--nick_name +Rob
```

Managing Roles

Roles are a useful tool for defining a common or shared level of privileges across an organization. For example, you may choose to grant a similar level of Oracle Beehive access to all of the vice-presidents in your company, or to all faculty at your university. You may define a role for a single user, a subset of users, or all users in your organization.

Roles are an intrinsic part of Oracle Beehive access control. See "Managing Oracle Beehive Access Control" for details about creating, modifying, and assigning roles to user accounts.

Deleting User Accounts

User account deletion is complex, because typically user accounts may have ownership of numerous artifacts, the disposition of which must be resolved before the user account can be fully erased from the system.

Note that, short of deletion, a user account may be locked, by setting it to locked status (temporarily removing the ability to log in), or disabled (removing access and removing the user from groups, preventing receipt of messages, and so forth). The difference between disabled and deleted is singular: a disabled account may be re-enabled, but a deleted account may not be undeleted.

For more information about user account statuses such as disabled and locked, see "About User Account Status" on page 3-18. For instructions on changing user account status, see "Changing Status" on page 3-25.

Caution: Oracle recommends making use of the disabled status when users leave your organization, reserving delete until some waiting period after they have left. This allows you to easily recover (by setting to enabled) any account in the event of an error, such as an incorrect user account being identified for deletion.

Deleting a user account does not delete data in the system, including user-owned data such as messages or files in the personal workspace. Such data remains in the system until you explicitly purge it. Purging user data is resource-intensive, so Oracle recommends you purge user data during a minimum-use period, such as late at night, to avoid causing a slowdown in system responsiveness.

When you delete a user account, you must either manually decide on the disposition of owned artifacts, or make use of a deprovisioning policy that automatically determines the disposition of owned artifacts according to policy rules. Ensure that all of a user's artifacts in Oracle Beehive are properly reassigned before purging the data.

For more information about managing artifacts, see "Managing Oracle Beehive Workspaces".

The procedure for deleting an account varies if you are using an external directory (a third-party LDAP-based directory server synchronized with UDS).

When deleting a user account mastered in UDS only, you can delete a user from UDS using `beectl delete_user`. When deleting a user account mastered in an external directory (a third-party directory server is synchronized with UDS), delete the account from the external directory server directly, using the external directory management tools.

Note: While a user account deleted in a third-party directory is automatically deleted by UDS, it is not purged. You must still manually purge the user, as in the following procedure Step 4. Until you purge the user account data, a new user created in the external directory with the same user name or ID will fail to synchronize with UDS.

When a user account is deleted from a a synchronized external third-party user directory, UDS automatically responds by deleting the UDS user account.

Note: For OpenLDAP 2.4.X, UDS synchronization relies on the `modifytimestamp` field for user/group synchronization. This field is stored with each user/group entry in OpenLDAP. Using this field, Oracle Beehive can detect if the user is created/modified. However, when the user is deleted the entry is permanently removed from the OpenLDAP repository. Consequently, there is no way for UDS to discover that a user account has been removed.

To work around this issue, you can periodically use the `beectl validate_directory_entry` command with the `--delete` command to clean up Oracle Beehive and remove users that have been deleted from OpenLDAP:

```
beectl> validate_directory_entry --delete --profile openldapprofile
--commit
```

You will see output similar to the following:

```
Total number of entries: 511
Number of entries to delete: 2


Number of successfully deleted entries: 2
Number of failed entries: 0
```

In this example, two user accounts in UDS were identified for deletion.

To manually delete a user account using `beectl`:

1. Set the user account to disabled status using the `beectl modify_user` command:

   ```
   beectl> modify_user { --user <User Identifier> | --email <Email address of the
   user> } --status DISABLED
   ```

2. Identify Oracle Beehive objects owned by the user, such as messages, groups, resources, folders, files, and workspaces, and dispose of them according to the requirements of your organization; by deleting, archiving, or reassigning ownership as appropriate. Commands useful for performing these actions include:

   - modify_group
   - modify_team_workspace
   - modify_personal_workspace
   - modify_resources
   - delete_group
   - delete_workspace

3. Delete the user account using the `beectl delete_user` command. You can specify the user using its unique user identifier, or its primary e-mail address:

   ```
   beectl> delete_user { --user <user_identifier> | --email <user_email> }
   ```

4. At your convenience, purge user data.

First, use the `beectl list_users` command with the `--is_deleted` option to list all accounts that are deleted and eligible to be purged:

```
beectl> list_users --is_deleted
```

Then, purge individual accounts using the `beectl delete_user` command with the `--purge` option. You may want to ensure that a system backup has taken place, to ensure rollback and recovery of user data is possible. Because purging is resource intensive, you should perform purge operations during a period of minimum system usage:

```
beectl> delete_user -{ --user <user_identifier> | --email <user_email> }
--purge
```

Deleting external contacts follows the same procedure as outlined for user accounts, except that external contacts do not have a status, so you cannot disable them. Delete an external contact by issuing the `beectl` delete_external_contact command:

```
beectl> delete_external_contact { --contact <identifier of external contact > |
--email <Email address of the contact> }
```

Then, purge the external contact data using the `--purge` option:

```
beectl> delete_external_contact { --contact <identifier of external contact > |
--email <Email address of the contact> } --purge
```

Managing Groups

A group is a logical construct in Oracle Beehive. One or more users are members within the group, and one or more members may have group management privileges. Many groups are owned by a particular user, as well.

> **See Also:** Group templates and XML examples are provided in "Oracle Beehive XML File Reference", in the *Oracle Beehive Administrator's Reference Guide*

This section contains the following topics:

- About Groups
- Listing groups
- Creating and Modifying Groups
- Enable Profiling on Groups
- Dynamic Group Query Construction
- Group Inheritance

About Groups

Oracle Beehive allows you to create all manner of nesting and overlapping logical groupings of user accounts. As with user accounts, a group has a globally unique collabID, a unique identifier, and some optional fields.

Groups may contain sub-groups, users, and resources.

Groups can themselves be contained by the enterprise, an organization, another group, or a workspace. Groups may only contain members at the parent level of scope; for example, a group created within an organization may only contain members from that organization. Likewise, a group contained by a workspace may only contain members of that workspace.

Groups are useful because you can use them as an alias to perform various collaboration activities. For example, you can send messages to everyone in a group, invite everyone in a group to a meeting or event, and assign everyone in a group to a workspace. Groups are also useful for managing access control; you can grant or revoke privileges based on group membership, and you can use groups as specified actors when creating Access Control Entities (ACEs).

When a user account is assigned to a group, it normally inherits any attributes (access privileges) of that group.

You can define access privileges on a group level, and then assign user accounts to such a group, thereby granting those privileges to all members of the group. By default, a user is considered to be granted a privilege if either they are explicitly granted it (the privilege is granted directly to their user account) or implicitly granted it (the user is a member of a group which is granted the privilege, or has a role which grants that privilege).

> **See Also:** For more information about managing access control, see the chapter "Managing Oracle Beehive Access Control".

You may create sub-groups within larger groups. There is no practical limit to how many nesting groups you may create. A user belonging to a sub-group is considered to also belong to any super-group that contains that sub-group.

A user account may belong to any number of groups.

Groups always have an owner. The owner of a group has privileges to add and remove membership of the group, as well as alter its editable fields. (Additional user accounts may be granted group management privileges as well.) When you create a group from the command line, you should usually add a user and assign ownership to that user. Until you do so, the owner of the group is the system actor, which means, it can only be manipulated by the command line user.

When a user account is deleted, it is removed from all groups to which it formerly belonged. When a group is deleted, membership in that group is removed from all user accounts to which it applied. When a user account having ownership of a group is deleted, ownership of that group is either manually reassigned (the account is not deleted until all decisions are made, using a set of human workflows), or reassigned automatically based on the deprovisioning policy defined for Oracle Beehive.

Administration privileges for a group may be assigned on a per-group basis. In other words, a given user account may have administration privileges for only specific groups (at any level of super-group or sub-group), without gaining administrative privileges over any other groups. By default, the creator of a group gains administrative privileges only for that group (and all sub-groups contained by it). For more information about privileges, see "Managing Oracle Beehive Access Control".

You can also base privileges and roles on group membership. For example, you can grant Oracle Beehive administrative privileges to everyone in a Beehive Managers group. Or, you could assign everyone in such a group a role you created called the business-administrator role.

You can also use group membership as a variable when writing policies. For example, you could define a provisioning policy that grants extra personal workspace quota to members of the Maintenance group.

Note: The ALL_USERS group is a pre-seeded group created during Oracle Beehive installation. It automatically contains every user of Oracle Beehive. It is very useful for assigning privileges to all users, broadcasting messages to all users, and other such global operations. Do not delete the ALL_USERS group.

There are two types of groups:

- Static groups
- Dynamic groups

Static groups

A static group has an explicit list of members. Users must be directly added to the group, and directly removed from the group.

Dynamic groups

Dynamic groups have a membership defined by a query, so that users fitting whatever criteria is being queried are automatically made members of the group. The dynamic group query is based on any combination of user attributes, properties, or addresses.

For example, all users with a particular manager could belong to a dynamic group defined by a query against the Manager attribute of user accounts. Whenever a user's manager attribute is changed to that particular manager, that user is automatically added to the group. Likewise, whenever a user's manager attribute changes (the user switches to a different manager), that user is removed automatically from the group.

Note: Dynamic groups may also have individual users defined as members, by defining a query which returns a specific user account.

Listing groups

You can list all groups by using the `beectl list_groups` command:

```
beectl> list_groups
```

You can list a specific group, using the `beectl list_groups` command with the `--group` option:

```
beectl> list_groups --group <Identifier of the group>]
```

You can list the members of a group by using the `beectl list_groups` with the `--show MEMBERS` option:

```
beectl> list_groups --group <Identifier of the group> --show MEMBERS
```

Searching for Groups

You can search for groups matching a certain pattern, using the `beectl list_groups` command with the `--match (ANY|ALL)` option, combined with one or more group attributes to match from the following list:

- --name
- --description
- --scope

- --organization
- --property
- --address
- --is_deleted
- --created_by
- --modified_by

Note: Search parameters for groups are case-insensitive.

You may also indicate how much detail about each matching group to display, by using the `--show (ALL|MORE|MEMBERS)` option.

For example, to search for all groups which were created at the level of a particular organization, and to display all attributes of each such group:

```
beectl> list_groups --match ALL --organization <organization identifier> --show
ALL
```

Note: The `--show ALL` option lists details for all members in the group, in addition to the group's description and status.

Creating and Modifying Groups

Administrators and privileged users may create any number of groups, and assign or invite users to them. As an administrator, you will probably create some base groups, and reassign ownership to various users, such as vice presidents or directors. Additionally, group creation privileges may be granted to some or all users. In this case, a user with group creation privileges may create any number of groups. Users may create public groups, which are listed publicly (all users can see that the group exists). Public groups require users to request membership from an approver.

Groups are nested, such that each group is a sub-group of some other context. Some groups are top-level groups, meaning they do not belong inside any other groups, but still belong inside the enterprise, an organization, or a workspace. Logically, all such groups also belong inside one enterprise-wide supergroup (the ALL_USERS group). Owners and those with administrative and group creation rights can create a sub-group inside any group which they control.

You can create or modify a group by creating an XML-formatted file that defines the group, and then importing the file from the command line.

When creating a static group, you can add specific users to a group by specifying them in the XML file (for static groups). For dynamic groups, users are added to the group according to the query criteria as soon as the group is created.

Note: In a static group template, users are specified in several ways, including by identifier and by e-mail address. Each user must only be specified once in a group template, however. Specify a user by any one method. For example, if you include both a user's identifier, and that user's e-mail address, group creation will fail.

To create a group, use the `beectl add_group` command:

```
beectl> add_group --file <filename>
```

If the group is created successfully, you should see output something like:

```
Successfully added 1 groups.
Failed to add 0 groups.
Total groups 1
```

To modify a group, use the `beectl modify_group` command:

```
beectl> modify_group --file <filename>
```

> **Note:** When modifying a group, the XML file must specify the full group CollabID in the `group` element.

Example XML-formatted files for group creation are provided in "Oracle Beehive XML File Reference", in the *Oracle Beehive Administrator's Reference Guide*.

Enable Profiling on Groups

You can enable profiling on a user group using the `beectl modify_expertise_profiling_configuration` command.

```
beectl modify_expertise_profiling_configuration --file <file name>
```

Where:

`<file name>` is the name of the XML file that contains information to enable profiling on the group. Following is an example XML file.

```
<?xml version = '1.0' encoding = 'UTF-8'?>
   <ExpertiseProfilingConfiguration
xmlns="http://xmlns.oracle.com/beehive/expertise">
           <all_users_group>
                   <profiling_enabled>true</profiling_enabled>
                   <bootstrap_enabled>true</bootstrap_enabled>
                   <terms_searchable>true</terms_searchable>
                   <relationships_searchable>true</relationships_searchable>
           </all_users_group>
  </ExpertiseProfilingConfiguration>
```

Dynamic Group Query Construction

To create a dynamic group, you first create an XML-formatted file, which contains the group creation information.

When creating dynamic groups, you can include one or more predicates, which define criteria for inclusion in the group. Each predicate specifies a single user account attribute, address, or property. In complex queries, predicates are linked using an operator.

> **Note:** You can create dynamic groups based on a single custom user account property. Multiple custom user account properties cannot be used to create dynamic group queries.

This section includes the following topics:

- Queries Using Attributes
- Queries Using Addresses
- Queries Using Properties
- Using Wildcards with Query Predicate Values
- Query Predicate Operators
- Example Queries

Queries Using Attributes

Predicates including an account attribute use the following format:

```
<predicate>
    <attribute>
      <name>FAMILY_NAME</name>
      <value>Example%</value>
    </attribute>
</predicate>
```

In this example, the attribute `FAMILY_NAME` is used, and a value is entered in the `value` element. User accounts with the `FAMILY_NAME` value specified will be matched by the query.

The following user account attributes can be used in the `name` child element of an `attribute` parent element:

- FAMILY_NAME
- GIVEN_NAME
- OFFICE_LOCATION
- ASSISTANT
- MANAGER
- COMPANY
- DEPARTMENT
- PROFESSION
- NAME
- ORGANIZATION
- TIME_ZONE

Notes:

- The names of the attributes are case-sensitive.
- The MANAGER, ASSISTANT, TIME_ZONE, and ORGANIZATION attributes accept identifiers only. Pattern-based searches of these fields is not available.

The content of the value element is the string that Oracle Beehive will attempt to match when performing the query.

Queries Using Addresses

Predicates including an address attribute use the following format:

```
<predicate>
  <address>
    <uri>scheme:value</uri>
    <type>type</type>
  </address>
</predicate>
```

User account address fields contain the following three parts:

- Scheme
- Value
- Type

The `uri` element contains the scheme and value of the address, separated by a `:` (colon) character. The following schemes are available:

- mailto
- tel
- fax
- ftp
- http
- https
- im
- imap
- ldap
- news
- nntp
- oraalert
- oraassistantphone
- oracallback
- oraisdn
- oramobile
- oracarphone
- orapager
- orapostal
- orapush
- oraradio
- orasms
- oratelex
- orattyttd
- oravmail

- pres
- sip
- urn
- xmpp

The type element contains one of the following:

- BUSINESS_X
- PERSONAL_X
- OTHER_X
- PROXY_X

For BUSINESS, PERSONAL, and OTHER, X is an integer from 1 to 5. For PROXY, X is an integer from 1 to 25.

Queries Using Properties

Predicates including an address attribute use the following format:

```
<predicate>
  <property>
    <name>exampleName</name>
    <value>exampleValue</value>
  </property>
</predicate>
```

Properties are defined as name value pairs. The name element can contain any property.

Note: The names of the properties are case sensitive.

Using Wildcards with Query Predicate Values

You can use an exact string in the value element of a query, but you can also use wildcards to match a pattern. Wildcards are formatted in the same manner as SQL LIKE wildcards. The wildcards % (percent sign) and _ (underscore) are allowed. The % wildcard matches one or more characters, while the _ wildcard matches exactly one character. Prefixing either wildcard symbol with a \ (slash) character will cause it to be treated as a literal (not a wildcard) in a query.

For example, if the following values of OFFICE_LOCATION exist: A15, A156, B156, B1568, and Main_12:

- %15% will match A15, A156, B156, and B1568
- A1_ will match A15
- A15% will match A15 and A156
- Main_1% will match Main_12

Query Predicate Operators

Operators link predicates together into a complete query. You can perform a simple query with a single predicate by using <operator type="NONE">, or you can perform a more advanced query by using <operator type="AND"> or <operator type="OR"> to assemble two or more predicates.

To create a single-predicate query, use the following format:

```
<operator type="NONE">
  <predicate>
.
.
.
  </predicate>
</operator>
```

Place the predicate content into the predicate element.

To create a complex query with two or more predicates, use one of the following formats:

```
<operator type="AND">
   <predicate>
.
.
.
   </predicate>
   <predicate>
.
.
.
   </predicate>
</operator>
```

or

```
<operator type="OR">
   <predicate>
.
.
.
   </predicate>
   <predicate>
.
.
.
   </predicate>
</operator>
```

Nested predicates are supported up to any level.

Example Queries

The following examples demonstrate how to assemble a dynamic group query using one or more query predicates, linked by query operators.

Example 3–2 creates a dynamic group that includes all users whose office location is HQ.

Example 3–2 Dynamic Group Simple Attribute Query

```
<operator type="NONE">
     <predicate>
       <attribute>
         <name>OFFICE_LOCATION</name>
         <value>HQ</value>
       </attribute>
     </predicate>
```

```
</operator>
```

Example 3–3 creates a dynamic group that includes all users whose office location includes the string HQ.

Example 3–3 Dynamic Group Simple Attribute Query with Wildcards

```
<operator type="NONE">
     <predicate>
       <attribute>
         <name>OFFICE_LOCATION</name>
         <value>%HQ%</value>
       </attribute>
     </predicate>
</operator>
```

Example 3–4 creates a dynamic group that includes all users whose manager is a user with the specified CollabID.

Example 3–4 Dynamic Group Simple Attribute Query Using CollabIDs

```
<operator type="NONE">
   <predicate>
     <attribute>
       <name>MANAGER</name>
       <value>6BAE:44D9:user:DCEDF8D6310B4AE69911C7607BBADBE4000000000000</value>
     </attribute>
   </predicate>
</operator>
```

Example 3–5 creates a dynamic group that includes all users whose manager is a user with the specified CollabID, or whose office location includes the string HQ.

Example 3–5 Dynamic Group Complex Query

```
<operator type="OR">
   <predicate>
      <attribute>
        <name>MANAGER</name>
        <value>6BAE:44D9:user:DCEDF8D6310B4AE69911C7607BBADBE4000000000000</value>
      </attribute>
   </predicate>
   <predicate>
      <attribute>
        <name>OFFICE_LOCATION</name>
        <value>%HQ%</value>
      </attribute>
    </predicate>
</operator>
```

Group Inheritance

When a user is added to a group, by default that user is considered a member of all super-groups to which that group belongs. Since access control (aside from group privileges) may be based on group membership, it is important that conflicts in privileges are resolved properly.

For example, if a super-group called Development contains a sub-group called QA, members of the QA group are automatically granted privileges granted to the

Development group. Any access control that allows access to members of the Development group, will by inference also allow access to members of the QA group. You could explicitly prevent such access, by granting access to members of Development only if they are not also members of the QA group.

By default, if a super-group grants a privilege, and a sub-group does not explicitly restrict it, members of the sub-group are granted the privilege.

You can override inheritance by explicitly granting or revoking privileges from a sub-group, or to individual members of the group.

For more information about access control, see "Managing Oracle Beehive Access Control".

Example XML Files

This reference section contains example XML-formatted files for use in user management.

Example Bulk User Provisioning XML File

This is an example XML-formatted file for bulk user provisioning. This particular example contains three users. Refer to the XSD file for a complete definition. The XSD may be found at `$ORACLE_HOME/beehive/templates/uds/User.xsd`

Notes:

- You must change the collabIDs used in all `<scope>` and `<membership>` elements to match those of your enterprise or organization
- Values for password elements must meet the default password policy requirements
- Certain elements in the XML file are order-sensitive and must appear in the correct order:

  Specify all extended enterprise users first, and then enterprise users

  Within a <user> element, specify all <givenname> elements before any <familyname> elements

  If you include a password element for a given principal, you must order the elements as shown in the first user in the example: first the `<name>` element, then the `<password>` element, then the `<type>` element

Example 3–6 Bulk User Provisioning XML File

```
<?xml version = '1.0' encoding = 'UTF-8'?>
<users xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <user>
      <scope>
         <cen>enpr=MyEnterprise</cen>
      </scope>
      <user_type>EXTENDED_ENTERPRISE_USER</user_type>
      <givenname>Robert</givenname>
      <familyname>Holmes</familyname>
      <name>Robert</name>
      <principals>
         <add>
            <principal>
               <name>+16505551234</name>
               <password>8675309</password>
               <type>VOICE</type>
            </principal>
         </add>
         <add>
            <principal>
               <name>rholmes</name>
               <password>password</password>
               <type>PRIMARY</type>
            </principal>
         </add>
      </principals>
      <addresses>
         <add>
            <item>
               <address>TEL:+16505551234</address>
               <addresstype>BUSINESS_1</addresstype>
            </item>
         </add>
         <add>
            <item>
               <address>IM:rholmes@example.com</address>
               <addresstype>BUSINESS_1</addresstype>
            </item>
         </add>
         <add>
            <item>
               <address>MAILTO:rholmes@example.com</address>
               <addresstype>BUSINESS_1</addresstype>
            </item>
         </add>
         <add>
            <item>
               <address>TEL:+16505551444</address>
               <addresstype>PERSONAL_1</addresstype>
            </item>
         </add>
      </addresses>
   </user>
   <user>
      <scope>
         <cen>enpr=MyEnterprise</cen>
      </scope>
      <user_type>ENTERPRISE_USER</user_type>
      <givenname>Mary</givenname>
      <familyname>Langdown</familyname>
      <name>Mary</name>
      <principals>
         <add>
            <principal>
               <name>+15145554321</name>
               <type>VOICE</type>
            </principal>
         </add>
         <add>
            <principal>
```

```
            <name>mary.langdown</name>
            <type>PRIMARY</type>
         </principal>
      </add>
   </principals>
   <addresses>
      <add>
         <item>
            <address>TEL:+15145554321</address>
            <addresstype>BUSINESS_1</addresstype>
         </item>
      </add>
      <add>
         <item>
            <address>IM:mary.langdown@example.com</address>
            <addresstype>BUSINESS_1</addresstype>
         </item>
      </add>
      <add>
         <item>
            <address>MAILTO:mary.langdown@example.com</address>
            <addresstype>BUSINESS_1</addresstype>
         </item>
      </add>
      <add>
         <item>
            <address>TEL:+16505559876</address>
            <addresstype>PERSONAL_1</addresstype>
         </item>
      </add>
   </addresses>
</user>
<user>
   <scope>
      <cen>enpr=MyEnterprise</cen>
   </scope>
   <user_type>ENTERPRISE_USER</user_type>
   <givenname>Thanh</givenname>
   <familyname>Tran</familyname>
   <name>mark</name>
   <principals>
      <add>
         <principal>
            <name>+16505556300</name>
            <type>VOICE</type>
         </principal>
      </add>
      <add>
         <principal>
            <name>thanh.tran</name>
            <password>Password2</password>
            <type>PRIMARY</type>
         </principal>
      </add>
   </principals>
   <addresses>
      <add>
         <item>
            <address>TEL:+16505556300</address>
            <addresstype>BUSINESS_1</addresstype>
            </item>
         </add>
         <add>
            <item>
               <address>IM:thanh.tran@example.com</address>
               <addresstype>BUSINESS_1</addresstype>
            </item>
         </add>
         <add>
            <item>
               <address>MAILTO:thanh.tran@example.com</address>
               <addresstype>BUSINESS_1</addresstype>
            </item>
         </add>
         <add>
            <item>
               <address>TEL:+16505551454</address>
               <addresstype>PERSONAL_1</addresstype>
            </item>
         </add>
      </addresses>
   </user>
</users>
```

4

Managing Oracle Beehive Resources

This module describes how to manage resource accounts in Oracle Beehive. The procedures described in this module for managing Oracle Beehive resources use the `beectl` command-line interface. You can also perform many of these tasks using Oracle Beekeeper.

See Also:

- For more information about the `beectl` commands used in this chapter, see Chapter 2, "Oracle Beehive Command-Line Utility" in the *Oracle Beehive Administrator's Reference Guide*
- You can also manage resources using Oracle Beekeeper. For complete instructions, see "Managing Resources" in Chapter 3, "Managing Enterprises," of *Oracle Beekeeper Online Help*.

This module includes the following sections:

Introduction to Oracle Beehive Resources

An Oracle Beehive bookable resource is an entity that users can search for, reserve, and use for a specified period of time, such as a conference room or a projector. In Oracle Beehive, users create, search for, and manage resources through the Resource Directory Service, and reserve them using the Time Management Service. Functionality provided by both services is exposed through the calendar functions of various Oracle Beehive clients.

Although it is usually done using team workspaces, bookable resource accounts can also be used as an alternative method to create calendars for tracking related enterprise-wide information, such as employees' travel schedules.

External resources are resources which are tracked in Oracle Beehive similarly to bookable resources, but which users cannot book using Oracle Beehive clients.

This section includes the following topics:

- About Oracle Beehive Bookable Resource Accounts
- Oracle Beehive Resource Attributes

About Oracle Beehive Bookable Resource Accounts

Similarly to an Oracle Beehive user, a bookable resource account has a calendar container associated with the account. The calendar container contains all events to which the bookable resource has been invited.

Unlike an Oracle Beehive user, it is not possible to authenticate with Oracle Beehive using a bookable resource account. Bookable resource accounts are managed by administrators, and users who have sufficient privileges.

Bookable resources can be set up to permit reservations on a first come first served basis to prevent double-bookings, or be set as open to permit more than one reservation at a time. You can also configure bookable resources so that only certain users are allowed to reserve them.

Oracle Beehive Resource Attributes

When an Oracle Beehive resource is created, a group of attributes are available to configure the resource properties. Some attributes are mandatory, while others are optional, populated by Oracle Beehive at the time of resource creation, or populated when a dependent attribute is modified. Table 4–1, " Oracle Beehive Resource Attributes" lists available Oracle Beehive resource attributes.

Table 4–1 Oracle Beehive Resource Attributes

Attribute Name	Description	Required Attribute	Accepted Values
Scope	You can create a resource at the Enterprise level (by default), or you can create it at the level of an Organization. Regardless of scope, all resources are visible to all users in the Enterprise.	No	A valid enterprise or organization ID
Resource ID	A `Resource ID` is generated when the resource is created. A Resource ID cannot be specified by administrators at the time of creation, nor modified thereafter.	Yes	N/A
Identifier	Specifies a resource identifier. You can reference a resource using its `Identifier` or its `Name`. This attribute can be a number assigned by your organization to the attribute, or an alternative identifier. For example, a room number or a projector serial number.	No	An alphanumeric string **Note:** The maximum number of characters allowed is 32.
Name	Specifies the display name for the resource account. You can reference a resource using its `Name` or its `Identifier`. This will be the name displayed when searching for a resource. It must be unique.	Yes	An alphanumeric string **Note:** The maximum number of characters allowed is 1000.
External	Specifies an external resource. If this option is not specified, a bookable resource will be created.	No	None (this command-line option takes no arguments).

Table 4–1 (Cont.) Oracle Beehive Resource Attributes

Attribute Name	Description	Required Attribute	Accepted Values
Description	Describes any additional characteristics of the resource.	No	An alphanumeric string **Note:** The maximum number of characters allowed is 4000.
Location	Specifies the location of the bookable resource. This attribute is broken down into sub-attributes, including description and time zone. **Note:** Although each resource can only have one `Location`, some clients may permit multiple values for certain sub-attributes: for example, multiple GPS coordinates for a vehicle.	No	A complex attribute (see `Description`)
Phone	Specifies the phone number of the resource.	No	A valid telephone number
Fax	Specifies the fax number of the resource.	No	A valid fax number
Website	Specifies the Web page of the resource.	No	An alphanumeric string
E-mail address	Specifies the e-mail address of the resource. The e-mail address must be unique. This e-mail address is primarily used for addressing purposes using standard-based clients such as CalDAV and Mozilla Thunderbird. No email inboxes will be created for the resource, and the resource account cannot receive e-mail messages. Oracle recommends that the e-mail address specified for a resource reside in the same domain as the e-mail addresses specified for the remaining organization.	Yes	An e-mail address conforming to the standard format
Postal Address	Specifies the postal address of the resource.	No	A complex string in a special format
Approver	Specifies a user who is the resource approver. This attribute acts as a pointer to an Oracle Beehive user. Once specified, the approver can manage the resource's event invites. **Note:** Multiple approvers can exist for a single bookable resource. With `beectl`, you can add additional approvers using the `--add_approver` option.	Yes	A valid Oracle Beehive user
Capacity	Specifies a description of the capacity of the bookable resource. This attribute is especially useful for conference rooms, offices, or equipment, such as company vehicles, that have a limited capacity. **Note:** Oracle Beehive does not restrict participant numbers based on this attribute.	No	A positive integer

Resource Type	Specifies a type for a bookable resource. The following mutually exclusive options are available: `Room`, `Equipment` or `Other`. Select the value most appropriate for the resource.	Yes	`Room`
			`Equipment`
			`Other`

Table 4–1 (Cont.) Oracle Beehive Resource Attributes

Attribute Name	Description	Required Attribute	Accepted Values
Booking Characteristics	Specifies the booking characteristics of a bookable resource. It can have the following two values: ■ Open ■ First-Come-First-Serve	No	O (for Open) or F (for First-Come-First-Serve)
Booking Info	Specifies information (for users) about booking this resource.	No	An alphanumeric string
Time Zone	Specifies the primary time zone for this resource	No	A valid time zone string
Custom Processing	Specifies whether this resource requires custom processing.	No	T (for True) or F (for False)
Accessible By	Specifies who can access a resource. It must be one of everyone, nobody, a group id or finally a user id, case insensitive. Use everyone to reset the accessibility to the default for the resource.		everyone nobody Or a group or user ID

Creating Oracle Beehive Resource Accounts

You can create Oracle Beehive resource accounts using Oracle Beekeeper or the Oracle Beehive beectl command-line utility.

To create an Oracle Beehive bookable resource using the beectl command-line tool, use the beectl add_resource command. Mandatory options when creating a resource are: **--name** , **--resource_type**, and **--email_address**.

For example, to create a bookable resource with the name Conference Room 1021 and e-mail address of room1021@domain.com, capacity of 10, and a bookable resource type of ROOM, run the following command:

```
beectl> add_resource --name "Conference Room 1021" --email_address
room1021@domain.com --resource_type ROOM --capacity 10

Resource is successfully created.
```

You can create an external resource using the beectl add_resource command with the --external option. For example:

```
beectl> add_resource --name "Conference Room 1021" --email_address
room1021@domain.com --resource_type ROOM --capacity 10 --external

Resource is successfully created.
```

If you do not specify the --external option, a bookable resource is created instead.

Listing Oracle Beehive Resource Accounts

You can list Oracle Beehive resource accounts using Oracle Beekeeper, or the beectl command-line utility.

To list Oracle Beehive bookable resources using the beectl command-line tool, use the beectl list_resources command. There are no mandatory options for the list_resources command.

For example, to list all of the bookable resources on Oracle Beehive:

```
beectl> list_resources
---------------------+-----------+----------------------------------------------
Name                 | Type      | Capacity
    +----------------+-----------+----------------------------------------------
    | Addresses
    +---+---------------------------+-------------+-----------------------------
        | ID                        | Parent ID   | Timezone
    +---+------------------+--------+---+---------+---+-------------------------
    | Location Description | Identifier | Description | BookingInfo
    +---+------------------+------+-----+-------------+-------------------------
        | Booking Characteristics | Approvers
        +-------------------------+---------------------------------------------

---------------------+-----------+----------------------------------------------
Conference Room 1021 | ROOM      | 10
    +----------------+-----------+----------------------------------------------
    | BUSINESS_1:mailto:room1021@example.com
    +---+---------------------------+-------------+-----------------------------
        | bkrs=Conference Room 1021 | enpr=Oracle | Etc/GMT
    +---+------------------+--------+---+---------+---+-------------------------
    |                      |            |             |
    +---+------------------+------+-----+-------------+-------------------------
        | FIRST_COME_FIRST_SERVED |
        +-------------------------+---------------------------------------------

---------------------+-----------+----------------------------------------------
Conference Room 1022 | ROOM      | 20
    +----------------+-----------+----------------------------------------------
    | BUSINESS_1:mailto:room1022@example.com
    +---+---------------------------+-------------+-----------------------------
        | bkrs=Conference Room 1022 | enpr=Oracle | Etc/GMT
    +---+------------------+--------+---+---------+---+-------------------------
    |                      |            |             |
    +---+------------------+------+-----+-------------+-------------------------
        | FIRST_COME_FIRST_SERVED |
        +-------------------------+---------------------------------------------

---------------------+-----------+----------------------------------------------

Company Vehicle 1    | EQUIPMENT | 4
    +----------------+-----------+----------------------------------------------
    | BUSINESS_1:mailto:company_vehicle_1@example.com
    +---+---------------------------+-------------+-----------------------------
        | bkrs=Company Vehicle 1    | enpr=Oracle | Etc/GMT
    +---+------------------+--------+---+---------+---+-------------------------
    |                      |            |             |
    +---+------------------+------+-----+-------------+-------------------------
        | FIRST_COME_FIRST_SERVED |
        +-------------------------+---------------------------------------------

---------------------+-----------+----------------------------------------------
Company Vehicle 2    | EQUIPMENT | 6
    +----------------+-----------+----------------------------------------------
    | BUSINESS_1:mailto:company_vehicle_2@example.com
    +---+---------------------------+-------------+-----------------------------
        | bkrs=Company Vehicle 2    | enpr=Oracle | Etc/GMT
    +---+------------------+--------+---+---------+---+-------------------------
    |                      |            |             |
    +---+------------------+------+-----+-------------+-------------------------
        | FIRST_COME_FIRST_SERVED |
        +------------------------+-------------------------------------------

4 Record(s) displayed.
4 resource(s) are found.
```

Modifying Oracle Beehive Resource Accounts

You can modify Oracle Beehive resource accounts using Oracle Beekeeper or the `beectl` command-line utility.

To modify an Oracle Beehive resource using `beectl`, use the `beectl modify_resources` command. Mandatory options when modifying a resource are – **--resource**, or **--select_by_name**; and a modifier option, such as **--name** or **--capacity**.

For example, to modify the name of the existing `Conference Room 1021` resource and its capacity, run the following command:

```
beectl> modify_resources --select_by_name "Conference Room 1021" --name
"Conference Room 1021: Executive Only" --capacity 15

Resource is successfully modified.
```

Deleting Oracle Beehive Resource Accounts

You can delete Oracle Beehive resource accounts using Oracle Beekeeper or the Oracle Beehive `beectl` command-line utility.

To delete an Oracle Beehive resource using `beectl`, use the `beectl delete_resource` command. Mandatory options when deleting a resource are – **--resource** or **--select_by_name**.

For example, to delete a resource with the name `Company Vehicle 2`:

```
beectl> delete_resource --select_by_name "Company Vehicle 2"

Resource is successfully deleted.
```

Managing Oracle Beehive Bookable Resource Approvers

To selectively accept or decline invitations to an event, a bookable resource can have an approver. When an approver is assigned to a resource, the approver will receive notifications whenever a user attempts to book the resource. The approver can open the resource’s calendar to accept or decline the resource’s invitations.

Oracle Beehive resource approvers can be assigned and removed usingOracle Beekeeper or the Oracle Beehive `beectl` command-line utility.

This section includes the following topics:

- Adding Oracle Beehive Bookable Resource Approvers Using beectl
- Removing Oracle Beehive Bookable Resource Approvers Using beectl

Adding Oracle Beehive Bookable Resource Approvers Using beectl

To assign an approver to an Oracle Beehive bookable resource using `beectl`:

1. Determine the name of the bookable resource. See "Listing Oracle Beehive Resource Accounts" for more information about listing resources.
2. Determine the user or users you want to assign as resource approver.s For information about listing users using the `beectl list_users` command, see "list_users" in Chapter 2 of the *Oracle Beehive Administrator's Reference Guide*.
3. Assign one or more approvers by running the following command with the specified options:

   ```
   beectl> modify_resources --select_by_name <resource name> --add_approver <user
   identifier>
   ```

   Where:

   - *<resource name>* represents the resource determined in Step 1.
   - *<user identifier>* represents the user identifier determined in Step 2.

   You can specify the `--add_approver` option multiple times to add multiple approvers to the resource.

Removing Oracle Beehive Bookable Resource Approvers Using beectl

To remove an approver from an Oracle Beehive bookable resource using `beectl`:

1. Determine the name of the bookable resource. See "Listing Oracle Beehive Resource Accounts" for more information about listing resources.
2. Determine the user or users you want to remove as the resource approver. For information about listing users using the `list_users` command, see "list_users" in Chapter 2 of the *Oracle Beehive Administrator's Reference Guide*.
3. Remove an approver by executing the following command with the specified options:

   ```
   beectl> modify_resources --select_by_name <resource name> --delete_approver
   <user identifier>
   ```

   Where:

   - *<resource>* represents the resource determined in Step 1.
   - *<user>* represents the user determined in Step .
4. To remove additional approvers, repeat step 3.

Oracle Beehive Bookable Resource Booking Characteristics

Each bookable resource has a booking characteristic. The booking characteristics determine the level of control that must be exercised over an existing bookable resource.

This section contains the following topics:

- Booking Characteristics Options
- Setting Booking Characteristics

Booking Characteristics Options

Depending on how you want to allow bookable resources to be booked, choose from these available options:

- Open
- First-Come-First-Serve

> **Note:** The default booking characteristic when the bookable resource is created is `First-Come-First-Serve`.

Open

When this option is used, a bookable resource can be booked by more than one user for the same time slot.

> **Note:** If the bookable resource has one or more approvers, those approvers can subsequently approve one user and deny other users the booking.

First-Come-First-Serve

When this option is used, double-booking is easily prevented. The first user to book the bookable resource in a time slot will automatically be accepted. Any subsequent requests to reserve the bookable resource for the same time slot will be refused.

Setting Booking Characteristics

You can set Oracle Beehive bookable resource booking characteristics when creating or when modifying a resource, using the `beectl add_resource` and `beectl modify_resources` commands.

When adding a resource:

```
beectl> add_resource [...] --resource <resourceid> --booking_characteristics
<[O]pen>|<[F]CFS>
```

In this example, other required parameters for the `beectl add_resource` command are omitted.

When modifying a resource:

```
beectl> modify_resources --resource <resourceid> --booking_characteristics
<[O]pen>|<[F]CFS>
```

Managing Oracle Beehive Resource Categorization

Oracle Beehive resources can be assigned categories. Categories are not unique to Oracle Beehive resources. Category definitions and hierarchies are defined in Oracle Beehive Workspaces. You can create a list of categories that users are allowed to apply to resources using Oracle Beekeeper, or `beectl`.

To modify the list of categories available for resources, use the `beectl modify_resource_classifications` command. Specify one or more categories using the `--category` option. For example:

```
beectl> modify_resource_classifications [--enterprise <enterpriseid>] --category
<categoryid>
```

Restricting Access to Resources

You can restrict the access to a resource using Oracle Beekeeper or `beectl`.

To restrict access to a resource using `beectl`, use the `--accessible_by` option with one of the following choices:

- **everyone**: Allows anyone to access the resource. You can use this option with the beectl modify_resources command to reset a restricted resource to unrestricted status.
- **nobody**: Prevents anyone from accessing the resource.
- a **user ID**: Only the specified user can access the resource.
- a **group ID**: Only members of the specified group can access the resource.

To restrict access when adding a resource:

```
beectl> add_resource [...] --accessible_by <everyone | nobody | groupid | userid>
```

When modifying a resource:

```
beectl> modify_resources [...] --accessible_by <everyone | nobody | groupid |
userid>
```

In these examples, other required parameters for the `beectl add_resource` command are omitted.

5

Managing Oracle Beehive Services

This module introduces the Oracle Beehive services, and the tasks and procedures for managing them. It contains the following sections:

- Introduction to Managing Oracle Beehive Services
- Managing Oracle Beehive Core Services
- Managing Collaborative Services
- Managing Enterprise Services
- Managing Platform Services

> **See Also:** For a list of service parameters for every Oracle Beehive component, their default and allowed values, and descriptions, see "Oracle Beehive Parameter Reference" in the *Oracle Beehive Administrator's Reference Guide*.

Introduction to Managing Oracle Beehive Services

Many of the services which underlie all Oracle Beehive deployments have associated management tasks for system and business administrators. These tasks revolve around changing configuration settings, establishing and managing business rules for how the system should operate, and performing routine maintenance procedures. In this module, tasks are broken down by service. You can look up any Oracle Beehive service, and review the associated management tasks at a high level.

To manage Oracle Beehive services, you make use of either the `beectl` command-line interface, or Oracle Beekeeper. With Oracle Beekeeper, you must have valid login credentials with an account having sufficient administration privileges.

> **See also:**
>
> - For a reference on using the `beectl` command-line interface, see Chapter 2, "Oracle Beehive Command-Line Utility," in the *Oracle Beehive Administrator's Reference Guide*.
> - For a list of service parameters for every Oracle Beehive component, their default and allowed values, and descriptions, see "Oracle Beehive Parameter Reference" in the *Oracle Beehive Administrator's Reference Guide*.

This section contains the following topic:

- About Oracle Beehive Services

About Oracle Beehive Services

Oracle Beehive provides a set of tightly integrated collaborative services built using J2EE and the Oracle Database. Many system functions are performed by services, which interact with each other and a common database to provide the various user features of the product.

In Oracle Beehive, there may be one or more server instances, each of which contains one each of all Oracle Beehive services. Therefore, a deployment containing several Application tiers will contain multiple instances of each service, one of each on each Application tier. In such a deployment, whenever you work with a service from the command line console of a given Application tier, you are working with the local instances of those services. In Oracle Beekeeper, you can review and manage services across all Application tiers, or select and configure specific service instances on a single Application tier.

When you make decisions about all instances of a given service, you are said to be "managing the service". When you make decisions about a specific service instance, you are said to be "managing the service instance". This distinction is important because some management tasks may be performed at either level. For example, you may configure the log level of any service instance, but you may also set the log level for a service (affecting all service instances automatically).

You can stop, start, and restart Oracle Beehive services at both levels (all instances of a service, or only one particular service instance) as well. However, in many cases, stopping individual services or service instances may cause the Oracle Beehive deployment to become unstable or fail in various ways. As a general rule, you should not stop or restart individual service instances or services unless advised to do so in the documentation, or by an Oracle support representative.

Instead, start, restart, or stop individual components, or entire Oracle Beehive servers, as described in Chapter 2, "Starting and Stopping Oracle Beehive".

Services and service instances are created during installation. You should always use the installer software when creating new service instances by creating new Application tiers.

Managing Oracle Beehive Core Services

Core services perform fundamental system functions, such as user management and authentication. This section describes the management tasks and commands for the following services:

- Managing the Access Control Service
- Managing the Audit Service
- Managing the Authentication Services
- Managing the Client Management Service
- Managing the Device Management Service
- Managing the Event Services
- Managing the Management Service
- Managing the Policy Service
- Managing the Presence Service
- Managing the User Directory Service

Managing the Access Control Service

The Access Control Service manages how users are permitted to access (see, use, and manipulate) entities in Oracle Beehive, such as files, workspaces, client services, and shared resources.

See Also: "Managing Oracle Beehive Access Control"

Related beectl Commands

The following `beectl` commands are related to the Access Control Service:

- `add_assigned_role`: Creates an AssignedRole entity
- `add_local_ace`: Adds an Access Control Entry (ACE) to an entity's Local Access Control List (LACL)
- `add_role_definition`: Creates a RoleDefinition entity
- `add_sensitivity`: Creates a Sensitivity entity
- `add_sensitivity_ace`: Adds an Access Control Entry (ACE) to a Sensitivity entity's Sensitivity ACL (SACL)
- `delete_assigned_role`: Deletes an AssignedRole entity
- `delete_local_ace`: Deletes an Access Control Entry (ACE) from the Local Access Control List (LACL) of an entity
- `delete_role_definition`: Deletes a RoleDefinition entity
- `delete_sensitivity`: Deletes a Sensitivity entity
- `delete_sensitivity_ace`: Deletes an Access Control Entry (ACE) from the Sensitivity Access Control List (SACL) of a Sensitivity entity
- `list_access_control_fields`: Lists the AccessControlFields of an entity
- `list_access_types`: Lists available access type names and identifiers
- `list_assigned_roles`: Lists AssignedRole entities
- `list_local_acl`: Lists the Local Access Control List (LACL) of an entity
- `list_privileges`: Lists available Privilege names
- `list_role_definitions`: Lists RoleDefinition entities
- `list_sensitivities`: Lists Sensitivity entities
- `list_sensitivity_acl`: Lists the Sensitivity Access Control List (SACL) of a Sensitivity
- `modify_access_control_fields`: Modifies the AccessControlFields of an existing entity
- `modify_assigned_role`: Modifies an existing AssignedRole entity
- `modify_local_ace`: Replaces an Access Control Entry (ACE) in the Local Access Control List (LACL) of an entity
- `modify_role_definition`: Modifies an existing RoleDefinition entity
- `modify_sensitivity`: Modifies an existing Sensitivity entity
- `modify_sensitivity_ace`: Replaces an Access Control Entry (ACE) in the Sensitivity Access Control List (SACL) of a Sensitivity entity

Managing the Audit Service

The Audit Service is the service interface to the Oracle Beehive Audit Framework, which supports and manages all aspects of auditing for system events.

When Oracle Beehive is installed, auditing functions are disabled by default. You can enable auditing by modifying the auditing policy. For instructions on how to enable auditing using the auditing policy, see Chapter 11, "Managing Oracle Beehive Events and Policies."

Once auditing is enabled, you can use the various `beectl` commands to create audit trails.

> **See Also:** Chapter 14, "Managing Oracle Beehive Auditing".

Related beectl Commands

The following `beectl` commands are available for you to use for managing the Audit service.

- `list_audit_policies`: Lists all audit policies, returning each policy's name and identifier
- `add_audit_policy`: Creates a new audit policy by importing from an XML file
- `modify_audit_policy`: Modifies an existing audit policy by importing changes from an XML file
- `delete_audit_policy`: Deletes a specified audit policy

> **See Also:** For more information about managing policies, including audit policies, see Chapter 11, "Managing Oracle Beehive Events and Policies."

- `list_audit_trails`: Lists all audit trails, returning each audit trail's name and CollabID
- `add_audit_trail`: Creates a new audit trail by importing from an XML file
- `modify_audit_trail`: Modifies an existing audit trail by importing changes from an XML file
- `delete_audit_trail`: Deletes a specified audit trail
- `export_audit_trail`: Exports an audit trail definition into an XML file

Managing the Authentication Services

The Authentication Services manage all aspects of user authentication for Oracle Beehive, including single sign-on (SSO), user repository authentication, authentication policies, and encryption. The Authentication Services leverage the components and protocols that support Java Authentication and Authorization Service (JAAS) and Simple Authentication and Security Layer (SASL). Client-specific authentication libraries can be supported as well.

> **See Also:** With Oracle Beehive Version 2 (2.0.1.1) and later, you can use third-party single sign-on providers for user authentication. For more information, see "Integrating Third-Party Single Sign-On Providers with Oracle Beehive," in the *Oracle Beehive Integration Guide*.

There are two services responsible for authentication functionality:

- Authentication Service
- Identity Provider Service

Managing the Authentication Service

The Authentication Service manages and supports a variety of authentication providers, including local authentication providers, existing LDAP servers, native Windows authentication providers, and Web-based SSO providers.

Related beectl Commands There are no `beectl` commands related to the Authentication Service.

Managing the Identity Provider Service

The Identity Provider Service provides certificate authority features for Oracle Beehive, enabling the system to manage digital certificates and other related security credentials.

> **See Also:** For information on how to set up Oracle Beehive with a digital certificate to enable secure communications, see "Configuring TLS with Oracle Wallet" in the Oracle Beehive Installation Guide for your platform.

Related beectl Commands There are no `beectl` commands related to the Identity Provider Service.

Managing the Client Management Service

The Client Management Service enables administrators to manage client software settings related to client connections, notification thresholds, and debugging.

Related beectl Commands

There are no `beectl` commands related to the Client Management Service.

Managing the Device Management Service

The Device Management Service is responsible for device and application program management. The Device Management Services consists of two areas of functionality:

- **Device Management**: The service manages user devices, including creating, deleting, updating, and retrieving devices. It allows administrators to manage device types and device profiles.
- **Application Management**: The service hosts application programs in the Oracle Beehive repository and makes it possible for users to install and configure applications on various devices (including PCs and mobile devices) with minimum user interaction.

Related beectl Commands

The following `beectl` commands are available for you to use for managing this service:

- `list_client_applications`: Lists all the client applications
- `upload_client_application`: Uploads the device management client binaries into the repository

- `delete_client_application`: Deletes a client application
- `list_client_application_configuration`: Lists all the client application configurations
- `add_client_application_configuration`: Creates a client application configuration object from a supplied input file. The resulting object will be used for client application provisioning
- `delete_client_application_configuration`: Deletes the client application configuration with the given identifier
- `export_client_application_configuration`: Exports a client application configuration to a local file
- `add_client_application_provisioning`: Provisions client applications to a community
- `list_client_application_versions`: Lists all the versions for a given client application
- `delete_client_application_version`: Deletes a client application version
- `list_client_application_patchsets`: Lists the patch sets for a given client application version
- `delete_client_application_patchset`: Deletes a client application patch set
- `list_client_application_modules`: Lists the modules for a given client application patch set
- `list_devices`: Lists the devices for a user
- `list_device_types`: Lists all the device types
- `list_device_profiles`: Lists all the device profiles
- `upload_device_profiles`: Uploads device profiles to the repository. An XML file is used as the source for the device profiles. The file can also contain device types and device profile schema
- `delete_device_profile`: Deletes the device profile with the given identifier. The command can also optionally delete all the device types associated with this device profile
- `list_device_commands`: Lists the device commands for a given device. The list can be further filtered by specifying the status of the device command
- `add_device_command`: Creates a device command for a given device
- `add_blocked_device`: Creates a blocked device. The system object identifier of the resulting object needs to be added to the 'DeviceManagementService' system object, using modify_property command, to block this device from accessing Beehive Mobile Services
- `delete_device_type`: Deletes the device type with the given identifier
- `download_syncml_messages`: Downloads SyncML messages

> **See Also:** For detailed information about managing mobile devices and mobile device software, see Chapter 7, "Managing Oracle Beehive Mobility Services."

Managing the Event Services

There are two services responsible for providing events functionality:

- Event Service
- Object Event Publisher Service

Managing the Event Service

The Event Service exposes business events for use by other services, including policies, notifications, logging, and auditing functions.

Related beectl Commands The following `beectl` commands are available for you to use for managing this service:

- `list_events`: Lists all business object events
- `add_event_subscription`: Fully Qualified Location of the XML file that holds data to create Event Subscription
- `delete_event_subscription`: Removes event subscription from Beehive Repository

Managing the Object Event Publisher Service

The Object Event Publisher Service handles the notification logic for object-level events in Oracle Beehive

Related beectl Commands There are no `beectl` commands related to this service.

Managing the Management Service

The Management Service supports various aspects of system administration for Oracle Beehive.

Related beectl Commands

There are no `beectl` commands related to managing this service.

Managing the Policy Service

The Policy Service enables organizations to centrally apply, manage, and store business logic for Oracle Beehive events.

> **See Also:** For information and instructions on managing policies, see Chapter 11, "Managing Oracle Beehive Events and Policies."

Related beectl Commands

The following `beectl` commands are available for you to use for managing this service:

- `list_events`: Provide the name of an event to view its description
- `list_policies`: lists the names and CollabIDs of all policies deployed in a given container. If a policy name is specified, the details of that policy are listed.
- `add_policy`: Create a policy by importing from an XML file
- `export_policy`: Exports a policy definition into an XML file

- `modify_policy`: Updates an existing policy by importing changes from an XML file
- `delete_policy`: Deletes a specified policy
- `list_policy_actions`: Lists all policy actions
- `list_policy_schemas`: Lists existing policies in a container. Details returned include policy names and identifiers
- `add_policy_schema`: Creates a new policy schema by importing from an XML file
- `modify_policy_schema`: Updates an existing policy schema by importing changes from an XML file
- `delete_policy_schema`: Deletes a specified policy schema
- `list_policy_templates`: Lists the names and CollabIDs of all policy templates (within a specified scope)
- `add_policy_template`: Creates a new policy template by importing from an XML file
- `modify_policy_template`: Updates an existing policy template by importing changes from an XML file
- `delete_policy_template`: Deletes a specified policy template
- `list_audit_policies`: lists the names and CollabIDs of all audit policies (within a specified scope)
- `add_audit_policy`: Creates a new audit policy by importing from an XML file
- `modify_audit_policy`: Updates an existing audit policy by importing changes from an XML file
- `delete_audit_policy`: Deletes a specified audit policy

Managing the Presence Service

The Presence Service supports and manages all aspects of user and resource presence for Oracle Beehive.

Related beectl Commands

There are no `beectl` commands related to managing this service.

Managing the User Directory Service

The User Directory Service (UDS) stores and retrieves information about all Oracle Beehive users. You can manage users, groups, and address books using UDS.

See Also:

- For more information about managing users and groups, see "Managing and Provisioning Oracle Beehive Users".
- For more information about managing coexistence users, see Chapter 3, "Integrating Microsoft Exchange Server 2003 or 2007 with Oracle Beehive" in the *Oracle Beehive Integration Guide*.
- For more information about setting up and managing an external LDAP-based user directory, see Chapter 4, "Integrating an External User Directory with Oracle Beehive" in the *Oracle Beehive Integration Guide*.

Related beectl Commands

The following `beectl` commands are available for you to use for managing this service:

- `add_directory_profile`: Adds directory profile to configuration repository.
- `add_group`: Adds a group containing users
- `add_preference_profile`: Adds an active preference profile for a consumer (user and groups only), if it already has a active preference profile command fails.
- `add_preference_property`: Adds a preference property to a preference set, business_hours (multiweek), datetime and datetime_list types are not implemented
- `add_preference_set`: Adds a preference set to a preference profile, template identifier is not implemented.
- `add_user`: Creates a user in a given enterprise and organization
- `delete_directory_profile`: Deletes directory profile from configuration repository.
- `delete_group`: Deletes an existing group
- `delete_preference_property`: Deletes a preference property of given name from preference set.
- `delete_preference_set`: Deletes a preference set.
- `delete_user`: Deletes an existing user.
- `download_ldap_user_data`: Generates user data from an external directory in XML format
- `list_custom_user_properties`: Lists all custom attributes
- `list_directory_profiles`: Prints the directory profiles to a file.
- `list_groups`: Lists groups and prints group information
- `list_max_address_count`: Lists the maximum entity address count and the supermax.
- `list_preference_profiles`: Lists preference profiles for a consumer.
- `list_preference_properties`: Lists all the effective preference properties of a preference set.
- `list_preference_sets`: Lists all the preference sets for a preference profile.
- `list_users`: Lists all users and prints user names and identifiers

- `modify_group`: Modifies group
- `modify_max_address_count`: Sets maximum entity address count in UDS service. NOTE: should be less than Supermax. Use `list_max_address_count` to get Supermax
- `modify_preference_set`: Sets the extends from attribute of the preference set, not yet implemented.
- `modify_user`: Modifies an existing user

Managing Collaborative Services

Collaborative Services provide collaboration-specific functionality that is leveraged among teams, including e-mail, time management, and instant messaging. This section describes management tasks and commands for the following services:

- Managing the Content Management Services
- Managing the Discussions Service
- Managing the E-mail Service
- Managing the Fax Message Service
- Managing the Instant Message Services
- Managing the Meetings Services
- Managing the Time Management Services
- Managing the Voice Message Service
- Managing the Zimbra Services

Managing the Content Management Services

There are two services that are primarily responsible for content management functionality:

- FTP Service
- WebDAV Service

Managing the FTP Service

The FTP Service supports and manages all content management-related features and settings that the system leverages over the File Transfer Protocol (FTP) and the Secure File Transfer Protocol (FTPS).

> **Caution:** If you change any port, including the FTP port, to a privileged port number (a port number below `1024`) on a UNIX or Linux installation, you must first perform a special procedure, and then change the port. First follow the special procedure in Chapter 4, "Oracle Beehive Property Reference," in the *Oracle Beehive Administrator's Reference Guide*, and then use the `beectl modify_port` command to change the port.

Using PORT Mode with the FTP Service

In PORT mode, FTP users can specify an IP address and port other than their own computer, and command the FTP server to establish a data channel. PORT mode is inherently insecure when the FTP server does not have any control over client-provided hosts and ports.

Because of this vulnerability, PORT mode is disabled by default.

If you want to enable PORT mode, you can do so by setting the activeModeEnabled parameter of the FTP Service. If `activeModeEnabled` is set to TRUE, the PORT command is enabled; if it is set to FALSE, active mode is disabled and PORT commands are not supported.

> **See Also:** For instructions on setting parameters, and descriptions of all Oracle Beehive component parameters, see "Oracle Beehive Parameter Reference" in the *Oracle Beehive Administrator's Reference Guide*.

Related beectl Commands The following `beectl` commands are related to this service:

- `modify_port --protocol FTP`: Allows you to change the FTP port without having to use the `modify_property` command

Managing the WebDAV Service

The Web-based Distributed Authoring and Versioning (WebDAV) Service supports and manages all content management-related features and settings that the system leverages over the WebDAV standard protocol.

Related beectl Commands There are no `beectl` commands related to this service.

Managing the Discussions Service

The Discussions service exposes an API which developers may use to add discussions functionality to the Beehive end-user services.

Related beectl Commands

There are no `beectl` commands related to managing this service.

Managing the E-mail Service

The E-mail Service supports all aspects of e-mail creation, delivery, and management for Oracle Beehive, including by leveraging existing e-mail applications and servers.

> **Caution:** If you change any port, including the various e-mail ports, to a privileged port number (a port number below `1024`) on a UNIX or Linux installation, you must first perform a special procedure, and then change the port. First follow the special procedure in "Oracle Beehive Parameter Reference" in the *Oracle Beehive Administrator's Reference Guide*, and then use the `beectl modify_port` command to change the port.

> **See Also:** "Managing Oracle Beehive E-mail"

Related beectl Commands

> **Caution:** If you change any port, including the various e-mail ports, to a privileged port number (a port number below 1024) on a UNIX or Linux installation, you must first perform a special procedure, and then change the port. First follow the special procedure in "Oracle Beehive Parameter Reference" in the *Oracle Beehive Administrator's Reference Guide*, and then use the `beectl modify_port` command to change the port.

The following `beectl` commands are available for you to use for managing this service:

- `modify_port --protocol SMTP --port <port_number>`

  `modify_port --protocol IMAP --port <port_number>`

  Allow you to change the SMTP and IMAP port numbers

  > **Note:** After changing SMTP or IMAP ports, you must run `activate_configuration`, just as though you modified these properties using the `modify_property` command.

- `modify_email_queue`: Enable or disable e-mail asynchronous queue processing, or process all the messages in the e-mail asynchronous queue immediately

Managing the Fax Message Service

The Fax Message Service supports and manages the delivery of fax messages to and from Oracle Beehive users.

Related beectl Commands

There are no `beectl` commands related to managing this service.

Managing the Instant Message Services

There are two services responsible for providing instant messaging functionality:

- Instant Message Service
- XMPP Service

Managing the Instant Message Service

The Instant Message Service provides core instant messaging features.

Related beectl Commands There are no `beectl` commands related to managing this service.

Managing the XMPP Service

The XMPP Service supports and manages all the features and settings that the system leverages over the Extensible Messaging and Presence Protocol (XMPP) v 0.9 and 1.0.

Caution: If you change any port, including the XMPP ports, to a privileged port number (a port number below 1024) on a UNIX or Linux installation, you must first perform a special procedure, and then change the port. First follow the special procedure in "Oracle Beehive Parameter Reference" in the *Oracle Beehive Administrator's Reference Guide*, and then use the `beectl modify_port` command to change the port.

Related beectl Commands

Caution: If you change any port, including the various e-mail ports, to a privileged port number (a port number below 1024) on a UNIX or Linux installation, you must first perform a special procedure, and then change the port. First follow the special procedure in "Oracle Beehive Parameter Reference" in the *Oracle Beehive Administrator's Reference Guide*, and then use the `beectl modify_port` command to change the port.

The following `beectl` commands are available for you to use for managing this service:

- `modify_port`: Allows you to change the XMPP port number

  ```
  modify_port --protocol XMPP --port <port_number>
  ```

Note: After changing XMPP ports, you must run `activate_configuration`, just as though you modified these properties using the `modify_property` command.

Managing the Meetings Services

The Meeting Services support and manage all aspects of voice and Web-based meetings and conferences for Oracle Beehive, enabling meeting organizers and participants to conduct collaborative sessions online through Oracle Beehive workspaces.

There are several services that are primarily responsible for meetings functionality:

- Conference Artifact Service
- Transcoding Service

Managing the Conference Artifact Service

The Conference Artifact Service provides web conference functionality.

Related beectl commands

The following `beectl` commands are related to this service:

- `add_conference`: Creates conference artifact under workspace
- `add_conference_template`: Creates conference-template artifact under workspace
- `delete_conference`: Deletes conference artifact under workspace

- `delete_conference_template`: Deletes conference template artifact under workspace
- `list_conference_templates`: Lists conference-template artifacts under workspace
- `list_conferences`: Lists conference artifacts under workspace
- `list_my_conferences`: Lists conference artifacts under workspace

Managing the Transcoding Service

The Transcoding Service supports and manages all the data-conversions and audio-conversions for Oracle Beehive voice and Web conferences.

Related beectl Commands There are no `beectl` commands related to managing this service.

Managing the Time Management Services

There are several services that are primarily responsible for calendar and time management functionality:

- Alarm Service
- CalDAV Service
- Resource Directory Service
- Time Management Service
- Time Zone Service

Managing the Alarm Service

The Alarm Service handles all time management-related alerts for the system.

Related beectl Commands There are no `beectl` commands related to managing this service.

Managing the CalDAV Service

The Calendaring Extensions to WebDAV (CalDAV) Service supports and manages all time management-related features and settings that the system leverages over the CalDAV standard protocol.

Related beectl Commands There are no `beectl` commands related to managing this service.

Managing the Resource Directory Service

The Resource Directory Service manages all aspects of the resources provided in Oracle Beehive directories, enabling users to view and schedule resources through supported time management features.

See Also: "Managing Oracle Beehive Resources".

Related beectl Commands The following `beectl` commands are related to this service:

- `add_resource`: Creates a new resource
- `delete_resources`: Deletes a resource

- `list_resources`: Lists all resources matching a given criteria (or all resources)
- `modify_resource`: Modifies an existing resource
- `modify_resource_classifications`: Sets the given category as resource root classification

Managing the Time Management Service

The Time Management Service provides the coordination services for people, teams and resources in Oracle Beehive. It supports all aspects of calendaring and scheduling, task management, resource scheduling and reminders.

Related beectl Commands The following `beectl` commands are related to managing this service:

- `import_icalendar`: Imports an iCalendar file to an existing calendar and/or task list
- `export_icalendar`: Exports invitations and assignments from a calendar and/or task list to an iCalendar file
- `list_calendars`: Lists the existing calendars of a user, resource or workspace
- `list_tasklists`: Lists existing task lists of a user, resource or workspace

Managing the Time Zone Service

The Time Zone Service supports and manages all aspects of synchronizing user schedules and calendar entries across global time zones. It acts as the central and only time zone authority for an entire Oracle Beehive deployment.

Related beectl Commands The following `beectl` commands are related to managing this service:

- `import_timezones`: Imports time zone definitions to the database
- `list_timezones`: Lists time zones in the database. The list can be limited by common time zones or by time zone names.
- `modify_timezones`: Identifies time zones as common or non-common

Managing the Voice Message Service

The Voice Message Service supports all aspects of voicemail management for Oracle Beehive.

> **See Also:** For instructions on managing the voice messaging functionality in Oracle Beehive, see Chapter 12, "Integrating Cisco Voice Gateway with Oracle Beehive Voicemail and Fax" in the *Oracle Beehive Integration Guide*.

Related beectl Commands

See Chapter 12, "Integrating Cisco Voice Gateway with Oracle Beehive Voicemail and Fax" in the *Oracle Beehive Integration Guide* for a list of `beectl` commands available for managing this service.

Managing the Zimbra Services

These services provide APIs for working with the Oracle Beehive platform:

- Zimbra Connector Service
- Zimbra UI Service

Managing the Zimbra Connector Service

The Zimbra Connector Service enables Oracle Beehive to connect to the Oracle Beehive Zimbra OC4J instance.

Related beectl Commands There are no `beectl` commands related to managing this service.

Managing the Zimbra UI Service

The Zimbra UI Service is used by Oracle Beehive to provide various user interface functionality for the Oracle Beehive Webmail.

Related beectl Commands There are no `beectl` commands related to managing this service.

Managing Enterprise Services

Enterprise services provide functionality that is leveraged across the enterprise such as search, mobile connectivity, and event subscription and notification. This section describes management tasks and commands for the following services:

- Managing the Information Rights Management (IRM) Service
- Managing the Mobility Services
- Managing the Records Management (URM) Service
- Managing the Search Service
- Managing the Subscription and Notification Services
- Managing the Workspaces Service

Managing the Information Rights Management (IRM) Service

The IRM Service is a special component only enabled if you choose to configure Oracle Beehive with Oracle Information Rights Management (IRM). Unless you configure and enable IRM, the IRM Service is disabled, and if you attempt to start it, it will shut down automatically.

> **See Also:** For detailed instructions on installing and configuring Oracle Beehive with IRM, see Chapter 5, "Integrating Oracle Information Rights Management (Oracle IRM) with Oracle Beehive" in the *Oracle Beehive Integration Guide*.

Related beectl Commands

The following `beectl` commands are related to IRM functionality:

- `add_record`: Adds a record or non-record
- `delete_record`: Deletes a record or non-record
- `list_file_plan`: Lists the file plan

Managing the Mobility Services

The Mobility Services provide Oracle Beehive users with access to their e-mail, voicemail, calendar data, and contacts through supported mobile devices, and the Oracle Beehive API. The Mobility Services also support standard protocol clients based on Open Mobile Alliance Data Synchronization (OMA-DS), and Push-IMAP (PIMAP).

There are several services responsible for providing mobility functionality:

- Managing the Mobile Device Management Service
- Managing the Mobile Data Synchronization Service
- Managing the Mobile Mail Service
- Managing the Mobile Push Service

> **See Also:** "Managing Oracle Beehive Mobility Services"

Managing the Mobile Device Management Service

The Mobile Device Management Service manages the configuration settings for the Mobile Device Management Server, which enables connections between Oracle Beehive and the Mobile Device Management Client installed on supported mobile and wireless devices.

The Mobile Device Management Service reads information from the Virtual Server component in order to configure a given device.

> **See Also:** For more information about the parameters of the Virtual Server component, see "VirtualServer" in Chapter 4, "Oracle Beehive Property Reference," of the *Oracle Beehive Administrator's Reference Guide*.

The information that Mobile Device Management Server can automatically send to devices during provisioning consists of the following:

- A 'friendly' user name, such as John Doe
- E-mail address, such as john.doe@example.com
- BTP(S) server host, such as beehive.example.com
- BPT(S) server port; either secure, such as 21401, or non-secure, such as 5224
- A boolean (0 or 1) that indicates whether or not the BTP(S) port is secure; 0 is non-secure, 1 is secure
- IMAP server port; such as IMAP (143) or IMAPS (993)
- A boolean (0 or 1) indicating whether IMAP SSL is enabled; 0 is IMAP, 1 is IMAPS
- SMTP Server port; such as SMTP (25) or SMTPS (465)
- A boolean (0 or 1) indicating whether SMTP SSL is enabled; 0 is SMTP, 1 is SMTPS
- A boolean indicating whether SMTP AUTH is required; 0 is non-required, 1 is required
- XMPP Server port; such as XMPP (5222) or XMPPS (5223)
- A boolean indicating whether XMPP SSL is enabled; 0 is XMPP, 1 is XMPPS
- A URL for mobile_ds_url; such as http(s)://server:port/mobilesync/server

- A URL for mobile_mail_url; such as http(s)://server:port/mobilemail/
- A URL for mobile_push_url; such as http(s)://server:port/mobilepush/

Related beectl Commands

There are no `beectl` commands related to managing this service.

Managing the Mobile Data Synchronization Service

The Mobile Data Synchronization Service manages all mobile-related features and settings that the system leverages through the Open Mobile Alliance (OMA) standard.

Related beectl Commands There are no `beectl` commands related to managing this service.

Managing the Mobile Mail Service

The Mobile Mail Service manages the features and settings related to push mail for supported mobile and wireless devices.

Related beectl Commands There are no `beectl` commands related to managing this service.

Managing the Mobile Push Service

The Mobile Push Service manages the features and settings that are related to the delivery of notifications to supported mobile and wireless devices.

Related beectl Commands There are no `beectl` commands related to managing this service.

Managing the Records Management (URM) Service

The Records Management Service is a special component only enabled if you choose to configure Oracle Beehive with Oracle Universal Records Management (URM). Unless you configure and enable records management, the Records Management Service is disabled, and if you attempt to start it, it will shut down automatically.

> **See Also:** For detailed instructions on installing and configuring Oracle Beehive with URM, see Chapter 7, "Integrating Oracle Universal Records Management (Oracle URM) with Oracle Beehive" in the *Oracle Beehive Integration Guide*.

Related beectl Commands

The following `beectl` commands are related to Records Management functionality:

- `add_record`: Adds a record or non-record
- `delete_record`: Deletes a record or non-record
- `list_file_plan`: Lists the file plan

Managing the Search Service

The Search Service supports and manages all aspects of user-initiated, text-based searches for Oracle Beehive.

Related beectl Commands

The following beectl commands are related to Search Service functionality:

- add_search_recovery_scope: Allows you to run a search index crawler process to index uncrawled data for the search service
- recover_search_failures: Recovers failed indexing processes

Indexing Old Data using the beectl add_search_recovery_scope Command

Beginning in Oracle Beehive 2.0.1.2.1, a new beectl command, add_search_recovery_scope, allows you to run a search index recovery crawler to process and index old (uncrawled) data.

> **Note:** While the add_search_recover_scope command can take as input an enterprise or organization scope, Oracle recommends using Workspace-level scopes, and limiting the number of items that are queued to be indexed at a single time to a medium size (~50,000 items). While monitoring the load on the server, you can then gradually add to the queue as it is processed.
>
> More information on how to monitor the load is detailed in the Search readme available at http://www.oracle.com/us/support (Doc ID 1135054.1).

The beectl add_search_recovery_scope command adds a list of items to be indexed to the SS_FEEDS table, where it will then be queued to process just like any other new or updated indexable content in your Oracle Beehive deployment.

> **See Also:** The SS_FEEDS table is discussed in the Search Service Architecture section of the Search readme available at http://www.oracle.com/us/support (Doc ID 1135054.1).

You should monitor the accumulation of data in the SS_FEEDS table as you run add_search_recovery_scope to ensure the queue is not increasing beyond the capacity of the system to process it.

> **Caution:** On many deployments, the add_search_recovery_scope command is capable of overwhelming the system's capacity to process the SS_FEEDS table in a timely manner, potentially degrading performance of the entire server. Oracle recommends using the following careful approach to adding to the queue to avoid a service outage.

Oracle recommends that you create a batch file to run this command for a few workspaces (personal or team) at a time. For very large workspaces, you can also limit the number of items based on date (age of the item).

To create the batch file, run beectl list_workspaces --type p (or t) to get a list of all personal (or team) workspaces, and then create a file which runs add_search_recovery_scope with the --scope option for each workspace. Oracle recommends starting with five workspaces and monitoring the load of the SS_FEEDS table. If your system appears to be adequately handling the load (the number of items waiting to be indexed decreases), run a batch file with another five or ten workspaces.

Managing the Subscription and Notification Services

The Subscription and Notification Services support and manage all aspects of user- and service-based subscriptions to business events and the resulting notifications. The following services are included in this category:

- Notification Delivery Service
- Subscription Service
- SMPP Delivery Service

Managing the Notification Delivery Service

The Notification Delivery Service handles all aspects of routing and channel support for notifications. It provides built-in e-mail, instant messaging, and SMS delivery channels.

> **Note:** The SMS delivery channel cannot be used until it is configured to use an SMS aggregator (such as Verisign). See "Configuring Notifications to use SMS" on page 9-5 for details.

Related beectl Commands There are no `beectl` commands related to managing this service.

Managing the Subscription Service

The Subscription Service handles all aspects of subscription logic for Oracle Beehive subscriptions.

Related beectl Commands The following `beectl` commands are related to managing this service:

- `add_user_subscription`: Creates a user subscription from a pre-defined rule in a subscription template.
- `modify_user_subscription`: Enables or disables an existing subscription

Managing the SMPP Delivery Service

The Subscription Service handles all aspects of subscription logic for Oracle Beehive subscriptions.

Related beectl Commands There are no `beectl` commands related to managing this service.

Managing the Workspaces Service

The Workspaces Service supports all the features and functionality provided by Oracle Beehive personal and team workspaces. Workspaces are the core of the user experience with Oracle Beehive, especially in regard to the collaborative activities of teams. Therefore, the Workspaces Service is responsible for consolidating and exposing, in a single location, the collaborative functionality provided by the other Oracle Beehive services.

> **See Also:** For instructions on how to manage workspaces, see "Managing Oracle Beehive Workspaces".

Related beectl Commands

The following `beectl` commands are available for you to use for managing this service:

- `list_categories`: Lists the categories in the enterprise. If the recurse option is used then sub-categories are also listed.
- `list_category`: Prints information about a category given a category identifier
- `add_category`: Creates a category at the enterprise scope
- `add_category_application`: Applies a category on a given entity
- `delete_category`: Deletes a category and all category applications
- `delete_category_application`: Removes a category from an entity
- `download_workspace_template_schema`: Downloads workspace template XML schema to a file
- `list_workspace_templates`: Lists all workspace templates
- `add_workspace_template`: Creates a workspace template in an organization or enterprise
- `modify_workspace_template`: Modifies an existing workspace template
- `delete_workspace_template`: Deletes an existing workspace template
- `list_workspaces`: Lists workspaces in an organization or enterprise
- `add_team_workspace`: Creates a team workspace from a template
- `modify_team_workspace`: Modifies an existing team workspace
- `delete_workspace`: Deletes an existing team workspace
- `modify_personal_workspace`: Modifies an existing personal workspace
- `add_sensitivity`: Creates a Sensitivity entity
- `list_sensitivities`: Lists Sensitivity entities
- `modify_sensitivity`: Modifies an existing Sensitivity entity
- `delete_sensitivity`: Deletes a Sensitivity entity

Managing Platform Services

Platform services enable organizations to integrate Oracle Beehive with existing environments and third-party components, and customize the platform to suit their needs. This section describes management tasks and commands for the following services:

- Managing the Coexistence Service
- Managing the Platform Services

Managing the Coexistence Service

The Coexistence Service enables organizations to integrate and leverage existing, third-party systems and components, such as Microsoft Exchange 2003, with Oracle Beehive for maximum interoperability.

See Also: For complete information on setting up and managing coexistence in Oracle Beehive, see Chapter 3, "Integrating Microsoft Exchange Server 2003 or 2007 with Oracle Beehive" in the *Oracle Beehive Integration Guide*.

Related beectl Commands

The following `beectl` commands are available for you to use for managing this service:

- `list_coexistence_systems`: List all configured coexistence systems
- `list_coexistence_connectors`: List all configured coexistence systems
- `add_coexistence_system`: Configure a new coexistence system
- `modify_coexistence_system`: Modify a configured coexistence system
- `delete_coexistence_system`: Delete a coexistence system

Managing the Platform Services

These services provide APIs for working with the Oracle Beehive platform:

- Platform Service
- Platform Web Service

Managing the Platform Service

The Platform API enables organizations to build and integrate custom solutions with Oracle Beehive through the Oracle Beehive API

Related beectl Commands There are no `beectl` commands related to managing this service:

Managing the Platform Web Service

The Platform Web Service is a Web-based API that enables organizations to build and integrate custom solutions with Oracle Beehive through Web Services.

Related beectl Commands There are no `beectl` commands related to managing this service.

6

Managing Oracle Beehive Workspaces

Workspaces are the central focus of Oracle Beehive. The great majority of user interactions and collaboration processes take place within the context of the workspace. Every Oracle Beehive user is presented with a personal workspace, and most users will collaborate, share information, and access project resources using team workspaces. This module describes the properties of workspaces, how to create and manage workspaces, and how to manage content stored in workspaces.

This module contains the following topics:

About Workspaces

From an architectural standpoint, workspaces are containers. They fit into a hierarchy of containers in Oracle Beehive referred to as "scope", in which a single enterprise contains organizations and workspaces, with any organization containing organizations and workspaces.

From a user's perspective, however, the workspace is at the top of a different hierarchy. A workspace may contain any number of calendars, folders (containing files or messages), address books, and other entities. Each user has a private "personal workspace", and may also have access to any number of "team workspaces".

Other than users, resources, and groups, all Oracle Beehive objects are stored within either a personal workspace, or a team workspace.

About Personal Workspaces

Every Oracle Beehive user has a single personal workspace, which acts as the container for all exposed Oracle Beehive services. The user's e-mail messages arrive in an Inbox within the personal workspace, the user's personal time management features such as calendar and task list are exposed as objects within the personal workspace, and the user can create folders and upload files to the personal workspace.

The Personal Workspace is the place where end users can see all information that is pertinent to them. E-mails and notifications are delivered to an Inbox, invitations are delivered to a personal calendar, tasks that are assigned to them or that they own are exposed in a task list. In addition, users can create folders to upload files and manage their messages, as well as manage their personal tags.

About Team Workspaces

Team workspaces are workspaces that may be created, managed, and deleted by users, and are designed for multiple users to access them and perform collaborative actions within them. Team workspaces have participants with assigned roles, and may contain shared calendars, files, folders, forums, topics, announcements, tasks, and other objects.

Team Workspaces can be specified as having Public Access, and non-participants will be granted a specific role (by default, `Read Only`, `Read, Comment and Post`, and `Read, Write, and Delete` roles are provided). If Public Access is enabled, the workspace is accessible to non-participants by using a workspace URL.

Commonly Used Commands

The following are commonly-used `beectl` commands related to managing workspaces:

- `list_categories`: Lists the categories in the enterprise. If the recurse option is used then sub-categories are also listed.

> **Note:** Categories are not currently exposed in Workspaces in any of the Oracle Beehive client user interfaces. However, you can use them with workspaces in custom client solutions you develop.
>
> See Also: *Oracle Beehive Application Developer's Guide*

- `list_category`: Prints information about a category given a category identifier
- `add_category`: Creates a category at the enterprise scope
- `add_category_application`: Applies a category on a given entity
- `delete_category`: Deletes a category and all category applications
- `delete_category_application`: Removes a category from an entity
- `download_workspace_template_schema`: Downloads workspace template XML schema to a file
- `list_workspace_templates`: Lists all workspace templates
- `add_workspace_template`: Creates a workspace template in an organization or enterprise
- `modify_workspace_template`: Modifies an existing workspace template
- `delete_workspace_template`: Deletes an existing workspace template

- `list_applied_workspace_templates`: Lists the workspace templates that have been applied to this workspace. For each applied template it indicates whether this workspace is a template evolution target.
- `list_template_evolution_target_workspaces`: Lists the workspaces to which the latest version of the given workspace template needs to be applied to make the workspaces conform to the latest version of the template
- `list_workspaces`: Lists workspaces in an organization or enterprise
- `add_team_workspace`: Creates a team workspace from a template
- `modify_team_workspace`: Modifies an existing team workspace
- `delete_workspace`: Deletes an existing team workspace
- `export_workspace`: Exports workspace content into an external directory
- `import_workspace`: Imports contents of an external directory into a new team workspace
- `modify_personal_workspace`: Modifies an existing personal workspace
- `add_sensitivity`: Creates a Sensitivity entity
- `list_sensitivities`: Lists Sensitivity entities
- `modify_sensitivity`: Modifies an existing Sensitivity entity
- `delete_sensitivity`: Deletes a Sensitivity entity

About Workspace Properties and Controls

Workspaces have a number of required and optional properties that, together, control how they are displayed to users, and what features are enabled within the workspace. They also have a variety of controls and options available for use by the workspace users.

The workspace properties and controls are:

- Name: A plain text name for the team workspace. Display names of workspaces must be unique within the current scope, and must not duplicate the names of organizations within the enterprise.
- Description: Optionally, a description of the workspace. By default it will be the same as the display name.
- Default role for new members: For team workspaces, the default workspace-scoped role is assigned to new members whenever they join or are added to the workspace. A workspace-coordinator or workspace-participant-coordinator can optionally assign a different role when adding a new user.
- Soft Quota: The soft quota defines a threshold at which a warning is given that quota is being exceeded. This value is set in MB, but may be left open (unbounded).
- Hard Quota: The hard quota defines a maximum consumption of space by quota-consuming artifacts in the team workspace. In Oracle Beehive, documents and messages are the only quota-consuming artifacts. Once the hard quota is reached, no further quota-consuming artifacts may be added. A hard-quota-exceeded error message will be given whenever an attempt is made to exceed the hard quota. This value is set in MB, and if set, must be equal to or greater than the soft quota, or it may be left open (unbounded). If the hard quota is

unbounded, the workspace may consume as much storage as has been allocated to its parent scope (its parent organization or enterprise).

- Default Sensitivity: The sensitivity that will be applied to artifacts created in the workspace by default. Sensitivities are unassigned (template) Access Control Lists.
- Members: Users and groups belonging to the workspace. Personal workspaces do not have the members attribute.
- Trash folder: A default trash folder is always created, and cannot be removed. When items are deleted from the workspace, they go in the trash folder, and can be recovered from the trash folder. Purging the trash folder permanently removes the items from the workspace.
- Inbox: A default inbox folder is always created. Messages addressed to a team workspace, or for personal workspaces, the user, will arrive in the inbox.
- Default calendar: A default calendar is always created in personal workspaces (according to the default personal workspace template). In team workspaces, the first calendar that is created becomes the default calendar (but if there are several, a user with the workspace-coordinator role can select which is the default calendar). When the workspace, or for personal workspaces, the user, is invited to calendar events such as meetings, they will be held in the default calendar.
- Default task list: A default task list is always created in personal workspaces (according to the default personal workspace template). In team workspaces, the first task list that is created becomes the default task list (but if there are several, a user with the workspace-coordinator role can select which is the default task list). Tasks assigned to the workspace, or for personal workspaces, the user, arrive in the default task list.
- Default Address Book: A default address book (contacts list) is always created in personal workspaces (according to the default personal workspace template). In team workspaces, the first address book that is created becomes the default address book (but if there are several, a user with the workspace-coordinator role can select which is the default address book). Members added to a team workspace (users and groups) are added to the default address book, and members of the workspace can add additional contacts as well.

Address Books

Team workspaces may have one or more address books to manage contacts related to projects and activities within the workspace. The address book uses the workspace membership list as one of its data sources. Address books can contain Enterprise, Extended-enterprise, and External contacts. Addressable groups for workspace-scoped groups are managed by the address book functionality of the User Directory Service. Other contacts can also be created in the workspace contacts list.

> **See Also:** For more information about Enterprise and Extended-enterprise users, and External contacts, see "About User Accounts" on page 3-3

Messaging

Team workspaces are addressable entities. Messages sent to the workspace address are stored in the workspace inbox, while messages sent to the workspace members group will be sent to each member.

Announcements

Announcements are communications to the entire team, which usually have an expiration date. A user with appropriate privilege (workspace-coordinator) can perform the following operations on a team workspace:

- Post an announcement.
- Edit or delete an existing announcement.

All members can view announcements that are posted in the workspace. There is a default folder in team workspaces where all workspace announcements appear. Announcements are a special forum in the team workspace. Each announcement has an optional activation and an expiration date.

Trash

There is always a default trash folder within a workspace. A user with appropriate privileges can delete an item by moving it to the trash folder. Any deleted item will show up in the trash folder before it is explicitly purged.

When an item is moved to the trash folder, bonds between the item and other related items still exist. Traversing bonds will not work while an artifact is in the trash, but if the item is undeleted bonds will remain intact and become traversable. For example, a link or reference to a file in a different workspace stops working if that file is moved to the trash, but will work again if the file is removed from the trash.

The trash folder is read only. Items in the trash may only be read, purged or restored. Explicit access control on an item remains on a deleted item.

Items (documents and messages) in the trash folder count against the workspace quota until they are purged.

About Workspace Events

The workspaces service raises events for the purpose of notifications, triggering policies, and auditing. When you are creating policies or event subscriptions, use the Workspace events.

Table 6–1, " Workspace Related Business Events" shows a list of the business events related to workspaces.

> **See Also:** For more information about events, see "Managing Oracle Beehive Events and Policies".

Table 6–1 Workspace Related Business Events

Event	Comments
ANNOUNCEMENT_CREATED	
ANNOUNCEMENT_DELETED	
ANNOUNCEMENT_READ	
ANNOUNCEMENT_UPDATED	
ATTACHMENT_CREATED	
ATTACHMENT_DELETED	
ATTACHMENT_UPDATED	
BOND_CREATED	

Table 6–1 (Cont.) Workspace Related Business Events

Event	Comments
BOND_DELETED	
CATEGORY_CLASS_CREATED	
CATEGORY_CLASS_DELETED	
CATEGORY_CLASS_UPDATED	
CATEGORY_CONFIGURATION_ADDED	
CATEGORY_CONFIGURATION_REMOVED	
CATEGORY_CONFIGURATION_UPDATED	
CATEGORY_INSTANCE_APPLIED	
CATEGORY_INSTANCE_REMOVED	
CATEGORY_INSTANCE_UPDATED	
DOCUMENT_ARCHIVE	
DOCUMENT_CHECK_IN	
DOCUMENT_CHECK_OUT	
DOCUMENT_COPIED	
DOCUMENT_COPIED_TO_LATEST_VERSION	
DOCUMENT_CREATED	
DOCUMENT_DELETED	
DOCUMENT_LOAD	
DOCUMENT_MOVED	
DOCUMENT_NEW_VERSION_AUTO_CREATED	
DOCUMENT_PURGE	
DOCUMENT_SECURITY_CONFIGURATION_ADDED	
DOCUMENT_SECURITY_CONFIGURATION_REMOVED	
DOCUMENT_SECURITY_CONFIGURATION_UPDATED	
DOCUMENT_UNDELETED	
DOCUMENT_UPDATED	
ENTERPRISE_CREATED	
ENTERPRISE_DELETED	
ENTERPRISE_SECURITY_CONFIGURATION_UPDATED	
ENTERPRISE_UPDATED	
ENTERPRISETRASH_PURGED	
ENTITY_LOCKED	
ENTITY_UNLOCKED	
FOLDER_ARCHIVE	
FOLDER_COPIED	
FOLDER_CREATED	
FOLDER_DELETED	

Table 6–1 (Cont.) Workspace Related Business Events

Event	Comments
FOLDER_MOVED	
FOLDER_PURGE	
FOLDER_SECURITY_CONFIGURATION_ADDED	
FOLDER_SECURITY_CONFIGURATION_REMOVED	
FOLDER_SECURITY_CONFIGURATION_UPDATED	
FOLDER_UNDELETED	
FOLDER_UPDATED	
FOLDER_VERSIONING_CONFIGURATION_ADDED	
FOLDER_VERSIONING_CONFIGURATION_REMOVED	
FOLDER_VERSIONING_CONFIGURATION_UPDATED	
LABEL_CLASS_CREATED	
LABEL_CLASS_DELETED	
LABEL_CLASS_UPDATED	
LABEL_INSTANCE_APPLIED	
LABEL_INSTANCE_REMOVED	
LABEL_INSTANCE_UPDATED	
LINK_CREATED	
LINK_DELETED	
LINK_UPDATED	
ORGANIZATION_CREATED	
ORGANIZATION_DELETED	
ORGANIZATION_SECURITY_CONFIGURATION_ADDED	
ORGANIZATION_SECURITY_CONFIGURATION_REMOVED	
ORGANIZATION_SECURITY_CONFIGURATION_UPDATED	
ORGANIZATION_UPDATED	
WIKIPAGE_CREATED	
WIKIPAGE_DELETED	
WIKIPAGE_MOVED	
WIKIPAGE_UNDELETED	
WIKIPAGE_UPDATED	
WORKSPACE_ARCHIVED	
WORKSPACE_CREATED	
WORKSPACE_HQUOTA_OVERFLOW	
WORKSPACE_PURGED	
WORKSPACE_SECURITY_CONFIGURATION_ADDED	
WORKSPACE_SECURITY_CONFIGURATION_REMOVED	
WORKSPACE_SECURITY_CONFIGURATION_UPDATED	

Table 6–1 (Cont.) Workspace Related Business Events

Event	Comments
WORKSPACE_SQUOTA_OVERFLOW	
WORKSPACE_TRASHFOLDER_EMPTIED	
WORKSPACE_UPDATED	

Managing Personal Workspaces

Personal workspaces are created automatically during user account creation, according to a personal workspace template. Oracle Beehive provides a default personal workspace template, but you can modify it or create additional personal workspace templates. For instructions on working with workspace templates, see "Using Workspace Templates" on page 6-8.

Personal workspaces are only deleted during user account deletion. Otherwise, they are undeletable.

A user may only have a single personal workspace.

If you create additional, custom personal workspace templates, the user provisioning policy determines which personal workspace template to use when creating a user account. For more information about managing policies, see Chapter 11, "Managing Oracle Beehive Events and Policies." For more information about managing and provisioning users, see Chapter 3, "Managing and Provisioning Oracle Beehive Users."

Personal workspaces can be modified using the Platform Web Services or using `beectl`. The following items may be modified using the `beectl` utility:

- Workspace name
- Workspace description
- Hard quota
- Soft quota

To modify a personal workspace, use the `beectl modify_personal_workspace` command:

```
beectl> modify_personal_workspace --workspace <Workspace identifier> --name
<Workspace name> --description <Description> --hard_quota <quota> --soft_quota
<quota>
```

Hard and soft quota values are in megabytes (MB). Use ’UNLIMITED’ to set an unlimited quota size.

Using Workspace Templates

A Workspace Template is a creation blueprint with well-defined points of variability. A template can be used for capturing best practices and for domain-specific customizations. For example, a `Project Workspace Template` could specify the blueprint for creating workspaces that are suitable for collaboration among members of teams responsible for managing a project.

An Oracle Beehive template specifies the values of attributes that are common across template instantiations, as well as the attributes whose values can differ from one instantiation to another (called 'points of variability'). It also specifies the dependencies between attribute values.

This section contains the following topics:

- About Workspace Templates
- Modifying Workspace Templates
- Creating a New Workspace Template
- Deleting a Workspace Template

About Workspace Templates

All workspaces are always created using a template. Personal workspaces are always created using a personal workspace template, and team workspaces are always created using a team workspace template.

Templates are stored in an XML format. To review the workspace template XML format, see Module 1, "Group, Policy, and Workspace Templates" in *Oracle Beehive Administrator's Reference Guide*.

Oracle Beehive comes with four pre-defined workspace templates. You can list them by using the `beectl list_workspace_templates` command:

```
beectl list_workspace_templates --scope <your enterprise identifier>
```

This produces output similar to the following:

```
----------------------------------------+---------------------------------------
Name                                    | Identifier
----------------------------------------+---------------------------------------
Basic Personal Workspace Template       | wstp=Basic Personal Workspace Template
                                        | ,enpr=yourcompany
----------------------------------------+---------------------------------------
Basic Team Workspace Template           | wstp=Basic Team Workspace Template,enp
                                        | r=yourcompany
----------------------------------------+---------------------------------------
Community of Practice Workspace Templat | wstp=Community of Practice Workspace T
e                                       | emplate,enpr=yourcompany
----------------------------------------+---------------------------------------
Project Workspace Template              | wstp=Project Workspace Template,enpr=y
                                        | ourcompany
----------------------------------------+---------------------------------------
4 Record(s) displayed.
```

The four workspace templates are:

- **Basic Personal Workspace Template**

  The personal workspace template is designed for personal workspaces, which are used solely by individual users to view and manage all of their content and collaborative activities in one primary location, including those that fall outside the scope of their team workspaces.

  By default, workspaces that are based on the Personal workspace are not listed in the system's public workspace directory. Also, although a user may not join another user's personal workspace, users can grant view-only access to each other's personal workspaces.

- **Basic Team Workspace Template**

  The Basic Team Workspace Template contains a wiki home page that members can edit to add information about the workspace, and some tasks to guide workspace coordinators as they configure the workspace. It also contains a calendar, discussion forum, and folder for sharing documents.

- **Community of Practice Workspace Template**

  The Community of Practice Template contains a FAQ template and wiki home page that members can edit to add information about this community, some tasks to guide workspace coordinators as they configure the workspace, and a wiki page, discussion forum, and folder to share best practices with other community members.

- **Project Workspace Template**

  The Project Workspace Template contains a set of tasks that can be used to get started with a project, and some wiki pages that can be used to create a business case, project plan, project completion analysis, and meeting agenda. It also contains a calendar, discussion forum, and folder for sharing documents.

Workspace Template Contents

A workspace template contains specification for workspace attributes, workspace members and entities contained in the workspace. It contains the following main items:

- **Template Attributes**

  In addition to the attributes that apply to all templates, a workspace template may also have the Domain attribute. The target domain of a workspace template is the line of business (such as life sciences, CRM, and so on) in which the template is intended to be used.

- **Workspace Attributes**

  A workspace template can include the specification of values for one or more workspace attributes (such as name, description, and so on).

- **Membership Information**

  A workspace template can specify members for the new workspace. For example, a workspace template can specify that the group PROJECT_MANAGERS should be a member of all workspaces created from the template.

- **Roles**

  A team workspace template can specify roles that can be granted to workspace members in the scope of the workspace. These roles specify privileges and access types that are granted (or denied) to an actor in the scope of the workspace.

- **Labels**

  A personal workspace template can specify one or more labels to be created for the owner of the personal workspace. For example, the default personal workspace template shipped with Oracle Beehive specifies the following two labels: Personal and Business.

- **Folder templates**

  A workspace template can include templates for the following types of folders:

  - Heterogeneous real folder (a folder for documents and messages)
  - Specialized real folder (such as a Calendar, Task List or Address Book)

  A folder template, in turn, can include templates for sub-folders. A folder template can also include templates for entities to be created in the folder. For example, a folder template can include templates for labels, policies, documents, meetings, tasks, or messages.

- **Document templates**

  A document template may optionally specify the body of the document. The body of a document is specified by reference:. a complete path name of an existing document is specified, and the content of this document is copied into the workspace at the time of template instantiation

- **Meeting (occurrence) templates**

  A meeting template specifies values for one or more attributes of an occurrence. Values of temporal attributes (such as start time) can be specified either using template variables or as offsets from workspace creation time. Meeting attendees can also be specified in the template. A meeting attendee could be any user or group in the system. In addition to ordinary meetings, templates for repeating meetings (occurrence series) can also be included in a workspace template.

- **Task templates**

  A task template specifies values for one or more attributes of a task. Values of temporal attributes (such as start time) can be specified either using user-defined template variables or as offsets from workspace creation time. Task assignees can also be specified in the template. A task assignee could be any user or group in the system.

- **Discussion forum templates**

  A discussion forum template specifies values for one or more attributes of a discussion forum. It can also include specifications for sub-forums, discussion topics and announcements.

- **Address Book templates**

  An address book template can include templates for one or more contacts

Using Expressions in Workspace Templates

Both Meeting and Task workspace templates allow you to specify multiple meetings or tasks. Oracle Beehive 1.3 and later includes a feature (the `temporalExpression` element) that allows you to use an expression to specify the time for an attribute (such as start time) for these meetings and tasks. Meetings or tasks can be specified relative to the set value, using a numerical expression.

For example, a consulting workgroup might routinely use a standardized set of tasks on each consulting engagement. A workspace administrator uses a custom consulting template to create a team workspace for the project. Within the template, an initial task is specified to kick-off the consulting project, and then additional, specific tasks follow on at various time intervals; a planning task that should be completed two days after the initial task, a milestone task that should be completed one week after the initial task, and so on.

You can set the start time variable for the first task. Then, using expressions, you can specify that the second task have an offset of 48 hours (two days), and the third task have an offset of 168 hours (seven days), and so on. Expressions can use the PLUS, MINUS, or PRODUCT arithmetic operators, and may use any template variable. You can establish a specific time value in a variable, and then specify offsets using the expressions. In this manner, expressions allow you to pre-set a complex arrangement of tasks and meetings in the workspace template, rather than having to re-create them by hand each time you create a new workspace.

For more details about using variables in workspace templates, see "Template Variables" in *Oracle Beehive Administrator's Reference Guide*.

For more details about using the expressions in workspace templates, see "Expressions" in *Oracle Beehive Administrator's Reference Guide*.

Modifying Workspace Templates

To modify a workspace template, first, download the workspace template to an XML file using the `beectl list_workspace_templates` command with the --file option:

```
beectl> list_workspace_templates --scope <Identifier of enterprise or
organization>
--name <Workspace template name> --file <Full path of the output file>
```

Note: For the `--name` option, you do not need to provide the workspace template's ID: just the name. Enclose names with spaces in double quotation marks.

The workspace template you specify will be downloaded to the file location and name you specify with the `--file` option.

Then, edit the file, and use the `beectl modify_workspace_template` command to upload your changes:

```
beectl> modify_workspace_template --template <Workspace template identifier>
--file <Full path of the input file> --name <Workspace template name>
```

Creating a New Workspace Template

You can create a new workspace template, by writing an XML-formatted file defining the template. For complete documentation on workspace template formatting, see Module 1, "Policy and Workspace Templates Reference" in *Oracle Beehive Administrator's Reference Guide*.

Then, use the `beectl add_workspace_template` command to upload the file, creating the new workspace template:

```
beectl> add_workspace_template --scope <Identifier of enterprise or organization>
--file <Full path of the input file> --name <Workspace template name>
```

If you create a workspace template at a scope other than Enterprise scope, it will only be available for creating workspaces at that scope. Using this technique, you could create different default workspace templates for members of different organizations.

Deleting a Workspace Template

You can delete a workspace template using the `beectl delete_workspace_template` command:

```
beectl> delete_workspace_template --template <Workspace template identifier>
```

Note: You should not delete the default workspace templates. You should not delete a workspace template that is used by a policy, because it could render that policy invalid in some cases.

Creating and Managing Team Workspaces

Although Oracle Beehive users can create team workspaces, you can also create and manage team workspaces from the command line.

This section contains the following topics:

- Creating Team Workspaces
- Viewing Team Workspaces
- Modifying Team Workspaces
- Deleting Team Workspaces
- Managing Categories

Creating Team Workspaces

There are four main ways to create new team workspaces:

- Using the Team Collaboration Web client
- Using developer tools exposed through the Beehive Development Kit
- Using a client which supports workspace creation, such as Oracle Beehive Extensions for Outlook (OBEO) or Oracle Beehive Extensions for Explorer (OBEE)
- Using the `beectl` command-line tool

If the Team Collaboration Web client is used, the user selects a scope to create the workspace in and which template to use. If OBEO or OBEE is used, the workspace is created in the same enterprise or organization scope as the creator's personal workspace, and the default template is used.

You can create a team workspace from the command-line. Optionally, you may create an XML-formatted file which defines one or more users as members of the workspace, and assigns those users with appropriate roles. You can then upload the XML file by designating it with the `--file` option during creation.

Create a new team workspace by using the `beectl add_team_workspace` command:

```
beectl> add_team_workspace --scope <Identifier of enterprise or organization>
--template <Workspace template identifier> --name <Workspace name> --file <Full
path of the input file>
```

A workspace always uses a template during creation. If you do not designate a template, the default workspace template for the given scope will be used.

Example 6–1, "Adding Members to a Team Workspace During Creation" shows the formatting of the XML file you may optionally upload when creating a workspace. In this example, two users are added to a workspace, and each user is given a role. In Oracle Beehive Release 1 version 1.2 or earlier, you must specify participants and roles using the `<cen>` element and full CollabIDs.

Example 6–1 Adding Members to a Team Workspace During Creation

```
<teamWorkspaceTemplate xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
    xmlns='http://xmlns.oracle.com/beehive/transportabletemplate'
    xsi:schemaLocation='http://xmlns.oracle.com/beehive/transportabletemplate
http://xmlns.oracle.com/beehive/transportabletemplate.xsd'>

    <templateAttributes>
    </templateAttributes>
    <body>
        <!-- Add users -->
        <participant>
            <identity type='USER'>
                <cen>0038:6B48:user:36C1F8C16EC34206A5021B92DDC97279000000000000000</cen>
            </identity>
            <role>
               <cen>0038:6B48:acrd:8B1514BAD5FC427E9AE42FB3A88664D200000000001A</cen>
            </role>
        </participant>
         <participant>
            <identity type='USER'>
                <cen>0038:6B48:user:36C1F8C16EC34206A5021B92DDC9727900000000000001B</cen>
            </identity>
            <role>
               <cen>0038:6B48:acrd:8B1514BAD5FC427E9AE42FB3A88664D200000000001A</cen>
            </role>
        </participant>
</body>
</teamWorkspaceTemplate>
```

In Oracle Beehive Release 1 version 1.3 or later, and Release 2, you can alternatively specify participants and roles using the shorter BODN identifiers:

```
<participant>
      <identity type="USER">
         <bodn>user=example_user1</bodn>
      </identity>
      <role>
         <bodn>acrd=test_role1,orgn=example_organization1,enpr=example.com</bodn>
      </role>
</participant>
```

Once you have created a team workspace, you can use the command-line to modify it, as described in "Modifying Team Workspaces" on page 6-15.

Viewing Team Workspaces

You can view the attributes and properties of a team workspace, by using the `beectl list_workspaces` command:

```
beectl list_workspaces --scope <Identifier of enterprise or organization> --type
<p|t|a> --name <Workspace name>
```

Provide a value for the `--name` option to show details of a specific workspace. Example 6–2, "Example Team Workspace" is an example of the output from such a command.

Example 6–2 Example Team Workspace

```
Workspace name:                        my_team_workspace
Description:                           my_team_workspace
```

```
Workspace type:                         TEAM
Identifier:                             wksp=wksp=my_team_workspace,orgn=human_
resources,enpr=mycompany
Hard quota in kilo-bytes (KB):          Unlimited quota
Soft quota in kilo-bytes (KB):          Unlimited quota
Workspace parent:                       orgn=human_resources,enpr=mycompany
Workspace path:                         /MYCOMPANY/HUMAN_RESOURCES/MY_TEAM_WORKSPACE
Default sensitivity:                    acsn=Normal,wksp=my_team_workspace,orgn=human_
resources,enpr=mycompany
Folder identifier:                      adbk=Contacts,wksp=my_team_workspace,orgn=human_
resources,enpr=mycompany
Folder identifier:                      fldr=Announcements,wksp=my_team_workspace,orgn=human_
resources,enpr=mycompany
Folder identifier:                      fldr=INBOX,wksp=my_team_workspace,orgn=human_
resources,enpr=mycompany
Folder identifier:                      fldr=Documents,wksp=my_team_workspace,orgn=human_
resources,enpr=mycompany
Folder identifier:                      fldr=Public Documents,wksp=my_team_workspace,orgn=human_
resources,enpr=mycompany
Folder identifier:                      clnd=Calendar,wksp=my_team_workspace,orgn=human_
resources,enpr=mycompany
Folder identifier:                      fldr=Workspace Trash,wksp=my_team_workspace,orgn=human_
resources,enpr=mycompany
Folder identifier:                      fldr=Tasks,wksp=my_team_workspace,orgn=human_
resources,enpr=mycompany
Workspace template identifier:          16C3:57F2:ttws:37D536448BC43F3DE040578C211A3EA80000000001D6
Is directory listed?:                   false
Default role definition:                acrd=workspace-member,enpr=mycompany
Participation mode:                     INVITE_ONLY
Workspace participant:                  user=jsmith
    Assigned role:                      acar=workspace-coordinator,wksp=my_team_
workspace,orgn=human_resources,enpr=mycompany
```

In this example, a workspace named "my_team_workspace" has been created using the default team workspace template. The workspace was created within the "human_ resources" organization of the enterprise called "mycompany". No quota or hard quota has been set. A user has been added, with the login ID of "jsmith", and granted the role of workspace-coordinator.

You can also use Oracle Beekeeper to view a list of team workspaces in the system and launch to the Team Collaboration client to take actions on behalf in the workspace. You must have the 'read all' privilege in order to properly load the workspace.

See Also: *Oracle Beekeeper Online Help*

Modifying Team Workspaces

Once a team workspace is created, you can modify it from the command-line to add or remove users, change its e-mail address, change its participation mode, and to modify the quota. You can make many other modifications to a workspace using the Team Collaboration Web client, OBEO, OBEE, and the Platform Web Services.

For information about adding and removing members, see "Managing Team Workspace Membership" on page 6-19.

To change the e-mail address of a team workspace from the command line, use the `beectl modify_team_workspace` command with the `--email_address` option:

```
beectl> modify_team_workspace --workspace <Workspace identifier> --email_address
<Team workspace email address>
```

To change the participation mode of a team workspace from the command line, use the beectl modify_team_workspace command with the --participation_mode option:

```
beectl> modify_team_workspace --workspace <Workspace identifier> --participation_
mode <Team workspace participation mode>
```

You can use any of the following values: INVITE_ONLY, OPEN, or APPROVE_REQUIRED

To modify the quota of a team workspace from the command line, use the beectl modify_team_workspace command with the --soft_quota or --hard_quota options:

```
beectl> modify_team_workspace --workspace <Workspace identifier> --hard_quota <new
quota in MB> --soft_quota <new quota in MB>
```

To modify whether a team workspace is directory-listed from the command line, use the beectl modify_team_workspace command with the --directory_listed option:

```
beectl> modify_team_workspace --workspace <Workspace identifier> --directory_
listed <TRUE|FALSE>
```

Deleting Team Workspaces

You can delete a team workspace by using the beectl delete_workspace command:

```
beectl> delete_workspace --workspace <Workspace identifier> [--purge]
```

When you delete a team workspace, all artifacts stored in that workspace are also deleted. Use the --purge option to delete artifacts manually.

Beginning with Oracle Beehive 2.0, an asynchronous delete flow for team workspaces has been introduced. After a user chooses to delete a workspace, it is taken 'offline' and will asynchronously be purged in the background. Depending on your business and system resource requirements, you may need to manage deleted, offline (not yet purged) workspaces. The following beectl commands are now available to assist you with listing deleted workspaces and manually purging deleted workspaces:

- list_deleted_workspaces: This command returns a list of deleted workspaces that are not yet purged in the system
- delete_workspace --purge: This command manually purges a deleted workspace synchronously. This command can also purge a visible workspace.

Managing Categories

Note: Categories are not exposed in the Oracle Beehive Team Collaboration Web client, OBEE, or OBEO. However, you can use categories with custom clients you build using the Oracle Beehive Development Kit (BDK). This section describes category capabilities you can use in your custom clients.

Categories are a hierarchical structure of designations that may be applied to entities, including all of the artifacts stored in a workspace. Categories always exist at the enterprise scope.

You can determine default categories available in a workspace during workspace creation: either from the workspace template, or, directly by specifying them in the XML file provided when you create the workspace.

In addition, you can create and delete categories, and you can apply and remove them from objects in workspaces. You create a category by uploading an XML formatted category definition file.

To create a new category, use the `beectl add_category` command:

```
beectl> add_category --file <path to the category XML file>
```

Example 6–3 shows an XML file for adding a simple category to an enterprise.

Example 6–3 Example Category XML File

```
<?xml version = '1.0' encoding = 'UTF-8'?>
<!-- Sample Template to add a Category -->
<CategoryDefinition xmlns="http://xmlns.oracle.com/beehive/category">
   <name>TTTesCat1->1179090518828</name>
</CategoryDefinition>
```

Example 6–4 shows an XML file for adding a category with attributes. An attribute has a default value, and can also have allowed alternate values.

Example 6–4 Example Category with Attributes XML File

```
<?xml version = '1.0' encoding = 'UTF-8'?>
<!-- Sample Template to Create a Category with Attributes -->
<CategoryDefinition xmlns="http://xmlns.oracle.com/beehive/category">
   <name>Test Category16</name>
   <description>Test Category-Desc</description>
   <abstract>T</abstract>
   <defaultTemplate>
      <copyOnVersion>T</copyOnVersion>
      <mandatory>F</mandatory>
      <finalInd>F</finalInd>
      <attributeTemplates>
         <attributeTemplate>
            <attributeDef>
               <name>AdefX1-1</name>
               <propertyType>STRING</propertyType>
            </attributeDef>
            <mandatory>F</mandatory>
            <prompted>T</prompted>
            <finalized>F</finalized>
            <forceDefault>F</forceDefault>
         </attributeTemplate>
         <attributeTemplate>
            <attributeDef>
               <name>AdefX2-1</name>
               <propertyType>STRING</propertyType>
            </attributeDef>
            <mandatory>T</mandatory>
            <finalized>F</finalized>
            <forceDefault>T</forceDefault>
            <allowedValues>
```

```
              <allowedVal>
                 <name>AL2</name>
                 <description>Desc-AL2</description>
                 <value>TestVal2</value>
              </allowedVal>
            </allowedValues>
            <defaultValue>
              <value>
                TestVal2
              </value>
            </defaultValue>
         </attributeTemplate>
      </attributeTemplates>
   </defaultTemplate>
  <attributes>
      <attribute>
         <name>AdefX1-1</name>
         <description>TestAdef1</description>
         <propertyType>STRING</propertyType>
         <searchable>T</searchable>
         <defaultValue>
           <value>
             TestVal1-Def
           </value>
         </defaultValue>
      </attribute>
      <attribute>
         <name>AdefX2-1</name>
         <description>TestAdef2</description>
         <propertyType>STRING</propertyType>
         <searchable>F</searchable>
         <defaultValue>
           <value>
             TestVal2
           </value>
         </defaultValue>
         <allowedValues>
              <allowedVal>
                 <name>AL1</name>
                 <description>Desc-AL1</description>
                 <value>11</value>
              </allowedVal>
              <allowedVal>
                 <name>AL2</name>
                 <description>Desc-AL2</description>
                 <value>TestVal2</value>
              </allowedVal>
            </allowedValues>
      </attribute>
   </attributes>
</CategoryDefinition>
```

To delete a category, use the `beectl delete_category` command:

```
beectl> delete_category --category <Identifier of the category to be deleted>
```

Note: When you delete a category, all applications of that category are automatically removed.

To apply a category to an entity in a workspace, use the `beectl add_category_application` command:

```
beectl> add_category_application --category <Identifier of the category to be
applied> -- entity <Identifier of the entity to which the category needs to be
applied>
```

To remove a category from an entity in a workspace, use the `beectl delete_category_application` command:

```
beectl> delete_category_application --category <Identifier of the category to be
removed> --entity <Identifier of the entity from which the category needs to be
removed>
```

Managing Team Workspace Membership

You can add members to a team workspace during creation by formatting an XML file for upload. In the file, you specify any number of users (and groups) to be members of the team workspace. You can also specify roles for the users. At least one user of any team workspace should have the workspace-coordinator role.

To view a list of all of the available roles, use the `beectl list_role_definitions` command:

```
beectl> list_role_definitions
```

For a list of team workspace-related roles, see Table 6–2, " Summary of Default Team Workspace Roles" on page 6-21.

Example 6–5, "Sample Team Workspace Adding Members XML File" is an example file, showing two members to be added to a workspace; each member is granted a role, by pasting in the CollabID of a role in the <cen> element.

Example 6–5 Sample Team Workspace Adding Members XML File

```
<teamWorkspaceTemplate xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
    xmlns='http://xmlns.oracle.com/beehive/transportabletemplate'
    xsi:schemaLocation='http://xmlns.oracle.com/beehive/transportabletemplate
http://xmlns.oracle.com/beehive/transportabletemplate.xsd'>

    <templateAttributes>
    </templateAttributes>
    <body>
        <participant>
            <identity type='USER'>

<cen>0038:6B48:user:36C1F8C16EC34206A5021B92DDC97279000000000000</cen>
            </identity>
            <role>

<cen>0038:6B48:acrd:8B1514BAD5FC427E9AE42FB3A88664D200000000001A</cen>
            </role>
        </participant>
         <participant>
            <identity type='USER'>

<cen>0038:6B48:user:36C1F8C16EC34206A5021B92DDC9727900000000001B</cen>
            </identity>
            <role>
```

```
<cen>0038:6B48:acrd:8B1514BAD5FC427E9AE42FB3A88664D200000000001A</cen>
            </role>
        </participant>
</body>
</teamWorkspaceTemplate>
```

You can add, modify, and remove members from an existing workspace, by using the `beectl modify_team_workspace` command. You will need the unique workspace identifier as well as the unique IDs of any users you will add, modify or remove from the workspace. In this context, modifying the user only means modifying a user’s role; you do not actually modify user accounts when managing team workspaces.

To add a user (or a group):

```
beectl> modify_team_workspace --workspace <workspace identifier> --add_participant
<user or group identifier> --role <workspace role>
```

For example:

```
beectl> modify_team_workspace --workspace wksp=our_project,orgn=human_
resources,enpr=mycompany --add_participant user=jsmith --role
acrd=workspace-coordinator,enpr=mycompany
```

In this example, a user with the ID of "jsmith" is added to a team workspace called "our_project", which is in the organization called "human_resources", and granted the role of workspace-coordinator.

To remove a user (or a group):

```
beectl> modify_team_workspace --workspace <workspace identifier> --remove_
participant <user ID>
```

Managing Team Workspace Access Control

In addition to explicit access control (using Access Control Entities to explicitly allow or disallow levels of access on objects), there are two methods for general control of access to entities in workspaces: roles, and sensitivities.

You can also manage the visibility of workspaces (and the content within them) to users in the enterprise who are not already members of the workspace.

This section contains the following topics:

- Enabling Public Access to Team Workspaces
- Managing Team Workspace Roles
- Creating Team Workspace Roles
- Managing Team Workspace Sensitivities

> **See Also:** For complete instructions on managing access control, see Chapter 13, "Managing Oracle Beehive Access Control."

Enabling Public Access to Team Workspaces

You can specify a team workspaces as having 'Public Access'. Non-participants will then be granted a specific role (by default, the 'Read Only', 'Read, Comment and Post', and 'Read, Write, Delete' roles are available). If Public Access is enabled, the workspace will be accessible to non-participants through a workspace URL.

Non-participants will not see the workspace listed anywhere; they will require the URL to access the workspace.

Managing Team Workspace Roles

Within team workspaces, roles are used to define levels of control which workspace members may exercise over the workspace and its content.

Users with sufficient administrative privileges can perform the following administrative operations on a team workspace:

- Make a user or a group member of the workspace. When a group is added as a member of a workspace, the group membership is honored dynamically. This means any new member of the group automatically becomes a member of the workspace (via the group).
- Remove an existing member. When removing a member, option exists whether to revoke all the permissions that have been granted to the user for this workspace
- Change the roles/permissions of a member
- Invite the contacts, including enterprise users and extended-enterprise users, to become members
- Remove himself or herself from the workspace

Users can perform the following read operations on a team workspace:

- View the members of a workspace
- Retrieve the workspace membership information of a specific user or group

Table 6–2, " Summary of Default Team Workspace Roles" shows the roles and granted privileges related to team workspaces.

Table 6–2 Summary of Default Team Workspace Roles

Role	Granted Privileges	Granted Access Types
workspace-coordinator	[ADDRESS_BOOK_MGR, CALENDAR_MGR, CONF_MGR, CONTENT_MGR, EMAIL_MGR, FORUM_MGR, IM_MGR, MARKER_MGR, MODIFY_ACL, NOTIFICATION_MGR, POLICY_MGR, PREFERENCE_MGR, READALL, ROLE_MGR, SECURITY, SUBSCRIPTION_MGR, USER_MGR, VERSION_MGR, WORKSPACE_MGR	discover, read, write, execute, delete
workspace-participant-coordinator	MODIFY_ACL, ROLE_MGR, USER_MGR	read, discover
workspace-document-coordinator	CONTENT_MGR, FORUM_MGR, MARKER_MGR, MODIFY_ACL, VERSION_MGR	discover, read, write, execute, delete
workspace-viewer	none	discover, read
workspace-member	none	discover, read, write, execute, delete
workspace-commenter	FORUM_WRITER	discover, read

The workspace-participant-coordinator role grants the ROLE_MGR privilege, which allows the creation and management of custom roles, within the scope of the workspace. In addition, ROLE_MGR allows the user to grant and revoke workspace-scoped custom roles to and from other users in the workspace.

In addition to the workspace roles, there are application-level roles which grant privileges over all workspaces. These roles are summarized in Table 6–3, " Summary of Default Application-Level Roles".

Table 6–3 Summary of Default Application-Level Roles

Role	Granted Privileges	Granted Access Types
enterprise-administrator	ARCHIVE_MGR, EXCEED_QUOTA, MARKER_MGR, ORGANIZATION_MGR, PREFERENCE_MGR, QUOTA_MGR, ROLE_MGR, USER_MGR, VERSION_MGR, WORKSPACE_MGR	discover, read, write, execute, delete
enterprise-system	BYPASS	discover, read, write, execute, delete

Creating Team Workspace Roles

When a participant is added to a workspace, they are assigned one or more roles. Roles are based on a Role Definition object. You can be create new role definitions at the enterprise, organization or workspace levels of scope. If you later change the name of a role definition, that change is reflected anywhere the custom role has been assigned.

Beginning with Oracle Beehive Release 2, a new role definition is provided: **workspace-commenter** (a user who has read, comment and post capabilities). In the following procedure, the creation of this role is used as a model to show how to create new roles using `beectl`.

To create a new role, perform the following steps:

1. Add a new role definition using the `beectl add_role_definition` command. For example:

   ```
   beectl> add_role_definition --scope <enterprise identifier> --name
   workspace-commenter --privilege FORUM_WRITER --access_types OR --always_enabled
   true
   ```

   In this example, the `FORUM_WRITER` privilege allows the assignee to create posts in forums in the scope in which they are assigned the role (typically, a workspace). The assignee is also granted access types `O` (discover) and `R` (read) for all entities within the scope they are granted the role.

2. Create an access control entity using the `beectl add_local_ace` command. For example:

   ```
   beectl> add_local_ace --entity <new role identifier> --accessor agrp=ALL_USERS
   --access_types R
   ```

   This step allows `ALL_USERS` access to Read the role, and therefore be able to the new role's existence.

3. Get the category identifier of the WorkspaceParticipantRoleDefinitionCategory by using the `beectl list_categories` command:

   ```
   beectl> list_categories
   ```

   Make a note of the identifier for use in the next step.

4. Apply the category to the new role using the `beectl add_category_application` command. For example:

```
beectl> add_category_application --category
<WorkspaceParticipantRoleDefinitionCategory Identifier> --entity <new role
identifier>
```

   Once this category is assigned, clients like the Team Collaboration web client, OBEO, and OBEE will display it. (The clients know to ask the system only for all roles with this category assigned to it.) This step is required to avoid displaying inappropriate roles (such as enterprise-quota-administrator) at the Workspace scope.

Managing Team Workspace Sensitivities

Sensitivities are unassigned access control lists, packaged and given a name. Users with appropriate privileges may assign sensitivities to entities under their control. This allows users to manage access control over entities without needing to learn about or understand how access control works in detail.

The default personal workspace creates two sensitivities: public and private.

You can define sensitivities during workspace creation, by specifying them in the workspace template.

For detailed information about creating and managing sensitivities, see "Creating and Managing Sensitivities" in Chapter 13, "Managing Oracle Beehive Access Control."

Managing Files

Workspace users can create heterogeneous folders (entity real folders) and sub folders within any workspace to manage artifacts, including "library content" (documents, URLs, notes, and links), topics and messages (e-mails, discussions, voice mail messages, fax messages), IM chat logs, calendar events, tasks, and contacts.

Some of these folders can contain artifacts that may be stored in external file system directories, or accessed over FTP and WebDAV protocols.

This section contains the following topics:

- Managing File System Directories
- Managing FTP and WebDAV Access to Files

Managing File System Directories

In Oracle Beehive, by default all user content is stored in the database. However, you may elect to store some content in file system directories. A file stored in a file system directory is treated as read-only by Oracle Beehive.

Whenever a user or process performs a read action on the file, the file is read from the file system directory.

At any time, if changes are made to the file in Oracle Beehive, such as if a user modifies the file content, the file is imported from the file system directory into the Oracle Beehive database. The unchanged, original file remains in the file system directory, but Oracle Beehive stores the new file in the database, and will continue to make use of only the database copy of the file.

> **Note:** This functionality allows you to expose your existing documents and files to Oracle Beehive users without having to perform a batch-import of all files to the Oracle Beehive database. Instead, map your files using the file system reference commands, and individual files will be automatically imported only as needed.

You can use the following `beectl` commands to manage file system directories and files:

- `add_filesystem_reference`: Creates a reference in Oracle Beehive to a directory on the file system
- `delete_filesystem_reference`: Removes a file system reference from Oracle Beehive
- `import_documents`: Imports documents into Oracle Beehive from files on the server without copying the file content. Data on the server files will be treated as read-only; should an imported document be edited in Oracle Beehive, a copy of the content will be made at that time.

  > **Caution:** Before importing documents to a workspace using the `import_documents` command, you should consider the effects of any existing policies on that workspace. A policy that is triggered on any new document created or added in a workspace could be triggered repeatedly as multiple files are imported.

- `list_filesystem_references`: Lists the file system path, read-only status, and identifier (CollabID) of all available file system references.

Using File System Directories with Multiple Oracle Beehive Servers

For high availability deployments with a shared file system (or that leverage the `filesystem_reference` object within workspaces), all computers on which Oracle Beehive Application Tier instances and Oracle Database instances reside should have access to the file system reference paths at the same logical location. This shared access may be accomplished using a Network File System (NFS) server, symbolic links (symlinks), or another supported method. Typically, organizations will experience optimal performance if their file systems reside on computers other than those on which Oracle Beehive and Oracle Database reside.

The following two requirements detail the necessary access for file system references to function properly:

- The BEECORE OC4J component executing the `beectl import_documents` command must have file system access to the specified server path.
- The computer hosting the Oracle Beehive database must have local file-system access to the specified server path. SQL requires a local file-system path when creating a BFILE.

Creating and Using File System References

To map existing files to Oracle Beehive, perform the following steps:

1. Use the `beectl add_filesystem_reference` command to map an existing server path for Oracle Beehive:

   ```
   beectl> add_filesystem_reference --name <Filesystem reference name>
   ```

```
--filesystem_path <Server path> --read_only <true or false>
```

Note: If you set the `--read_only` flag to `true`, Oracle Beehive will treat the file objects as read-only internally, meaning, users will not be allowed to modify them. If you set it to false, users will be allowed to modify the files, which will trigger the file importation into the Oracle Beehive database.

Under no conditions will files on the file system be modified by Oracle Beehive.

2. Use the `beectl import_documents` command to create individual references to all of the documents within Oracle Beehive:

```
beectl> import_documents --filesystem_reference_id <CollabId of the filesystem
reference> --folder_path <Folder path> [--name_filter <name filter>]
[--conflict_res_mode <ABORT|OVERWRITE|CREATE_UNIQUE>]
```

Caution: Before importing documents to a workspace using the `import_documents` command, you should consider the effects of any existing policies on that workspace. A policy that is triggered on any new document created or added in a workspace could be triggered repeatedly as multiple files are imported.

The `--folder_path` specifies the folder path within Oracle Beehive to import the files. For example, you could specify a folder within a specific workspace.

The `--name_filter` option allows you to specify only a subset of the files in the file system directory to be imported. For example, you could specify the filter `%.doc` to only import files with the `.doc` extension.

the `--conflict_res_mode` determines how Oracle Beehive should treat files to be imported from the file system directory, when a file already exists in the target Oracle Beehive directory with the same name. You may choose to skip such files with the ABORT option, overwrite them, or create a new, unique file name for the file automatically.

You can also manage existing file system directories by listing them and deleting them.

To list all file system directories currently mapped in Oracle Beehive, use the `beectl list_filesystem_references` command:

```
beectl> list_filesystem_references
```

To delete a file system reference, use the `beectl delete_filesystem_reference` command:

```
beectl> delete_filesystem_reference --filesystem_reference_id <CollabID>
```

When you delete a file system reference, any files currently linked-to that have not been imported into the Oracle Beehive database become unavailable. Files already imported into the Oracle Beehive database remain available and are treated as normal files.

Managing FTP and WebDAV Access to Files

Content stored in workspaces may be made available to users over FTP and WebDAV protocols. FTP access is controlled by the FTP Service, and WebDAV access is controlled by the WebDAV service. When these protocols are enabled, users with supported clients can authenticate with Oracle Beehive, and then access files stored in any workspace with which they have sufficient privileges. In all respects, access via FTP and WebDAV is treated the same as access from any other user client; explicit and implicit access control is respected. User actions over these protocols are restricted to uploading, moving, and downloading files, and creating, moving, and deleting folders. Users cannot apply or change sensitivities or categories on files through these protocols.

For information about how to configure and enable FTP and WebDAV, see "Managing Oracle Beehive Services".

Managing Records Management

Records Management is an optional service, which is enabled by installing and configuring Oracle Beehive with Oracle Universal Record Manager (URM). URM provides lifecycle and disposition management of records managed Oracle Beehive artifacts. Once URM is installed and configured with Oracle Beehive, you can start the Records Management Service, and begin filing records for documents and e-mails.

Records Management documentation can now be found in Chapter 7, "Integrating Oracle Universal Records Management (Oracle URM) with Oracle Beehive" in the *Oracle Beehive Integration Guide*.

Example Workspace Template Contents

Example 6–6, "Example Workspace Template XML File" shows an example workspace template XML file (in this case, the Community of Practice Workspace template):

Example 6–6 Example Workspace Template XML File

```
<teamWorkspaceTemplate
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns="http://xmlns.oracle.com/beehive/transportabletemplate"
 xsi:schemaLocation="http://xmlns.oracle.com/beehive/transportabletemplate
http://xmlns.oracle.com/beehive/transportabletemplate.xsd">
<templateAttributes>
<author>Oracle</author>
<authorCreationTime>Oct, 2009</authorCreationTime>
<contactInfo>Oracle Corporation</contactInfo>
<copyrightInfo>Copyright (c) 2009, Oracle Corporation. All rights reserved.</copyrightInfo>
<description>This community of practice template contains an FAQ template and wiki home page that
members can edit to add information about this community, some tasks to guide the coordinators as
they configure the workspace, and a wiki page, forum and folder to share best practices with other
community members.</description>
<name>Community of Practice Workspace Template</name>
<templateId>oracle.com.community-of-practice-workspace-template</templateId>
<domain>general</domain>
</templateAttributes>
<body>
   <publicSensitivityTemplateBodyId>public_sensitivity</publicSensitivityTemplateBodyId>
   <defaultSensitivityTemplateBodyId>default_sensitivity</defaultSensitivityTemplateBodyId>
  <attributes>
```

```
     <name prompt="true" promptMessage="enter workspace name">Community of Practice
Workspace</name>
     <description>This workspace was created using the Community of Practice template.  It contains
an FAQ template and wiki home page that members can edit to add information about this community,
some tasks to guide the coordinators as they configure the workspace, and a wiki page, forum and
folder to share best practices with other community members.</description>
     <publiclyListed>F</publiclyListed>
     <participationMode>INVITE_ONLY</participationMode>
  </attributes>
  <preferenceSet>
      <name>ForumPreferences</name>
      <description>forum preferences</description>
      <property>
        <name>DISC_UPDATE_WHO</name>
        <value type="STRING">AUTHOR_ONLY</value>
        <allowOverride>true</allowOverride>
      </property>
      <property>
        <name>DISC_DELETE_WHO</name>
        <value type="STRING">AUTHOR_ONLY</value>
        <allowOverride>true</allowOverride>
      </property>
      <property>
        <name>DISC_UPDATE_WHEN</name>
        <value type="STRING">NO_REPLIES</value>
        <allowOverride>true</allowOverride>
      </property>
      <property>
        <name>DISC_DELETE_WHEN</name>
        <value type="STRING">NO_REPLIES</value>
        <allowOverride>true</allowOverride>
      </property>
  </preferenceSet>
  <property>
      <name>forumDefaultDest</name>
      <description>forum default destination</description>
      <value type = "STRING">Forum List</value>
  </property>
  <property>
      <name>libDefaultDest</name>
      <description>library default destination</description>
      <value type = "STRING">Default Folder</value>
  </property>
  <property>
      <name>wikiDefaultDest</name>
      <description>wiki default destination</description>
      <value type = "STRING">All Pages</value>
  </property>
  <sensitivity id="default_sensitivity">
      <name>Normal</name>
      <description>normal sensitivity</description>
      <sensitivityOnly>false</sensitivityOnly>
      <delegatable>true</delegatable>
      <ace>
          <grantAccessType>DISCOVER</grantAccessType>
          <accessor type="GROUP"><systemDefinedGroupName>ALL_
USERS</systemDefinedGroupName></accessor>
      </ace>
  </sensitivity>
  <sensitivity id="public_sensitivity">
```

```
      <name>Public</name>
      <description>public sensitivity</description>
      <sensitivityOnly>false</sensitivityOnly>
      <delegatable>true</delegatable>
      <ace>
          <grantAccessType>DISCOVER</grantAccessType>
          <grantAccessType>READ</grantAccessType>
          <accessor type="GROUP"><systemDefinedGroupName>ALL_
USERS</systemDefinedGroupName></accessor>
      </ace>
  </sensitivity>
  <defaultAnnouncementsForum id="default_ann_forum">
     <name>Announcements</name>
     <description>Forum for workspace announcements</description>
        <announcement>
           <subject>Welcome to the ${sys.workspace.name} Workspace</subject>
           <messageBody>
              <mediaType>text/plain</mediaType>
              <body>This workspace was created using the Community of Practice template.  It
contains an FAQ template and wiki home page that members can edit to add information about this
community, some tasks to guide the coordinators as they configure the workspace, and a wiki page,
forum and folder to share best practices with other community members.</body>
           </messageBody>
           <expiresOn prompt="false" offsetUnit="DAY"><offset>7</offset></expiresOn>
        </announcement>
  </defaultAnnouncementsForum>
  <defaultAddressBook id="default_address_book">
     <name>Contacts</name>
     <description>Workspace address book</description>
  </defaultAddressBook>
  <defaultCalendar id="default_calendar">
     <name>Calendar</name>
     <description>Workspace Calendar</description>
     <enrollmentType>PUBLIC</enrollmentType>
     <selfEnrollmentOpenToAll>false</selfEnrollmentOpenToAll>
     <enrollWorkspaceMembers>false</enrollWorkspaceMembers>
  </defaultCalendar>
  <defaultDiscussionsForum id="default_disc_forum">
     <name>Discussions</name>
     <description>Default forum for team discussions</description>
  </defaultDiscussionsForum>
  <defaultDocumentsFolder id="documents_folder">
     <name>Documents</name>
     <description>A folder for documents that you want to share with participants of this
workspace.</description>
     <entities>
        <folder>
           <name>Best Practices</name>
           <description>A folder for sharing best practice documents with the participants of this
workspace.</description>
        </folder>
     </entities>
  </defaultDocumentsFolder>
  <defaultEmailInbox id="inbox_folder">
     <name>INBOX</name>
     <description>Inbox for email messages</description>
  </defaultEmailInbox>
  <defaultTaskList id="default_task_list">
     <name>Tasks</name>
     <description>Team tasks</description>
     <todo>
       <name>Configure and customize the workspace</name>
       <description>
Modify the FAQ, Best Practices and Wiki Home pages.
Add additional content.
This task has been assigned from the ${sys.workspace.name} workspace.
</description>
       <startTime offsetUnit="SECOND"><offset>1</offset></startTime>
       <dueTime offsetUnit="DAY"><offset>1</offset></dueTime>
       <organizer type="USER"><cen>${sys.workspace.owner.collabid}</cen></organizer>
      </todo>
      <todo>
       <name>Identify and add workspace participants</name>
       <description>
Add participants to the community who can help contribute knowledge using the New Participants
function.  If you are adding a user who is manager, you will have the option of including their
direct reports or their entire organization.  You can also choose to add a group to this workspace.
If you know you need a person with specific expertise but don't know who  that person is, you can
try the Search Interests and Expertise function or New Connection Request to find the appropriate
person.
Assign tasks to the newly added  participants.
This task has been assigned from the ${sys.workspace.name} workspace.
</description>
       <startTime offsetUnit="SECOND"><offset>1</offset></startTime>
       <dueTime offsetUnit="DAY"><offset>1</offset></dueTime>
       <organizer type="USER"><cen>${sys.workspace.owner.collabid}</cen></organizer>
      </todo>
  </defaultTaskList>
  <defaultWikiFolder>
      <name>Wiki</name>
      <description>default folder for wiki pages</description>
      <entities>
        <wikiPage>
          <name>Home</name>
          <description>Home page</description>
          <extractReferences>T</extractReferences>          <body>
            <inlineBody>
=Community Charter=

(insert the community charter and background information here)

=Shortcuts to Relevant Information=

|Wiki Pages|[[FAQ|FAQ]]\\[[Best Practices|Best Practices]]\\(insert other links here)|
|Documents|(insert links here)|
|Other Sites|(insert links here)|

=Help=

Here is some basic information about this workspace

|End-user Help|End-user help is always available through the Help icon in the upper-right corner.|
|Calendar Enrollment|Participants of this workspace are not automatically enrolled in its calendar.
Events created in this workspace will not be added to participants personal calendars unless they
enroll themselves. Participants can enroll themselves in the Overview page. Alternatively,
workspace coordinators can access the Settings to change the default settings for new
participants.|
|Library|By default, the Library contains a Documents folder and a Public Documents folder. The
Documents folder is used for sharing documents with the team. The Public Documents folder is
available to any authenticated user, provided you share the URL to the folder or the document.|
|Forums|This workspace contains two forums  a Best Practices forum and a Discussions forum. In the
Best Practices forum,  participants can discusse and contribute best practices. The Discussions
forum is can be used for generic discussions. By default, the Forum tool displays the list of
forums in this workspaces. Workspace coordinators can modify this setting in the Defaults tab of
the Settings page.|
|Tags|Tags allow workspace members to describe and organize various objects like wiki pages,
topics, documents and tasks.  As a specific tag is applied by more users to an object, the strength
of that tag as a descriptor for that object increases.  Tags enable easy discovery of related
content and object-agnostic navigation.|
|Enable Public Access|Workspace coordinators can access the Settings to enable Public Access for
this workspace and specify the type of access that non-participants will have.  Non-participants
will be able to access this workspace if they are given the URL.  Additionally, non-participants
will see search results with wiki pages and documents from this workspace.|

</inlineBody>
          </body>
          <isDefaultWikiPage>true</isDefaultWikiPage>
        </wikiPage>
       <wikiPage>
          <name>FAQ</name>
          <description>null</description>
          <extractReferences>T</extractReferences>          <body>
            <inlineBody>
&lt;&lt;toc>>
=(insert your first question here)=

(insert the answer to your first question here)

=(insert your second question here)=

(insert the answer to your second question here)

=(insert your third question here)=

(insert the answer to your third question here)
           </inlineBody>
          </body>
        </wikiPage>
      <wikiPage>
          <name>Best Practices</name>
          <description>null</description>
          <extractReferences>T</extractReferences>          <body>
            <inlineBody>
&lt;&lt;toc>>
The purpose of this page is to provide a framework to capture best practices in your community.

=(insert first best practice title)=

==Point of Contact==

(insert name, email address, phone number of contact person for this best practice)

==Description and Details==

(insert a description and details about this best practice)

==Benefits==

(briefly describe the benefits derived from implementing this best practice)
```

```
==Issues and Lessons Learned==

(briefly describe any problems or issues experienced with the best practice that, if avoided, would
make the implementation of this best practice easier in the future)


=(insert second best practice title)=

==Point of Contact==

(insert name, email address, phone number of contact person for this best practice)

==Description and Details==

(insert a description and details about this best practice)

==Benefits==

(briefly describe the benefits derived from implementing this best practice)

==Issues and Lessons Learned==

(briefly describe any problems or issues experienced with the best practice that, if avoided, would
make the implementation of this best practice easier in the future)
           </inlineBody>
          </body>
        </wikiPage>
      </entities>
    </defaultWikiFolder>
  <entities>
     <folder id="public_documents_folder">
        <name>Public Documents</name>
        <description>A folder for workspace documents that you want to share with people that are
not members of this workspace. These users must be logged into the system to  view these documents,
but they don&#39;t have to be members of this workspace.</description>
        <appliedSensitivity><entityTemplateBodyId>public_
sensitivity</entityTemplateBodyId></appliedSensitivity>
     </folder>
     <forum>
        <name>Best Practices</name>
        <description>Forum for discussing best practices with the community.</description>
     </forum>
  </entities>
</body>
</teamWorkspaceTemplate>
```

7

Managing Oracle Beehive Mobility Services

This module describes how to perform administration tasks relating to Oracle Beehive Mobility Services. The module contains the following topics:

- Introduction
- Controlling Mobile Device Types
- Controlling Mobile Applications
- Managing Language Packs for Oracle Beehive Mobile Client Applications
- Configuring Mobile Services
- Deploying Oracle Beehive on iPhone and Blackberry

Introduction

Oracle Beehive Mobility Services are available for use by end-users immediately following Oracle Beehive installation. Although additional configuration is not required for users to retrieve their e-mail and synchronize their calendar data, Oracle Beehive administrators may want to control certain actions, impose restrictions, or update applications.

This module explains how to perform administrative tasks for Oracle Beehive Mobility Services using the `beectl` command-line tool. You can also perform most of these tasks using Oracle Beekeeper.

See Also:

- "Oracle Beehive Command-Line Utility" in the *Oracle Beehive Administrator's Reference Guide*
- *Oracle Beekeeper Online Help*
- *Oracle Beehive Registering and Configuring Mobile Devices*, available on the Oracle Technology Network website at the following URL:

  `http://www.oracle.com/technology/products/beehive/beehive_users/2_0/mobile.htm`

Controlling Mobile Device Types

By default, Oracle Beehive allows a specific list of mobile device types to synchronize with the Mobile Services. The following topics describe how to manage the list of device types:

- Uploading a Device Profile

- Customizing Device Profiles
- Adding a New Device Type to a Profile

Uploading a Device Profile

Occasionally, you may need to upload a new device profile file to accommodate new device types that are available in the mobile market, or apply changes after updating an existing device profile file. Device profile files contain device identification information, and various configuration parameters specific to a device or device family.

To upload a device profile file to the Device Management Service using `beectl`:

1. Save the device profile XML file in a directory accessible by the Oracle user.
2. Use the `beectl upload_device_profiles` command to upload the new device profile:

   ```
   beectl> upload_device_profiles --file <file>
   ```

   Where *<file>* represents the full path and file name of the device profile file saved in Step 1.

> **Note:** New Oracle Beehive device profiles may be made available periodically by Oracle in subsequent patches.

Customizing Device Profiles

Device profile files are located in the `$ORACLE_HOME/beehive/seed/oma` directory of your Oracle Beehive deployment. You can customize the default values in these files to accommodate users' needs.

To customize the device profile defaults using beectl:

1. Open the `$ORACLE_HOME/beehive/seed/oma/`*<deviceprofile>*`.xml` file with a text editor.

   Where *<deviceprofile>* represents the name of the device profile file that you want to configure.
2. Locate the `<Configuration>` section of the file. Within this section various `<PreferenceSet>` sections exist. Each configurable attribute is defined in an `<AttributeDefinitionName>` XML tag. Table 7–1, " `<Configuration>` Attributes in a Device Profile File" lists the configurable attributes in the `<Configuration>` section.

   > **Caution:** Only make changes to the content in the <Configuration> section. Making changes to other parts of the file may be harmful.

3. To change the value of an attribute, modify the value surrounded by the `<DefaultValue>` XML tag within the appropriate `<AttributeDefinitionName>` section.
4. Repeat Step 3 for any configurable attribute that you want to customize.
5. Save and exit the device profile file.

Table 7–1 `<Configuration>` ***Attributes in a Device Profile File***

Preference Set	Attribute	Description	Accepted Values
Oma	max_object_size	Maximum object size allowed in bytes.	Positive integer
Oma	max_message_size	Maximum message size allowed in bytes.	Positive integer
Event	sync_range_back	Specify number of days in the past that should be synchronized. **See Also:** sync_range_forward, del_out_of_range.	Positive integer
Event	sync_range_forward	Specify number of days in the future that should be synchronized. **See Also:** sync_range_back, del_out_of_range.	Positive integer
Event	del_out_of_range	Delete events on the mobile device that appear outside of the boundaries of the sync_range_back and sync_range_forward attributes. **See Also:** sync_range_back, sync_range_forward.	true, false
Event	want_refused_entries	Allow refused events to be synchronized with your mobile device.	true, false
Event	want_default_alarms	Assign the default alarm to events. **See Also:**default_alarm.	true, false
Event	default_alarm	The time before an event begins, in minutes, when an alarm should be triggered. **See Also:** want_default_alarms.	Positive integer
Event	conflict_resolution	Specify the what entry should take precedence if two entries have been modified between a synchronization. When set to SERVER, the entry on Oracle Beehive will take precedence over the entry on the device.	SERVER, CLIENT
Task	sync_range_back	Specify number of days in the past that should be synchronized. **See Also:** sync_range_forward, del_out_of_range.	Positive integer
Task	sync_range_forward	Specify number of days in the future that should be synchronized. **See Also:** sync_range_back, del_out_of_range.	Positive integer
Task	del_out_of_range	Delete tasks on the mobile device that appear outside of the boundaries of the sync_range_back and sync_range_forward attributes. **See Also:** sync_range_back, sync_range_forward.	true, false
Task	want_refused_entries	Allow refused tasks to be synchronized with your mobile device.	true, false

Table 7–1 (Cont.) `<Configuration>` Attributes in a Device Profile File

Preference Set	Attribute	Description	Accepted Values
`Task`	`want_default_alarms`	Assign the default alarm to tasks. **See Also:** `default_alarm`.	Positive integer
`Task`	`default_alarm`	The time before a task is due, in minutes, when an alarm should be triggered. **See Also:** `want_default_alarms`.	Positive integer
`Task`	`confict_resolution`	Specify what entry should take precedence if two entries have been modified between a synchronization. When set to `SERVER`, the task on Oracle Beehive will take precedence over the task on the device.	`SERVER`, `CLIENT`
`Email`	`sync_range_back`	Specify number of days in the past that should be synchronized.	Positive integer
`Email`	`limit`	The limit, in bytes, of e-mail that can be synchronized.	Positive integer
`Email`	`want_attachements`	Allow synchronization of attachments.	`true`, `false`
`Contact`	`categories`	Specify the contact categories that should be synchronized. When a asterisk (`*`) is specified, all categories are synchronized. To specify a single category or multiple categories, the values must be surrounded by quotes, and separated by commas. For example, if only the `business` and `corporate` type categories should be synchronized the argument for this attribute should be `"business, corporate"`.	`*`, Category names separated by commas (,) surrounded by quotes ("")
`Contact`	`conflict_resolution`	Specify what entry should take precedence if two entries have been modified between a synchronization. When set to `SERVER`, the contact on Oracle Beehive will take precedence over the contact on the device.	`SERVER`, `CLIENT`

Adding a New Device Type to a Profile

A device type is a specific model in a family of devices, and is defined in a device profile file. For example, a new model of a mobile phone that a particular vendor has recently released.

By default, Oracle Beehive does not allow uncertified devices to access the Mobility Services. However, it is possible to add a new device to a device profile file.

> **Note:** As an alternative to manually adding new device types to a profile, Oracle recommends uploading new device profiles made available through certified Oracle Beehive patches. For more information about uploading new device profiles see "Uploading a Device Profile" on page 7-2.

To add a new device to a device profile file using beectl:

1. Temporarily allow uncertified devices to access Oracle Beehive Mobility Services by executing the following command:

   ```
   beectl> modify_property --component _DeviceManagementService --name
   UncertifiedDeviceAllowed --value true
   ```

2. Temporarily enable SyncML logging to discover the device information by executing the following command:

   ```
   beectl> modify_property --component _OmaService --name SyncmlLogRequired
   --value true
   ```

3. Run the following command to activate your proposed configuration changes:

   ```
   beectl> activate_configuration
   ```

   > **Note:** Oracle recommends waiting a few minutes before proceeding to the following step to ensure that the changes have been applied.

4. Synchronize the new device with Oracle Beehive. To ease the retrieval of device information in Step 7, take note of the time the synchronization was initiated.

5. Retrieve recent SyncML log messages using the following command:

   ```
   beectl> download_syncml_messages --directory <path> --date <YYYY-MM-DD>
   ```

   Where <*path*> represents the path to the directory where the SyncML messages will be stored, and <*YYYY-MM-DD*> represents the current date.

6. Open the SyncML messages file downloaded in Step 5. The file will be located in the **--directory** <*path*> argument specified in Step 5.

7. Locate the device SyncML message in the file by looking for the time at which the synchronization attempt was initiated. The device information will be presented in a way similar to the following example:

   ```
   <Item>
           <Source>
             <LocURI>./devinf12</LocURI>
           </Source>
           <Data>
             <![CDATA[<DevInf><VerDTD>1.2</VerDTD><Man>MySync
   Client</Man><Mod>MySync Client 123</Mod><OEM>Synthesis
   AG</OEM><FwV>5.1.195</FwV><SwV>3.0.2.4</SwV>.....</DevInf>]]>
           </Data>
   </Item>
   ```

8. Take note of the values associated with the following XML tags: `<VerDTD>`, `<Man>` `<Mod>`, and `<SwV>`.

9. Close the SyncML message file.

10. Open the `$ORACLE_HOME/beehive/seed/oma/<`*deviceprofile*`>.xml` file with a text editor.

    Where <*deviceprofile*> represents the name of the device profile file of the family of the device that you are adding.

11. Within the `<DeviceTypes>` section of the file, add a new `<DeviceType>` section, including the information noted in Step 7.

    For example, using the information gathered in Step 7, the following entry could be added to the `<DeviceTypes>` section:

    ```
    <DeviceType>
    <DeviceProfileName>MySyncClient</DeviceProfileName>
    <Name>MySync Client</Name>
    <DeviceClass></DeviceClass>
    <Processor/>
    <OS/>
    <Dev_inf_dtd_version>1.2</Dev_inf_dtd_version>
    <Model>MySync Client 123</Model>
    <Manufacturer>MySync Client<</Manufacturer>
    </DeviceType>
    ```

12. Save and close the device profile file.

13. Disallow uncertified devices from accessing Oracle Beehive Mobility Services by executing the following command:

    ```
    beectl> modify_property --component _DeviceManagementService --name
    UncertifiedDeviceAllowed --value false
    ```

14. Disable SyncML logging by executing the following command:

    ```
    beectl> modify_property --component _OmaService --name SyncmlLogRequired
    --value false
    ```

15. Run the following command to activate the proposed configuration changes applied in Steps 13 and 14:

    ```
    beectl> activate_configuration
    ```

16. Upload the device profile file saved in Step 12. For more information about uploading a device profile file, refer to the instructions in "Uploading a Device Profile" on page 7-2.

Controlling Mobile Applications

You can make use of the Oracle Beehive mobility services to deploy third-party applications to users' mobile devices. You can also use alternative methods of deploying Oracle Beehive mobile client components.

This section contains the following topics:

- Adding a New Mobile Application for Use
- Deploying Oracle Beehive Mobile Client Components using Storage Cards

Adding a New Mobile Application for Use

Occasionally, you may want to upload and provision new applications to allow users access to more recent versions. New mobile software, for example, is periodically made available by third-party vendors and can be uploaded to Oracle Beehive to allow users to retrieve the software.

To upload new applications to Oracle Beehive using `beectl`:

1. Create a new application `ZIP` archive in an Oracle Beehive directory accessible by the Oracle user.

   The application `ZIP` file must contain the following items:

   - The application
   - A `metadata.xml` file describing the application

   > **Note:** For a sample XML file, view the `metadata.xml` file supplied within an existing `ZIP` file in the `$ORACLE_HOME/beehive/seed/dm` directory. The file should only be used as a guideline. The values within the XML file should be replaced with appropriate values pertaining the application that you want to upload.

2. Execute the following command to upload the new application software:

   ```
   beectl> upload_client_application --file <file>
   ```

   Where <*file*> represents the full path and file name of the new application software saved in Step 1.

3. Using the `list_enterprises` `beectl` command, determine the identifier of the Oracle Beehive enterprise:

   ```
   beectl> list_enterprises
   ```

4. Take note of the identifier of the enterprise to which the application will be provisioned.

5. Using the `add_client_application_provisioning` `beectl` command, provision the application to the enterprise:

   ```
   beectl> add_client_application_provisioning --community <id> --all
   ```

   Where <*id*> represents the enterprise identifier noted in Step 4.

After completing the steps above, the application will be available to all users on the enterprise.

Deploying Oracle Beehive Mobile Client Components using Storage Cards

You can deploy Oracle Beehive mobile client components to users' devices using storage cards. This allows you or your users to install the software without requiring users to download over a network or the Internet.

To deploy Oracle Beehive mobile software using storage cards:

1. On any Oracle Beehive server, copy the file `$BEEHIVE_HOME /beehive/bootstrap/mobile/ommc.exe` to a computer running Microsoft Windows

2. Run the following command from the Windows computer:

   ```
   ommc.exe --storagecard <drive>: --username <username> --dmserverurl
   http://<server:port>/mobiledm/
   ```

For example, if the storage card drive is `F`, username is `john.doe`, and the server and port is `beehive1.example.com:7777`:

```
ommc.exe --storagecard F: --username john.doe --dmserverurl
http://beehive1.example.com:7777/mobiledm/
```

The application will be copied to the storage card and is ready to be used in a mobile device.

3. Insert the storage card into the device, and then access the applications and run them from the storage card. The various Oracle Beehive mobile client components will install with the correct configuration to access your Oracle Beehive server.

> **See Also:** ""Configuring your Apple iPhone or iPad", "Configuring your BlackBerry", Configuring your Windows Mobile Professional (Pocket PC) and Standard (Smartphone)", and "Configuring your Nokia Series 60" in the *Registering and Configuring Mobile Devices Help*, available on the Oracle Technology Network website at the following URL:
>
> `http://www.oracle.com/technology/products/beehive/beehive_users/2_0/mobile.htm`

Managing Language Packs for Oracle Beehive Mobile Client Applications

Oracle Beehive mobile client applications include language packs and their translation files, which are automatically uploaded during the Oracle Beehive installation process. You can customize the language packs that Oracle Beehive provides as well as others that it does not.

This section contains the following topics:

- Mobile Client Application Language Packs Provided by Oracle Beehive
- Customizing Language Packs and Translation Files for Oracle Beehive Mobile Client Applications

Mobile Client Application Language Packs Provided by Oracle Beehive

Oracle Beehive provides the following language packs for its mobile client applications:

- Chinese (Simplified)
- Chinese (Traditional)
- French
- German
- Italian
- Japanese
- Korean
- Portuguese (Brazilian)
- Spanish

To customize the translation files of a language pack, refer to "Customizing Language Packs and Translation Files for Oracle Beehive Mobile Client Applications".

Customizing Language Packs and Translation Files for Oracle Beehive Mobile Client Applications

To customize a language pack, you modify one or more of its translation files. You then upload the customized translation files using one of the following containers:

- **A patch set:** Used for language packs provided by Oracle Beehive. For more information, please refer to "Creating and Uploading a Patch Set for a Language Pack Provided by Oracle Beehive".
- **A new language pack:** Used for language packs that are not provided by Oracle Beehive. For more information, please refer to "Creating, Uploading, and Provisioning a Language Pack Not Provided by Oracle Beehive".

> **Note:** For the list of language packs that Oracle Beehive provides, please refer to "Mobile Client Application Language Packs Provided by Oracle Beehive".

Creating and Uploading a Patch Set for a Language Pack Provided by Oracle Beehive

To customize the translation files for a language pack provided by Oracle Beehive, you create and upload a patch set.

To create and upload a patch set:

1. Modify the translation file of a supported language pack, as follows:
   a. Access the zip file of the mobile client application, located in the `$ORACLE_HOME/beehive/seed/dm` directory. For example, the translation files for the Mobile Mail plug-in for Windows Mobile devices appear in `pushmail_release.PPC5.0_ARM.element.zip`.
   b. From the zip file, open the XLIFF file for one of the languages that Oracle Beehive supports by default. For example, the French translation file for the Mobile Mail plug-in is `oracle.ocs.mobileclient.wince.pushmail_fr.xlf`.
   c. Modify the translation strings in the file, as needed.
   d. Save the file in UTF-8 encoding. All XLIFF files must be UTF-8 encoded.
   e. Repeat these steps as necessary for the translation files for other languages.
2. Create a new `zip` file that contains all customized translation files. Ensure that the new `zip` file is located in an Oracle Beehive directory that is accessible by the Oracle user.
3. Create a new `metadata.xml` to describe the new patch set, as follows:
   a. Access the existing zip file of the mobile client application, located in the `$ORACLE_HOME/beehive/seed/dm` directory.
   b. From the `zip` file, open `metadata.xml`.
   c. Increment the value of the `<patchsetnumber>` attribute so that it is *higher than* the current value. All other attributes under the `<property>` element should remain the same as distributed in the original `metadata.xml` file.

      For example, a `metadata.xml` for a patch set targeted towards Windows Mobile 5.0 devices and that contains customized French and Japanese translation files will look similar to the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<application>
    <property>
        <name> Mobile Mail </name>
        <description> MobileMail Client </description>
        <os> wince5.0  </os>
        <processor>ARM </processor>
        <deviceclass> Smartphone </deviceclass>
        <language> all </language>
        <version> 1.4.0.0.0. </version>
        <versionnumber> 4 </versionnumber>
        <patchsetnumber> 1 </patchsetnumber>
        <vendor> Oracle </vendor>
        <isPlatform> false </isPlatform>
    </property>
    <modules>
        <module>
            <name> oracle.ocs.mobileclient.wince.pushmail_fr.xlf </name>
            <src> . </src>
            <dest> %CSIDL_WINDOWS% </dest>
            <contenttype> text/xml</contenttype>
        </module>
        <module>
            <name> oracle.ocs.mobileclient.wince.pushmail_ja.xlf </name>
            <src> . </src>
            <dest> %CSIDL_WINDOWS% </dest>
            <contenttype> text/xml</contenttype>
        </module>
    </modules>
</application>
```

d. Save the new `metadata.xml`.

e. Add the new `metadata.xml` to the `zip` file created in Step 2.

4. Upload the new patch set, as follows:

   a. Launch the `beectl` command line utility.

   b. Issue the `upload_client_application` command, as follows:

   ```
   beectl> upload_client_application --file <file>
   ```

   Where `<file>` represents the absolute path of the zip file that you created in Step 2.

Creating, Uploading, and Provisioning a Language Pack Not Provided by Oracle Beehive

To customize the translation files for a language pack not provided by Oracle Beehive, you create a new language pack based on an existing one. Once created, you can upload and provision the new language pack.

To create, upload, and provision a new language pack:

1. Modify the translation file of a supported language pack with the strings required in the new language pack, as follows:

   a. Access the zip file of the mobile client application, located in the `$ORACLE_HOME/beehive/seed/dm` directory. For example, the translation files for the Mobile Mail plug-in for Windows Mobile 5.0 devices appear in `pushmail_release.PPC5.0_ARM.element.zip`.

b. From the zip file, open the XLIFF file for one of the languages that Oracle Beehive supports by default. For example, the French translation file for the the Mobile Mail plug-in is `oracle.ocs.mobileclient.wince.pushmail_fr.xlf`.

c. Modify the translation strings, as needed.

d. Modify the `target-language` attribute to specify the new target language and country. For example, if you modify the French translation file for the Mobile Mail plug-in with Danish strings, change the `target-language` attribute from "fr-FR" to "dk-DK".

e. Rename the XLIFF file by replacing the original language code with the new language code and, if appropriate, the new country code.

> **Note:** XLIFF file naming conventions follow the Java standard, which supports a two-letter lowercase language code (ISO 639) and a two-letter uppercase country code (ISO 3166). For example, if you modify the French translation file for the Mobile Mail plug-in with Danish strings, rename the file and save it in UTF-8 encoding as `oracle.ocs.mobileclient.wince.pushmail_dk.xlf`. Or, if you create a Canadian French translation file, save it as `oracle.ocs.mobileclient.wince.pushmail_fr_CA.xlf`.

2. Create a new `zip` file that contains all customized translation files. Ensure that the new `zip` file is located in an Oracle Beehive directory that is accessible by the Oracle user.

3. Create a new `metadata.xml` to describe the new language pack, as follows:

a. Access the existing zip file of the mobile client application, located in the `$ORACLE_HOME/beehive/seed/dm` directory.

b. From the `zip` file, open `metadata.xml`.

c. Replace the values of the `<name>` and `<description>` attributes with the name and description of the new language pack. All other attributes under the `<property>` element should remain the same as distributed in the original `metadata.xml` file.

For example, a `metadata.xml` for a language pack targeted towards Windows Mobile 5.0 devices and that contains customized Danish translation files will look similar to the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<application>
    <property>
        <name> Mobile Mail Langugage Pack - DK </name>
        <description> MobileMail Client Danish Language Pack</description>
        <os> wince5.0  </os>
        <processor>ARM </processor>
        <deviceclass> Smartphone</deviceclass>
        <language> all </language>
        <version> 1.4.0.0.0. </version>
        <versionnumber> 4 </versionnumber>
        <patchsetnumber> 0 </patchsetnumber>
        <vendor> Oracle </vendor>
        <isPlatform> false </isPlatform>
    </property>
    <modules>
```

```
            <module>
                <name> oracle.ocs.mobileclient.wince.pushmail_dk.xlf </name>
                <src> . </src>
                <dest> %CSIDL_WINDOWS% </dest>
                <contenttype> text/xml</contenttype>
            </module>
        </modules>
    </application>
```

d. Save the new `metadata.xml`.

e. Add the new `metadata.xml` to the zip file created in Step 2.

4. Upload the new language pack, as follows:

   a. Launch the `beectl` command line utility.

   b. Issue the `upload_client_application` command, as follows:

   ```
   beectl> upload_client_application --file <file>
   ```

   Where `<file>` represents the absolute path of the `zip` file that you created in Step 2.

5. Provision the new language pack, as follows:

   a. In the `beectl` command line utility, issue the `list_enterprises` command to determine the identifier of the Oracle Beehive enterprise:

   ```
   beectl> list_enterprises --entity_format id
   ```

   b. Take note of the identifier of the enterprise to which the language pack will be provisioned.

   c. Issue the `add_client_application_provisioning` command to provision the language pack to the enterprise:

   ```
   beectl> add_client_application_provisioning --community
   <id> --all
   ```

   Where `<id>` represents the enterprise identifier

Configuring Mobile Services

You can control the service-level configuration of the Mobile Services by making adjustments to service properties. This section contains the following topics related to managing Mobile Services properties:

- Listing Mobile Services Configurable Properties
- Configuring the Mobile Data Sync Service
- Configuring the Mobile Mail Service

Listing Mobile Services Configurable Properties

To list the configurable properties of a service, the component identifier is required. To obtain the component identifier, use the `list_components` command with the **--type** option. The mobile service component types are:

- `OmaService`: the Mobile Data Sync Service
- `PushMailService`: the Mobile Mail Service
- `MobileDmService`: the Mobile Device Management Service

- `PushService`: the Mobile Push Service

To list the configurable properties for a service:

1. Determine the component identifier by running the `beectl list_components` command, using the component identifier for the `--type` option. For example, to list the component identifier of the Mobile Data Sync Service:

   ```
   beectl> list_components --type OmaService
   -----------------------------------------------
   | Component Type       | Component Identifier |
   -----------------------------------------------
   | OmaService           | _OmaService          |
   -----------------------------------------------
   ```

2. Using the component identifier determined in Step 1, list the configurable properties for the service by running the `beectl list_properties` command. For example:

   ```
   beectl> list_properties --component _OmaService
   ```

   The command will return a list of properties for the service.

   > **See Also:** For more information about managing component properties, and a complete list of the properties for each component, see Chapter 4, "Oracle Beehive Property Reference," of the *Oracle Beehive Administrator's Reference Guide*.

Configuring the Mobile Data Sync Service

Oracle Beehive allows you to configure certain Mobile Data Sync Service properties. This section explains how to modify Data Sync properties using `beectl` commands, and contains the following topics:

- Controlling Sychronized Data Types
- Controlling MD5 Authentication
- Controlling Synchronization Ranges

Controlling Sychronized Data Types

Oracle Beehive administrators can control the type of data that users are allowed to synchronize. By default users are allowed to synchronize e-mail, calendar (including events and tasks), and contacts.

The items listed in Table 7–2 represent data type properties that can be modified using the `modify_property` command with the **--component**, **--name**, and **--value** options.

Table 7–2 Data Type Properties

Data Type Properties	Accepted Values

`CalendarSyncEnabled`	`true`
	Enables event and task synchronization.
	`false`
	Disables event and task synchronization.

`ContactsSyncEnabled`	`true`
	Enables contact synchronization.
	`false`
	Disables contact synchronization.

Table 7–2 (Cont.) Data Type Properties

Data Type Properties	Accepted Values

`EmailSyncEnabled`	`true`
	Enables e-mail synchronization.
	`false`
	Disables e-mail synchronization.

To enable or disable synchronization of data types:

1. Enable or disable the synchronization of data types using the following command:

   ```
   beectl> modify_property --component _OmaService --name <DataTypeProperty>
   --value <value>
   ```

   Where <*DataTypeProperty*> represents a data type property listed in Table 7–2, and <*value*> represents either `true` (to enable) or `false` (to disable).

2. Activate the proposed configuration by executing the following command:

   ```
   beectl> activate_configuration
   ```

Example 7–1 displays how to disable the contacts synchronization data type property. The resulting output of the command is also displayed.

Example 7–1 Disabling the Contacts Synchronization Data Type

```
beectl> modify_property --component _OmaService --name ContactsSyncEnabled --value
false
Changes to configuration repository are not activated.
Successfully stored the property for component id
1e54ba56-7448-4849-b987-8dda59d26f4d.
```

Note: You must run the `beectl activate_configuration` command after modifying a property, to apply your proposed configuration changes to the active configuration.

Controlling MD5 Authentication

The Mobile Data Sync service supports MD5 and basic authentication. Basic authentication is clear text-based authentication whereas MD5 authentication is more secure. By default, basic authentication is used with Mobile Data Sync.

Many devices support MD5 authentication; however, by default, the Mobile Data Sync service does not allow MD5 authentication. The Mobile Data Sync service can be configured, globally or per device profile, to accept MD5 authentication requests.

This section contains the following topics:

- Controlling MD5 Authentication for all Devices
- Controlling MD5 Authentication for a Specific Device Profile

Note: The Oracle Beehive Authentication service may not be able to support MD5 when configured with certain third-party LDAP servers.

If it is not supported, then the Mobile Data Sync service `Md5Supported` property should be set to `false`.

Controlling MD5 Authentication for all Devices There are two service properties that control authentication requirements at the Mobile Data Sync service-level: `Md5Supported` and `Md5Required`.

> **See Also:** For a complete list of properties specific to the Mobile Data Sync service, see "Listing Mobile Services Configurable Properties".

The `Md5Supported` property controls whether MD5 authentication is allowed. The `Md5Required` property, when set to `true`, enforces MD5 authentication for all devices.

To enforce MD5 Authentication for all devices using the Mobile Data Sync service:

1. Allow MD5 authentication using following command:

   ```
   beectl> modify_property --component _OmaService --name Md5Supported --value
   true
   ```

2. Force all devices to use MD5 authentication using following command:

   ```
   beectl> modify_property --component _OmaService --name Md5Required --value true
   ```

3. Activate the proposed property changes by executing the following command:

   ```
   beectl> activate_configuration
   ```

> **Note:** When using the above settings, if a device does not support MD5, it will not be able to authenticate with the Oracle Beehive Mobile Data Sync service.
>
> To allow MD5 and basic authentication, omit Step 2.

Controlling MD5 Authentication for a Specific Device Profile To control MD5 authentication requests at the device profile level:

1. Open a device profile file with a text editor. Device profile files are located in the `$ORACLE_HOME/beehive/seed/oma` directory.

2. Locate the section of the file with the following text:

   ```
   <Capability>
        <Name>oma.support_md5</Name>
        <Type>boolean</Type>
        <Value>true</Value>
    </Capability>
   ```

3. Modify the argument of the `<Value>` XML tag located in Step 2 to `true` or `false`, depending on the desired outcome.

> **Note:** When this tag is set to `true`, devices are forced to use MD5 authentication. They will only be forced to use MD5 authentication if the Mobile Data Sync service `Md5Supported` property is set to `true`.

4. Save and exit the device profile file.

5. Upload the device profile file to Oracle Beehive. For information about uploading the device profile, see Uploading a Device Profile.

Controlling Synchronization Ranges

You can control the maximum number of days, in the past or the future, that users are allowed to synchronize. By default the synchronization range depends on the type of device a user is using. Each device profile contains a default range appropriately adjusted to the capabilities of the device. You can change the default range for a particular device type by editing the range within the profile. Users can request a larger range by specifying a range within the Mobile Data Sync URI. By default the maximum range allowed is 365 days in the past, and 365 days in the future.

> **Note:** The synchronization range discussed in this section controls the range limits, and does not affect default values.

To modify the maximum date range allowed for data synchronization using beectl:

1. Modify the synchronization range using the following command:

   ```
   beectl> modify_property --component _OmaService --name <PropertyName> --value
   <value>
   ```

   Where <*PropertyName*> represents a `MaxSyncRangeBack` (for the maximum number of days in the past) or `MaxSyncRangeForward` (for the maximum number of days in the future), and <*value*> represents a positive integer indicating the number of days.

2. Activate the proposed property changes by executing the following command:

   ```
   beectl> activate_configuration
   ```

Example 7–2 displays how to enforce a data synchronization limit of four weeks in the past. The resulting output of the command is also displayed.

Example 7–2 Enforcing a Four Week Data Synchronization Limit

```
beectl> modify_property --component _OmaService --name MaxSyncRangeBack --value 28
Changes to configuration repository are not activated.
Successfully stored the property for component id
1e54ba56-7448-4849-b987-8dda59d26f4d.
```

> **Note:** You must run the `beectl activate_configuration` command after modifying a property, to apply your proposed configuration changes to the active configuration.

Configuring the Mobile Mail Service

Oracle Beehive allows you to configure certain Mobile Mail Service properties.

> **Note:** When changing Mobile Mail service properties, you will be modifying the absolute maximum values. Users will still have the option to change these values on their mobile devices, but will be limited by the Mobile Mail service absolute maximum.

This section explains how to modify Mobile Mail Service properties using `beectl` commands, and contains the following topics:

- Controlling Maximum Number of E-mails Pushed to a Device
- Controlling the Maximum Message Size
- Controlling Past E-mail Push

Controlling Maximum Number of E-mails Pushed to a Device

You can control the maximum number of e-mails that can be pushed to a device at one time. By default 200 e-mail messages can be pushed.

To modify number of e-mails that can be pushed to a device using beectl:

1. Modify the number of e-mails that can be pushed to a device using the following command:

   ```
   beectl> modify_property --component _PushMailService --name MaxInboxMessages
   --value <value>
   ```

   Where <*value*> represents an integer that is greater than 200.

2. Activate the proposed property changes by executing the following command:

   ```
   beectl> activate_configuration
   ```

Example 7–3 displays how to change the maximum number of e-mails that can be pushed to a device to 500. The resulting output of the command is also displayed.

Example 7–3   Enforce a Maximum Number of E-Mails to Push to a Device

```
beectl> modify_property --component _PushMailService --name MaxInboxMessages
--value 500
Changes to configuration repository are not activated.
Successfully stored the property for component id
ae373546-48e3-442d-8177-ae7e8f02e31e.
```

> **Note:** You must run the `beectl activate_configuration` command after modifying a property, to apply your proposed configuration changes to the active configuration.

Controlling the Maximum Message Size

You can restrict e-mail messages of a certain size from being pushed to a device at one time. By default the maximum e-mail message size that can be pushed to a device is 50 KB.

To the modify maximum e-mail message size that can be pushed to a device using beectl:

1. Modify the maximum e-mail message size using the following command:

   ```
   beectl> modify_property --component _PushMailService --name MaxMessageSize
   --value <value>
   ```

   Where <*value*> represents a positive integer.

2. Activate the proposed property changes by executing the following command:

   ```
   beectl> activate_configuration
   ```

Example 7–4 displays how to modify the maximum e-mail message size to 100. The resulting output of the command is also displayed.

Example 7–4 Modify the Maximum Message Size

```
beectl> modify_property --component _PushMailService --name MaxMessageSize
--value 100
Changes to configuration repository are not activated.
Successfully stored the property for component id
ae373546-48e3-442d-8177-ae7e8f02e31e.
```

> **Note:** You must run the `beectl activate_configuration` command after modifying a property, to apply your proposed configuration changes to the active configuration.

Controlling Past E-mail Push

You can control the maximum number of days in the past for which to push e-mail to a device at one time. By default the maximum number of days in the past is seven.

To modify the maximum number of days in the past for which to push e-mail to a device using `beectl`:

1. Modify the maximum number of days in the past using the following command:

   ```
   beectl> modify_property --component _PushMailService --name NumberDaysPast
   --value <value>
   ```

   Where <*value*> represents a positive integer, greater than 7.

2. Activate the proposed property changes by executing the following command:

   ```
   beectl> activate_configuration
   ```

Example 7–5 displays how to modify the maximum number of days in the past of e-mail that can be pushed to a device to 14. The resulting output of the command is also displayed.

Example 7–5 Modify Past E-mail Push

```
beectl> modify_property --component _PushMailService --name NumberDaysPast --value
14
Changes to configuration repository are not activated.
Successfully stored the property for component id
ae373546-48e3-442d-8177-ae7e8f02e31e.
```

> **Note:** You must run the `beectl activate_configuration` command after modifying a property, to apply your proposed configuration changes to the active configuration.

Deploying Oracle Beehive on iPhone and Blackberry

This section includes the following topics:

- Deploying an iPhone Configuration File
- Deploying Oracle Beehive on Blackberry

Deploying an iPhone Configuration File

Over-the-air access to an Apple iPhone XML Configuration file during device registration can be setup such that all necessary IMAP, SMTP, and CalDAV settings are provisioned automatically for the user. However, this file must be generated using the Apple iPhone Configuration Utility (iPCU) and then placed where Oracle Beehive can access it.

The iPhone Configuration Utility (iPCU) lets you easily create, maintain, encrypt, and install configuration profiles, track and install provisioning profiles and authorized applications, and capture device information including console logs.

Configuration profiles are XML files that contain device security policies, VPN configuration information, Wi-Fi settings, APN settings, mail and calendar settings, and certificates that permit iPhone and iPod touch to work with your enterprise systems.

In order for Oracle Beehive's device registration process to deliver an iPhone configuration file to user's iPhones it needs to first be generated using Apple's iPCU and then uploaded into Oracle Beehive.

1. Generate a configuration file. Be sure to include settings for IMAP, SMTP, and CalDAV.
2. In order to make the generated XML file generic such that it can be used for all Oracle Beehive users, you need to edit the XML and replace the hard coded values with tokens that Oracle Beehive can replace at runtime when a user requests to upload the file.

   > **Note:** You must generate the profile with a security option of none. If you wish to sign your profile you can enter explicit values for each value but you will need to leave the user name blank and the user will be asked for it (as they are asked for their password) when they download the profile.

   The following sample samples shows the end result with the tokens highlighted in bold:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>PayloadContent</key>
    <array>
           <dict>
                <key>EmailAccountDescription</key>
                <string>Oracle Beehive Mail</string>
                <key>EmailAccountName</key>
                <string>$$DISPLAY_NAME$$</string>
                <key>EmailAccountType</key>
                <string>EmailTypeIMAP</string>
                <string>EmailTypeIMAP</string>
                <key>EmailAddress</key>
                <string>$$EMAIL_ADDRESS$$</string>
                <key>IncomingMailServerAuthentication</key>
                <string>EmailAuthPassword</string>
                <key>IncomingMailServerHostName</key>
                <string>$$IMAP_ADDRESS$$</string>
                <key>IncomingMailServerPortNumber</key>
                <real>993</real>
                <key>IncomingMailServerUseSSL</key>
                <true/>
                <key>IncomingMailServerUsername</key>
                <string>$$EMAIL_ADDRESS$$</string>
                <key>OutgoingMailServerAuthentication</key>
                <string>EmailAuthPassword</string>
                <key>OutgoingMailServerHostName</key>
                <string>$$SMTP_ADDRESS$$</string>
                <key>OutgoingMailServerPortNumber</key>
                <real>465</real>
                <key>OutgoingMailServerUseSSL</key>
                <true/>
                <key>OutgoingMailServerUsername</key>
                <string>$$EMAIL_ADDRESS$$</string>
                <key>OutgoingPasswordSameAsIncomingPassword</key>
                <true/>
                <key>PayloadDescription</key>
                <string>Configures email account.</string>
                <key>PayloadDisplayName</key>
                <string>Oracle Beehive Mail</string>
                <key>PayloadIdentifier</key>
                <string>com.oracle.beehive.email</string>
                <key>PayloadOrganization</key>
                <string></string>
                <key>PayloadType</key>
                <string>com.apple.mail.managed</string>
                <key>PayloadUUID</key>
                <string>74753AFC-0702-4174-A3CA-621A603D2AF2</string>
                <key>PayloadVersion</key>
                <integer>1</integer>
           </dict>
           <dict>
                <key>CalDAVAccountDescription</key>
                <string>Oracle Beehive CalDAV</string>
                <key>CalDAVHostName</key>
                <string>$$BEEHIVE_SERVER_ADDRESS$$</string>
                <key>CalDAVPort</key>
                <integer>443</integer>
                <key>CalDAVPrincipalURL</key>
<string>$$BEEHIVE_URL$$caldav/$$ENPR_NAME_
URLENC$$/principals/individuals/$$EMAIL_ADDRESS$$</string>
                <key>CalDAVUseSSL</key>
                <true/>
                <key>CalDAVUsername</key>
                <string>$$USERNAME$$</string>
                <key>PayloadDescription</key>
                <string>Configures CalDAV account.</string>
                <key>PayloadDisplayName</key>
                <string>CalDAV (Oracle Beehive CalDAV)</string>
                <key>PayloadIdentifier</key>
                <string>com.oracle.beehive.caldav</string>
                <key>PayloadOrganization</key>
              <string></string>
              <key>PayloadType</key>
              <string>com.apple.caldav.account</string>
              <key>PayloadUUID</key>
              <string>40A76E20-A6DE-4867-845C-549BBD287277</string>
              <key>PayloadVersion</key>
              <integer>1</integer>
       </dict>
   </array>
   <key>PayloadDescription</key>
   <string>Beehive Profile for IMAP/SMTP and CalDAV.</string>
   <key>PayloadDisplayName</key>
   <string>Oracle Beehive</string>
   <key>PayloadIdentifier</key>
   <string>com.oracle.beehive</string>
   <key>PayloadOrganization</key>
   <string></string>
   <key>PayloadRemovalDisallowed</key>
   <false/>
   <key>PayloadType</key>
   <string>Configuration</string>
   <key>PayloadUUID</key>
   <string>B24A2C42-277E-4D46-872E-33917300906C</string>
   <key>PayloadVersion</key>
   <integer>1</integer>
</dict>
</plist>
```

> **Note:** In addition to email and calendar settings, an Oracle Beehive administrator can include any other settings important for their deployment. When the user registers their device with Oracle Beehive and uploads the configuration file they will get the email and calendar settings as well as any additional settings. This can be done as part of one configuration profile or you can upload multiple profiles.

3. To upload the configuration file to Oracle Beehive you need to create a `ZIP` file that contains the edited configuration file as well as a file called `metadata.xml` which should contain the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<application>
    <property>
        <name> Beehive Mobile PIM Bootstrap </name>
        <display_name> Beehive Mail and Calendar Configuration Profile
</display_name>
       <description> Oracle Beehive iPhone Configuration Profile for IMAP/SMTP
and CalDAV. </description>
        <os> iphone </os>
        <processor>all </processor>
        <deviceclass> apple </deviceclass>
        <language> all </language>
        <version> 2.0.0.0.0 </version>
        <versionnumber> 20 </versionnumber>
        <patchsetnumber> 0 </patchsetnumber>
        <vendor> Oracle </vendor>
        <application_type>BOOTSTRAP</application_type>
    </property>
    <modules>
         <module>
             <name>iphone_system.mobileconfig</name>
             <src> . </src>
             <dest> . </dest>
             <contenttype>application/x-apple-aspen-config</contenttype>
         </module>
    </modules>
    <configuration>
        <param name="config_file" value="iphone_system.mobileconfig"/>
        <param name="download_mode" value="file" />
        <param name="replace_tokens" value="true" />
    </configuration>
</application>
```

Note: The display name in the sample above can be customized to the needs of the deployment. Also in this example the configuration file is referred to as `iphone_system.mobileconfig`, be sure to save the config file with this name or make sure to use your own name in the `metadata.xml` file.

4. Upload the configuration file into Oracle Beehive:

   beectl upload_client_application –file iphoneconfig.zip

   With the configuration file now in place Oracle Beehive will automatically start offering it to users who register an iPhone.

Note: In the sample configuration file shown above ports `993` (incoming email), `465` (outgoing email), and `443` (CalDAV over HTTPS), were used. In addition the Beehive Mobile Communicator uses the Beehive secure client port known as BTPS which is usually set to `5224`. In order for iPhone users to have access to the Oracle Beehive functionality described above, these ports must be exposed outside the Enterprise firewall.

In cases where exposing such ports outside the firewall is not possible you may want to consider providing VPN access to your iPhone users. The iPhone Configuration Utility also includes support for Cisco AnyConnect and Juniper Networks SSL VPN clients.

Deploying Oracle Beehive on Blackberry

Before installing and configuring Oracle Beehive on their Blackberry device, users must activate their BlackBerry with BlackBerry® Enterprise Server for MDS Applications.

The BlackBerry device and BlackBerry® Enterprise Server for MDS Applications use symmetric encryption to maintain a secure communication channel between each other. To do that, both ends need to know a shared secret, the encryption key. The way that Research in Motion implements this is to use email to transfer the secret key from device to server.

To start, you need an IMAP account that the BlackBerry® Enterprise Server for MDS Applications can read. The Oracle Beehive administrator creates an account for the user, using their PIN (each BlackBerry has a unique PIN). They then give this to the user (usually verbally in person or over the phone).

The user then goes into their device, to the Enterprise activation screen, enters an email address (the IMAP account mentioned above) and the password supplied by the Oracle Beehive administrator.

The device then encrypts its local private key using the password entered, that then sends that payload to the email address supplied.

BlackBerry® Enterprise Server for MDS Applications monitors the IMAP account. When an email arrives it reads it and attempts to decrypt it with the password supplied. If it works, it then activates the account, and then starts pushing data to the device using the private key supplied by the device.

Follow these steps to deploy Oracle Beehive on Blackberry devices:

1. Create a unique IMAP account within Oracle Beehive for each BlackBerry® Enterprise Server in your environment (For example, `bes@yourorg.com`). Configure each IMAP account to send and receive information in plain text. IMAP port `143` needs to be open from Blackberry® Enterprise Server to Oracle Beehive.

   Note: Do not use a personal IMAP account for BlackBerry Enterprise Server activations. The BlackBerry Messaging Agent searches the mailbox for unread activation messages; if an activation message is marked as read before the BlackBerry Messaging Agent processes the message, enterprise activation does not complete successfully.

2. Once BlackBerry® Enterprise Server for MDS Applications is activated on their BlackBerry, users can proceed to the Mobile Center in Oracle Beehive Central and click on **New** to register their BlackBerry with Oracle Beehive.

   See the *Oracle Beehive Registering and Configuring Mobile Devices* end-user pages for more information on registering and configuring mobile devices and the *Oracle Beehive Using Blackberry* end-user pages for more information on how to access Oracle Beehive from your RIM BlackBerry device using Oracle Beehive Client and Oracle Beehive Communicator.

8

Managing Oracle Beehive E-mail

This module presents instructions for performing a variety of configuration tasks involving how Oracle Beehive handles e-mail. The instructions in this module assume that you are already familiar with the use of beectl commands to set properties of the E-mail Service. For detailed instructions on setting service properties, see "Oracle Beehive Parameter Reference" in the *Oracle Beehive Administrator's Reference Guide*.

For details about exporting and importing individual users' e-mail folders, see "Backing Up and Recovering Individual E-mail Accounts" on page 15-8.

This module contains the following topics:

- Introduction to Oracle Beehive E-mail
- Managing Oracle Beehive E-mail Components
- E-mail Coexistence in a Single Domain

Introduction to Oracle Beehive E-mail

This section contains the following topics:

- About Configuring Oracle Beehive E-mail
- About the Oracle Beehive SMTP Server

About Configuring Oracle Beehive E-mail

Broadly, there are two general categories of Oracle Beehive E-mail configuration properties: those which you can freely modify using any administrator tool, and those which should only be adjusted using Oracle Beekeeper.

- Configuring Oracle Beehive E-mail Parameters using `beectl` or Oracle Beekeeper
- Configuring Complex Rule-Based E-mail Parameters using Oracle Beekeeper

Configuring Oracle Beehive E-mail Parameters using `beectl` or Oracle Beekeeper

Configuration of Oracle Beehive E-mail is accomplished by setting parameters of the E-mail Service. The commands, syntax, and a reference of parameters for all components are provided in Chapter 4, "Oracle Beehive Parameter Reference," of the *Oracle Beehive Administrator's Reference Guide*. You can also configure many of these parameters using Oracle Beekeeper. When you are making simple configuration changes, you can edit parameter values directly using either tool.

Configuring Complex Rule-Based E-mail Parameters using Oracle Beekeeper

Oracle E-mail stores server-side e-mail rules by using XML-formatted code, stored in various E-mail Service parameters. Manual edits to this XML is not supported.

Beginning with Oracle Beehive Release 1, version 1.4, you should configure Oracle E-mail rules using the Oracle Beekeeper administration console. The console will make changes to the XML in the background. This method provides the safest way of modifying the XML-based configuration without introducing errors.

You must first install and configure Oracle Beekeeper, following the instructions in the *Oracle Beehive Installation Guide* for your platform.

Oracle Beekeeper includes integrated help topics for many of the configuration options for the E-mail Service. You should refer to the online help for details about the various configuration options.

About the Oracle Beehive SMTP Server

The SMTP Server component is a robust, scalable, and flexible component of Oracle Beehive that provides open protocol access to the Oracle Beehive E-mail Service. The SMTP Server has been designed to be both Internet and internal facing. The SMTP server can be divided into Virtual Mail Servers (VMSes). Oracle Beehive SMTP Servers may have multiple `Endpoints`, and are configured with dispatch rules. These configuration options provide flexibility to facilitate servicing Internet and internal SMTP clients.

Virtual Mail Servers

A virtual mail server is a component which engages in SMTP conversations with clients. Two VMSes are differentiated by their behavior, which is controlled by many properties. For example, SMTP defines a time-out for inactivity. A VMS servicing the Internet may be configured with a low value for this time-out, to prevent service loss. A VMS servicing internal clients may permit a longer time-out (because internal clients are assumed not to be hostile). Time-outs are only one of more than a dozen properties of a VMS. Note that the primary driver for having VMSes is to facilitate different behavior for internal clients compared to external clients.

In Oracle Beekeeper, configuration options for VMSes are found by selecting the **E-mail Service**, the **Configuration** tab, and then the **SMTP Properties** subtab.

The following general concepts apply to Virtual Mail Servers in Oracle Beehive:

- SMTP Mail (into and out of Oracle Beehive) is processed by a Virtual Mail Server (VMS)
- A VMS is capable of various rule-based processing choices
- In a default Oracle Beehive installation, there are two default VMSes configured: Inbound VMS and Outbound VMS
- Each VMS has two 'matchers,' which can be used to filter which addresses are acceptable:
  - a Sender Matcher
  - a Recipient Matcher

  By default, both matchers are enabled for both default VMSes.

The Virtual Mail Server components of Oracle Beehive make use of an Evaluator to determine where and how e-mail messages are relayed. The VMS Evaluator picks which VMS will be used to process the message based on the source of the message or the Endpoint the client connects to.

In Oracle Beekeeper, the VMS Evaluator is configured using the **VMS Routing** section of the **SMTP Properties** tab.

The rules for the VMS Sender and Recipient matching decide which e-mail messages are allowed to be delivered to an Oracle Beehive recipient, are permitted by Oracle Beehive to be relayed, or will be rejected.

The following are typical types of rules for how Oracle Beehive relays e-mail messages:

- Oracle Beehive accepts e-mail to addresses that it is configured to recognize as local. This generally means Oracle Beehive users, but there may be some special cases where a single domain includes some users in Oracle Beehive and some in another system. In this case, the Oracle Beehive VMS evaluators should be configured so that e-mail may be sent to any address in the domain, and the server determines the correct destination to deliver the message
- Messages that aren't addressed to recipients in your local domains are routed to their destination. If the Oracle Beehive server is connected directly to the Internet, then the server can perform MX resolution through DNS to find the IP address of the recipient's server. If the Oracle Beehive e-mail server is behind a gateway or proxy, Oracle Beehive should relay all outbound traffic to some other "smart host" to send it out to the public Internet
- Messages that aren't **to** your domains, and are also not **from** your local domains, are most likely spam or malicious, and e-mail servers should be configured to prevent or block such traffic

Endpoints

In addition to one or more virtual mail servers, the SMTP server presents "endpoints." Endpoints represent a logical listening point. In this terminology, a web server has two endpoints, one for SSL (port 443) and one for regular HTTP (port 80). In Oracle Beehive, the SMTP server can listen for both traditional TCP traffic from clients (on the traditional port 25 default, or on other ports as well or instead), and for internal traffic from other Oracle Beehive services.

A given SMTP Server may listen for SMTP connections on more than one endpoint, and each endpoint is uniquely identified by a logical name. In most cases, an endpoint works the same as a TCP port. By default, the E-mail Service is configured to "assume single endpoint," meaning it assumes all SMTP connections will be made over a single port.

In Oracle Beekeeper, you can configure endpoints by selecting the **E-mail Service**, selecting the **Configuration** tab, and selecting the **SMTP Properties** subtab.

Dispatch Rules

Dispatch rules map incoming connections to a VMS. The mapping is based on one of the following properties of the incoming connection:

- The Endpoint at which the connection arrived (if you have multiple endpoints)
- The network (IP) address of the client
- The sender’s e-mail address contained in the e-mail header

The dispatch rules accept as input these data points, and return the appropriate VMS. Note that in the configuration, the Email Service parameter that stores dispatch rules is called `VMSEvaluators`.

In Oracle Beekeeper, you can configure dispatch rules by selecting the **E-mail Service**, selecting the **Configuration** tab, and selecting the **SMTP Properties** subtab. Expand the **VMS Routing** sectiManaging Dead Letteron to view the current rules.

See "Setting Up E-mail Relay Routing" on page 8-14 for details about configuring VMS routing.

Managing Oracle Beehive E-mail Components

This section contains procedures for accomplishing a variety of tasks using rules configuration with Oracle Beehive. It contains the following topics:

- Configuring Dispatch Rules
- Configuring Sent E-mail Plugins
- Configuring VMS Routing to Relay Messages
- Using the Reject All VMS
- Specifying a Local Users Domain
- Configuring VMS Evaluator to Prevent Open Relay
- Configuring SMTP to Require Authentication
- Using a Whitelist for E-mail Addresses
- Setting Up E-mail Relay Routing
- Configuring E-mail Archiving Rules
- Adding a Virus Engine to Oracle Beehive
- Managing Attachment Blocking and Virus Scanning
- Managing Dead Letter
- Setting and Modifying Multiple Endpoints
- Delivering Remote E-mails with a Local E-mail Domain
- Configuring Oracle Beehive E-mail Logs
- Viewing Email Queues

Configuring Dispatch Rules

You can use dispatch rules to configure domains (and domain IP ranges) for routing outbound messages to the outbound VMS. Domains not specified will not be routed to the outbound VMS.

Use Oracle Beekeeper to make changes to the SMTP Server's VMS Evaluator, which determines which VMS various messages should be routed to.

To configure outbound Dispatch Rules, perform the following procedure:

1. Log in to Oracle Beekeeper, select the **E-mail Service**, select the **Configuration** tab, and click **Edit**
2. Select the **SMTP Properties** tab
3. Expand the **VMS Routing** section
4. By default, there is one Host Filter Rule, which directs traffic from all hosts to the Outbound VMS. You can edit this default rule

5. Under **Add mapping rule**, select **Host Filter Rule** and click the plus button to add additional host dispatch filters. You can select any VMS to route messages to, based on the originating host(s). You can choose IP addresses or IP address ranges, or specify hosts using their fully-qualified host names. Wildcards are accepted
6. By default, there is one **Endpoint Filter Rule**, which directs traffic from all endpoints to the Outbound VMS. You can edit this default rule.
7. Under **Add mapping rule**, select **Endpoint Filter Rule** and click the plus button to add additional endpoint dispatch filters. You only need to do this if you have added additional endpoints. You can select any VMS to route messages to, depending on the originating endpoint
8. You can promote or demote rules using the up and down buttons, which change the order in which the VMS Evaluator processes the rules. The Catch-All Rule is always last
9. You can designate any VMS for the Catch-All Rule. Messages which are not matched by any previous rule will be matched by the Catch-All Rule and routed to the designated VMS
10. When you have finished making your configuration changes, click **Apply** to apply your changes to the proposed configuration without closing the configuration window, or **Save and close** to apply your changes to the proposed configuration and close the window.
11. Activate the configuration by clicking **Configuration Control** in the **System** box, and then clicking **Activate**.

Configuring Sent E-mail Plugins

By default, sent e-mails are not auditable, and are not eligible to be filed as records for Records Management. You can configure either or both of these capabilities, but only after you enable this functionality by setting a property of the Email Transport Properties component. By doing so, you cause an event to be raised whenever an e-mail is sent, and that event can be used in audit or records management policies.

To enable audit policies and records management policies to include sent e-mail messages, use the `beectl modify_property` command:

```
beectl> modify_property --component _EmailService:TransportProperties --name
SentEmailPluginEnabled --value true
```

If you want to file records of outgoing e-mail messages, you must also designate an Oracle Beehive user account as the special `RmAdminEmailId`. This user account will be used to store outgoing e-mail messages so that they can be filed as records. You can designate any account, but Oracle recommends creating an account specifically for this purpose.

Once you have created the account, get the identifier of the URM component by using the list_components command:

```
beectl> list_components --type Urm
```

Then, use the `beectl modify_property` command to add the account to the `RmAdminEmailId` property of the URM component:

```
beectl> modify_property --component <URM component ID> --name RmAdminEmailId
--value user=<userID>
```

After modifying properties, to apply the proposed configuration change, you must run the `beectl activate configuration` command:

```
beectl> activate_configuration
```

You can also modify these properties using Oracle Beekeeper.

Note: After setting the `SentEmailPluginEnabled` property, you still must create or modify audit policies and records management policies to include sent e-mails.

See Also:

- For more information about auditing sent e-mails, see: Chapter 14, "Managing Oracle Beehive Auditing"
- For more information about filing records for sent e-mails, see: "Step 2C: Enabling Record Filing of Sent E-mails" in Chapter 7, "Integrating Oracle Universal Records Management (Oracle URM) with Oracle Beehive," of the *Oracle Beehive Integration Guide*.

Configuring VMS Routing to Relay Messages

This section describes how to configure the VMS Evaluator to decide which VMS should process a given message, based on the source of the message (by its hostname or IP address) or the Endpoint the client connects to (if you have multiple Endpoints).

Configure VMS routing using Oracle Beekeeper. Log in to Oracle Beekeeper, select the **E-mail Service** from the list of services, select the **Configuration** tab, and then select the **SMTP Properties** subtab. You can view the existing configuration by expanding the **VMS Routing** section. You can make changes to the current configuration by clicking the **Edit** button.

Note: Messages internal to Oracle Beehive (from one Oracle Beehive user to another Oracle Beehive user using native clients such as Oracle Beehive Integration for Outlook) are never routed to a VMS Evaluator. They are delivered directly to recipients without requiring the E-mail Service to evaluate or route them.

Messages sent using a client that connects to Oracle Beehive using SMTP are routed through the VMS Evaluator.

You can perform the following routing tasks using Oracle Beekeeper:

- Modify an Endpoint Filter Rule to configure which VMS messages from a given Endpoint should be routed to
- Modify a Host Filter Rule to configure which VMS messages from one or more hosts or IP address ranges should be routed to
- Modify the Catchall Rule to configure which VMS messages that do not match any previous rule should be routed to
- Add additional Endpoint Filter Rules or Host Filter Rules, by selecting one or the other from the **Type** dropdown list under **Add Mapping Rule**, and then clicking the plus sign

- Change the order in which filter rules will be evaluated, by clicking the up or down arrows on any rule to promote or demote it in the order
- Remove any Endpoint Filter Rule or Host Filter Rule by clicking the red **X**

When you have finished making your configuration changes, click **Apply** to apply the changes to the proposed configuration without closing the configuration window, or **Save and close** to apply the changes to the proposed configuration and close the window.

Activate the configuration by clicking **Configuration Control** in the **System** box, and then clicking **Activate**.

Using the Reject All VMS

The Reject All VMS is a special VMS that rejects all e-mail messages sent to it. You can set any Endpoint Filter Rule, Host Filter Rule, or the Catchall Rule to route e-mail to the Reject All VMS by selecting a check box.

If you use the Reject All VMS, you must enable it as well. Unless you have enabled the Reject All VMS, checking the **Use Reject All VMS** check box has no effect (messages are not rejected).

First, enable the Reject All VMS by performing the following procedure:

1. Log in to Oracle Beekeeper, select the **E-mail Service** from the list of services, and click the **Manage** button.
2. Click **SMTP Properties** and expand the **VMSDefinition** section.
3. Check the **Use Reject All VMS** check box.
4. When you have finished making your configuration changes, click **Apply** to apply the changes to the proposed configuration without closing the configuration window, or **Save and close** to apply the changes to the proposed configuration and close the window.

   Activate the configuration by clicking **Configuration Control** in the **System** box, and then clicking **Activate**.

Then, follow the steps in "Configuring VMS Routing to Relay Messages" on page 8-6 to configure VMS routing rules. Check the **Use Reject All VMS** check box in any rule.

Specifying a Local Users Domain

The Local E-mail Patterns property is a list of domains, subdomains, or other e-mail address patterns. The users who connect to Oracle Beehive using a standards-based e-mail application, such as Eudora or Mozilla Thunderbird, should originate from a domain which you recognize as internal. If you choose not to specify local users using the Local E-mail Patterns, you will need to enter this information each time you configure the Sender Matcher or Recipient Matcher of a VMS to recognize local e-mail addresses.

The local e-mail patterns list can be used by VMSes to serve as an exception to a blocking rule, or as a basis for an accepting rule (whitelist).

By default, there is no preset list of local e-mail patterns. You can add local e-mail patterns using Oracle Beekeeper. To specify one or more local e-mail patterns, perform the following procedure:

1. Log in to Oracle Beekeeper, select the **E-mail Service**, select the **Configuration** tab, and click **Edit.**

2. Select the **General Properties** tab, and expand the **General Settings** section.
3. In the **List of valid local e-mail address patterns**, click the **Add Pattern** button to add additional patterns.
4. Click the **X** button to delete existing patterns.
5. When you have finished making your configuration changes, click **Apply** to apply the changes to the proposed configuration without closing the configuration window, or **Save and close** to apply the changes to the proposed configuration and close the window.
6. Activate the configuration by clicking **Configuration Control** in the **System** box, and then clicking **Activate**.

When you configure any VMS (such as the Inbound VMS or Outbound VMS), you can check the **Match local email patterns** check box in the Recipient Matcher or Sender Matcher, to include the list of local e-mail patterns in the rule.

Configuring VMS Evaluator to Prevent Open Relay

Note: Oracle recommends using a third-party e-mail gateway between Oracle Beehive SMTP ports and the Internet. Oracle Beehive does not provide spam filtering and other more advanced e-mail security and processing features typically recommended for production e-mail servers.

In order to prevent your Oracle Beehive deployment from acting as an open relay, you should use a configuration similar to the following:

1. The Inbound VMS should reject recipients that are not in recognized local domains.
2. The Outbound VMS should reject senders that are not allowed (either by mail address or by network location).

Note: The terms Inbound VMS and Outbound VMS are conventions used as labels for two arbitrary VMSes. In all examples in this module, the Inbound VMS is a VMS being used for inbound e-mail messages, and the Outbound VMS is a VMS being used for anything originating from Oracle Beehive. But in both cases, there is nothing special about either VMS other than its particular configuration, which you can modify. You can set up additional VMSes, and remove VMSes, according to your needs.

To prevent Open Relay, any VMS that is accepting e-mails with destinations outside your domain (such as the default sender matcher) should only accept messages from senders that are permitted (local e-mail patterns).

By default, there are no restrictions on the sender pattern matcher, so you should configure it to restrict sender matching as needed.

"Outbound VMS" is for anything originating from Oracle Beehive (either by an Oracle Beehive user through internal Oracle Beehive services, or an Oracle Beehive user through SMTP) and "Inbound VMS" is for anything originating outside of Oracle Beehive (whether sent to an Oracle Beehive user or for relay, if allowed).

There are other ways you could configure Oracle Beehive. For example, you could have multiple end-points and different ports, secured behind port-mappings behind a load balancer or router, and so forth.

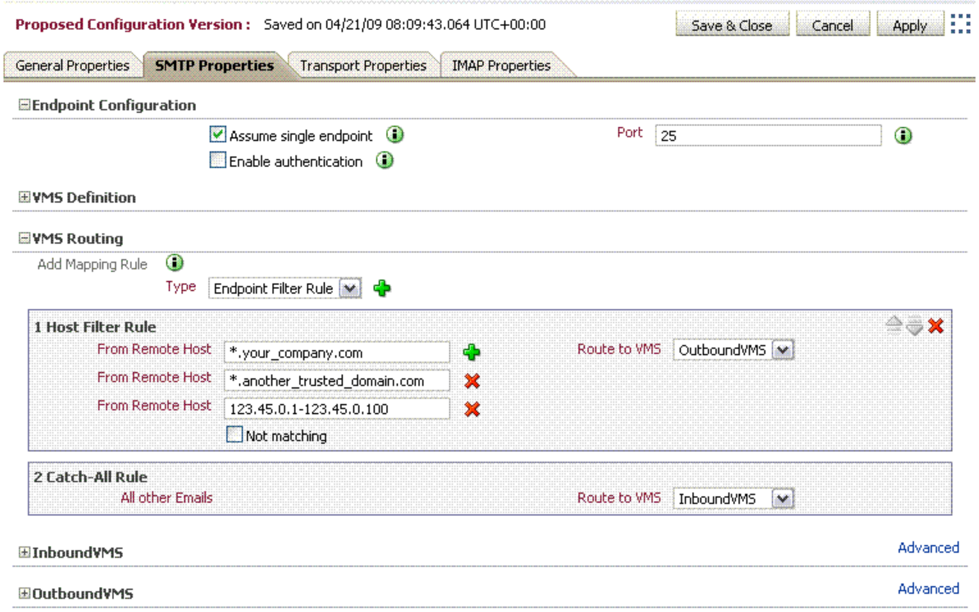
Examples of Modifying the Inbound and Outbound VMSes

One method for preventing Oracle Beehive from acting as an open relay is to set the VMS Routing and Inbound and Outbound VMS rules using the settings shown in Figure 8–1, Figure 8–2, Figure 8–3, and Figure 8–4. This example may or may not be appropriate for your own deployment.

First, in Figure 8–1, "Configuring VMS Routing Rules", the E-mail VMS Routing Rules are configured to insure that e-mail messages sent from trusted, local senders (those in your local domains and IP address ranges) are connected to the Outbound VMS (by the **Host Filter Rule**). (In this example, this will be the 'trusted' VMS that lets a sender route a message to anyone. See Figure 8–3.)

Messages sent from all other senders (those that do not match your local domains or IP address ranges) are connected to the Inbound VMS (by the **Catch-All Rule**). In this example, the Inbound VMS is configured so that messages routed through it must be addressed to a local user, so relay of messages is disallowed.

Figure 8–1 Configuring VMS Routing Rules



Proposed Configuration Version : Saved on 04/21/09 08:09:43.064 UTC+00:00

⊟Endpoint Configuration

☑ Assume single endpoint

□ Enable authentication

Port 25

⊞VMS Definition

⊟VMS Routing

Add Mapping Rule

Type Endpoint Filter Rule

1 Host Filter Rule

From Remote Host *.your_company.com

From Remote Host *.another_trusted_domain.com

From Remote Host 123.45.0.1-123.45.0.100

□ Not matching

Route to VMS OutboundVMS

2 Catch-All Rule

All other Emails

Route to VMS InboundVMS

⊞InboundVMS Advanced

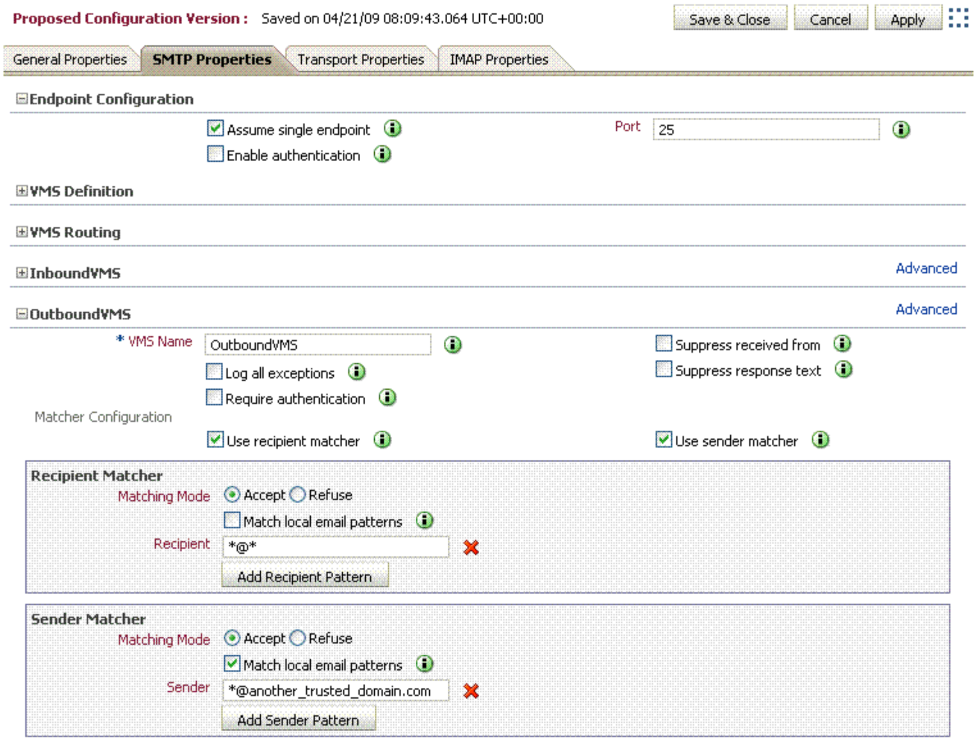
⊞OutboundVMS Advanced

In Figure 8–2, "Configuring the Outbound VMS Sender Matcher", the Outbound VMS is set to accept messages sent to any address (in the **Recipient Matcher**), and accepts messages originating from the internal addresses that you defined in your local email patterns, as well as a trusted domain `another_trusted_domain.com` (in the **Sender Matcher**). This means that connections originating from these sources are considered safe, and allowed to send messages outbound (to addresses outside of Oracle Beehive).

The **Match local e-mail patterns** check box is checked. This means that local e-mail patterns (set on the General Properties tab) will also be accepted as valid senders by the Outbound VMS.

Note: In a default Oracle Beehive install, `*@*` is used as the initial setting, which means any client connecting to the SMTP Server will be allowed to send messages, regardless of IP address or sender domain.

Figure 8–2 Configuring the Outbound VMS Sender Matcher



Proposed Configuration Version : Saved on 04/21/09 08:09:43.064 UTC+00:00

⊟Endpoint Configuration

☑ Assume single endpoint
Port: 25
☐ Enable authentication

⊞VMS Definition

⊞VMS Routing

⊞InboundVMS

⊟OutboundVMS

* VMS Name: OutboundVMS
☐ Suppress received from
☐ Log all exceptions
☐ Suppress response text
☐ Require authentication

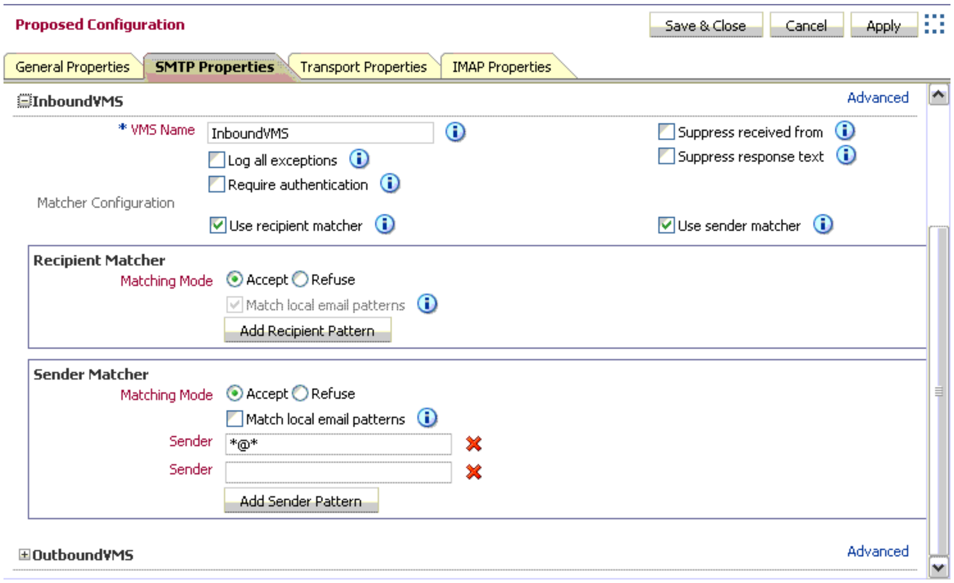
Matcher Configuration
☑ Use recipient matcher
☑ Use sender matcher

Recipient Matcher
Matching Mode: ● Accept ○ Refuse
☐ Match local email patterns
Recipient: *@*

Sender Matcher
Matching Mode: ● Accept ○ Refuse
☑ Match local email patterns
Sender: *@another_trusted_domain.com

The Inbound VMS Evaluator, as shown in Figure 8–3, "Configuring the Inbound VMS Evaluator", may be left at its default settings. In this configuration example, any external sender (as defined by the Sender Matcher) may send e-mail messages to the Oracle Beehive SMTP Server; however, only e-mail messages with a recipient that matches the local e-mail patterns (see Figure 8–4) will be delivered.

Figure 8–3 Configuring the Inbound VMS Evaluator



InboundVMS

* VMS Name: InboundVMS

□ Log all exceptions

□ Require authentication

□ Suppress received from

□ Suppress response text

Matcher Configuration

☑ Use recipient matcher

☑ Use sender matcher

Recipient Matcher

Matching Mode ● Accept ○ Refuse

☑ Match local email patterns

Sender Matcher

Matching Mode ● Accept ○ Refuse

□ Match local email patterns

Sender: *@*

Sender:

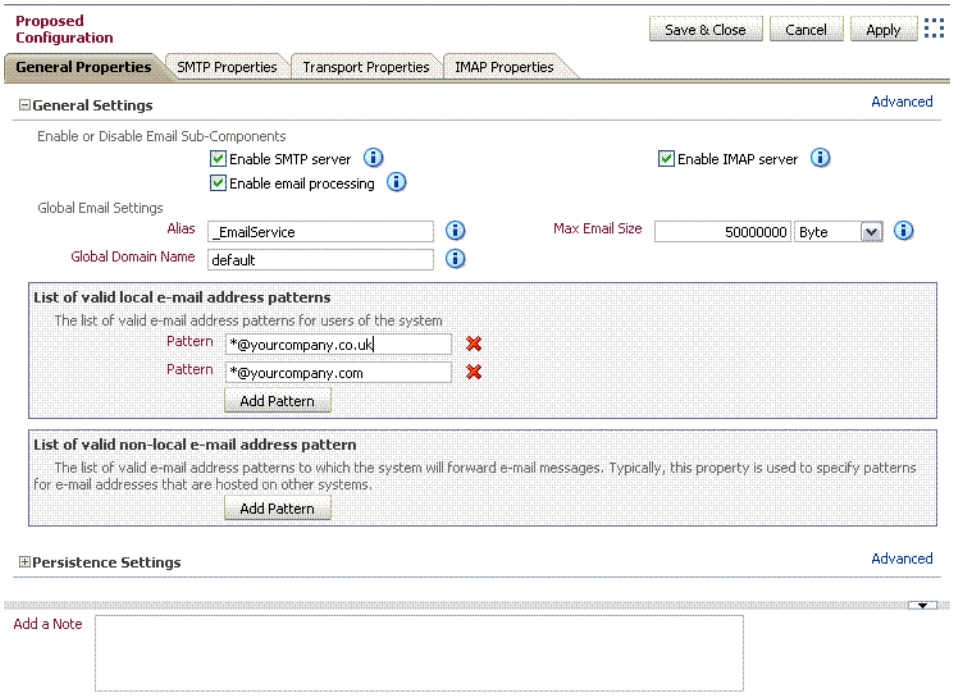
OutboundVMS

The Inbound VMS Evaluator by default uses the **Match local email patterns** setting. Local e-mail patterns are set on the General Properties tab. Unless you set a more specific local e-mail pattern, all e-mail addresses are accepted, so the Inbound VMS will accept any e-mail message: even those addressed to external addresses.

Both the Inbound and Outbound VMSes are now making use of the **Match local email patterns** setting, to determine which domains or patterns are recognized as valid internal recipients and senders of e-mail messages. Figure 8–4, "Configuring Local E-mail Patterns", shows how to set the local e-mail patterns using Oracle Beekeeper.

On the **General Settings** tab, under the **List of valid local e-mail address patterns**, click **Add Pattern** to add local e-mail pattern(s), according to your internal e-mail domain(s). Under the **List of valid non-local e-mail address patterns**, click **Add Pattern**, and add a dummy e-mail address.

Figure 8–4 Configuring Local E-mail Patterns



General Settings

Enable or Disable Email Sub-Components

☑ Enable SMTP server

☑ Enable email processing

☑ Enable IMAP server

Global Email Settings

Alias: _EmailService

Global Domain Name: default

Max Email Size: 50000000 Byte

List of valid local e-mail address patterns

The list of valid e-mail address patterns for users of the system

Pattern: *@yourcompany.co.uk

Pattern: *@yourcompany.com

List of valid non-local e-mail address pattern

The list of valid e-mail address patterns to which the system will forward e-mail messages. Typically, this property is used to specify patterns for e-mail addresses that are hosted on other systems.

Persistence Settings

Add a Note

When you have finished making your configuration changes, click **Apply** to apply the changes to the proposed configuration without closing the configuration window, or **Save and close** to apply the changes to the proposed configuration and close the window.

Activate the configuration by clicking **Configuration Control** in the **System** box, and then clicking **Activate**.

> **Note:** As further protection from unauthorized relay, consider requiring authentication from clients connecting to Oracle Beehive to send outbound messages. To set up required authentication, see "Configuring SMTP to Require Authentication" on page 8-12.

Configuring SMTP to Require Authentication

You may want to configure Oracle Beehive's SMTP endpoint to require authentication from users attempting to use Oracle Beehive for sending messages. This means that users connecting to Oracle Beehive using a standards-based client will have to provide login credentials when sending e-mail messages.

To configure the SMTP endpoint to require authentication, perform the following steps:

1. Log in to Oracle Beekeeper, select the **E-mail Service**, select the **Configuration** tab, and click **Edit**.
2. Select the **SMTP Properties** tab, and expand the section for the **Endpoint Configuration**.
3. Check the **Enable Authentication** box.

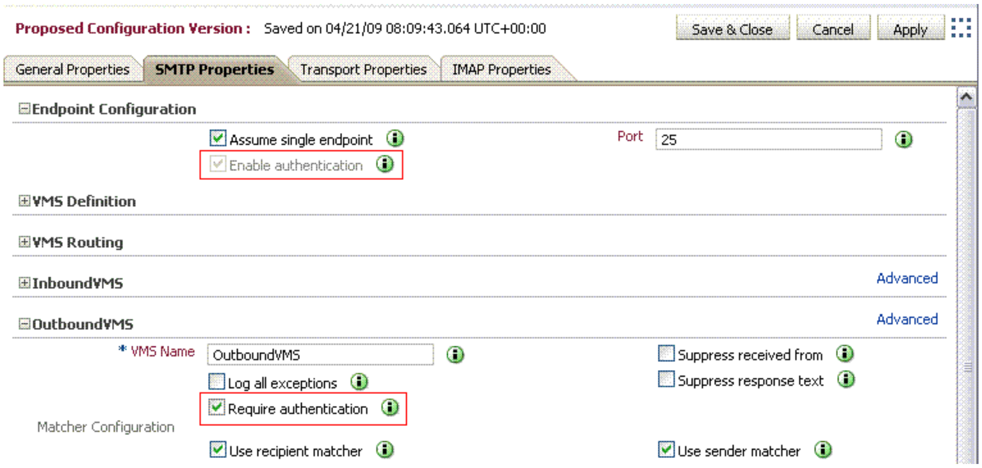
Note: This option allows, but does not require, endpoints to accept authentication.

---

4. Expand the OutboundVMS Section.
5. Check the **Require Authentication** box.

Note: This option requires authentication for connections to the OutboundVMS. It does not force other endpoints to require authentication.

Figure 8–5 Configuring OutboundVMS Authentication



Proposed Configuration Version : Saved on 04/21/09 08:09:43.064 UTC+00:00

☑ Assume single endpoint

☑ Enable authentication

Port 25

* VMS Name OutboundVMS

□ Log all exceptions

☑ Require authentication

□ Suppress received from

□ Suppress response text

☑ Use recipient matcher

☑ Use sender matcher

6. When you have finished making your configuration changes, click **Apply** to apply the changes to the proposed configuration without closing the configuration window, or **Save and close** to apply the changes to the proposed configuration and close the window.
7. Activate the configuration by clicking **Configuration Control** in the **System** box, and then clicking **Activate**.

Using a Whitelist for E-mail Addresses

For any VMS, you can specify a "whitelist". E-mails that match the specified list, which may contain wildcards, are accepted; those that do not match are blocked.

The VMS Evaluator reads the sender's e-mail address, IP, and hostame, and decides which VMS will be used to process the message. Then each VMS has a set of sender and recipient matching algorithms, which act to accept or block e-mail. For example, a typical configuration of the Outbound VMS allows only "internal" users (users with an internal e-mail address) to send e-mail out of the system, which prevents unauthorized users from relaying through the server (a form of spam control).

Each VMS has two properties, Sender Matcher and Recipient Matcher, which are checked during the SMTP transaction between client and server.

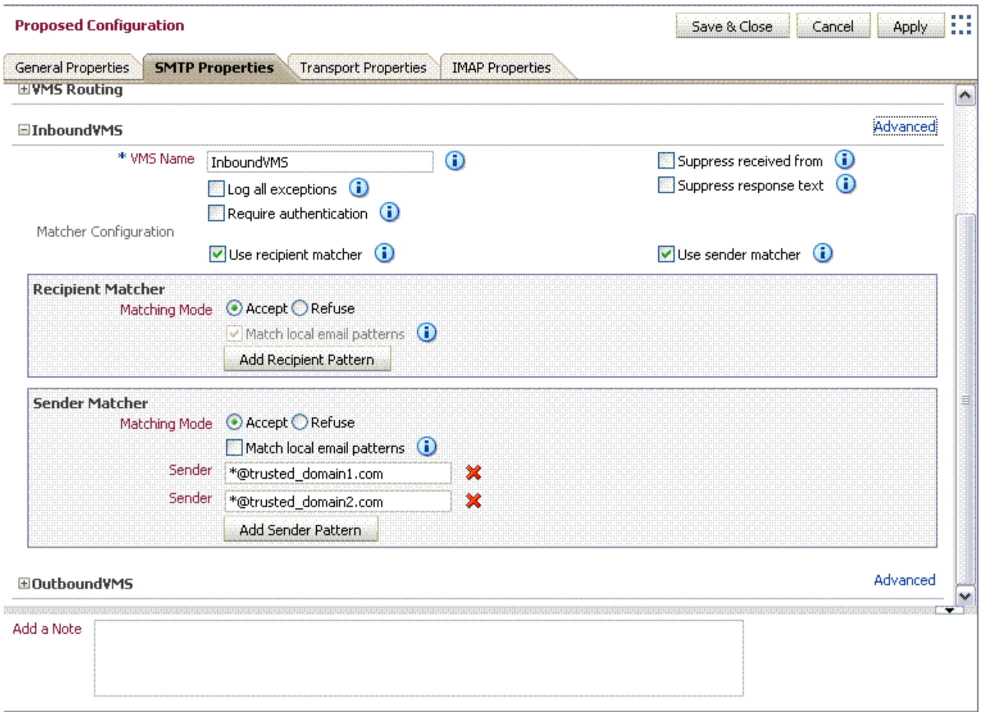
By default, the Sender Matcher of the Inbound VMS filters e-mail messages coming in to the Oracle Beehive SMTP server from an external source. In the default configuration, the Sender Matcher accepts e-mails from all senders (it is set to `*@*`

which matches any e-mail address). As long as the sender is in the format of an e-mail address (contains an @ sign), messages from any sender are accepted.

In Oracle Beekeeper, configuration options for VMSes are found by selecting the **E-mail Service**, the **Configuration** tab, and then the **SMTP Properties** subtab. To edit these properties, click the **Edit** button.

To specify a whitelist, expand the **Outbound VMS** section and edit the **Sender Matcher** settings, as shown in Figure 8–6.

Figure 8–6   Configuring an E-mail Whitelist



* VMS Name: InboundVMS

□ Log all exceptions

□ Require authentication

□ Suppress received from

□ Suppress response text

☑ Use recipient matcher

☑ Use sender matcher

Recipient Matcher — Matching Mode: ● Accept ○ Refuse; ☑ Match local email patterns

Sender Matcher — Matching Mode: ● Accept ○ Refuse; □ Match local email patterns

Sender: *@trusted_domain1.com

Sender: *@trusted_domain2.com

You can add or remove as many lines as you like.

When you have finished making your configuration changes, click **Apply** to apply the changes to the proposed configuration without closing the configuration window, or **Save and close** to apply the changes to the proposed configuration and close the window.

Activate the configuration by clicking **Configuration Control** in the **System** box, and then clicking **Activate**.

Setting Up E-mail Relay Routing

You can configure Oracle Beehive so that it acts as a relay server, accepting outbound e-mail messages from internal users, and forwarding them along to another SMTP service (such as Sendmail server, gateway, or an Internet Service Provider) for further processing. In Relay mode, you specify one or more servers to which all outgoing messages should be routed:

- If all Beehive outgoing e-mail is routed through a gateway, ISP, or some other MTA, you can use "Simple Relay Mode."

- If you need to route directly to the Internet, route to different relay servers based on a recipient's address, or route to non-standard port numbers, you'll need to use more complex delivery routing rules.

If you specify more than one server, the first server will be used and additional servers will serve as fallback servers.

Note: Messages internal to Oracle Beehive (from one Oracle Beehive user to another Oracle Beehive user using native clients such as Oracle Beehive Integration for Outlook) are never routed to a VMS Evaluator. They are delivered directly to recipients without requiring the E-mail Service to evaluate or route them.

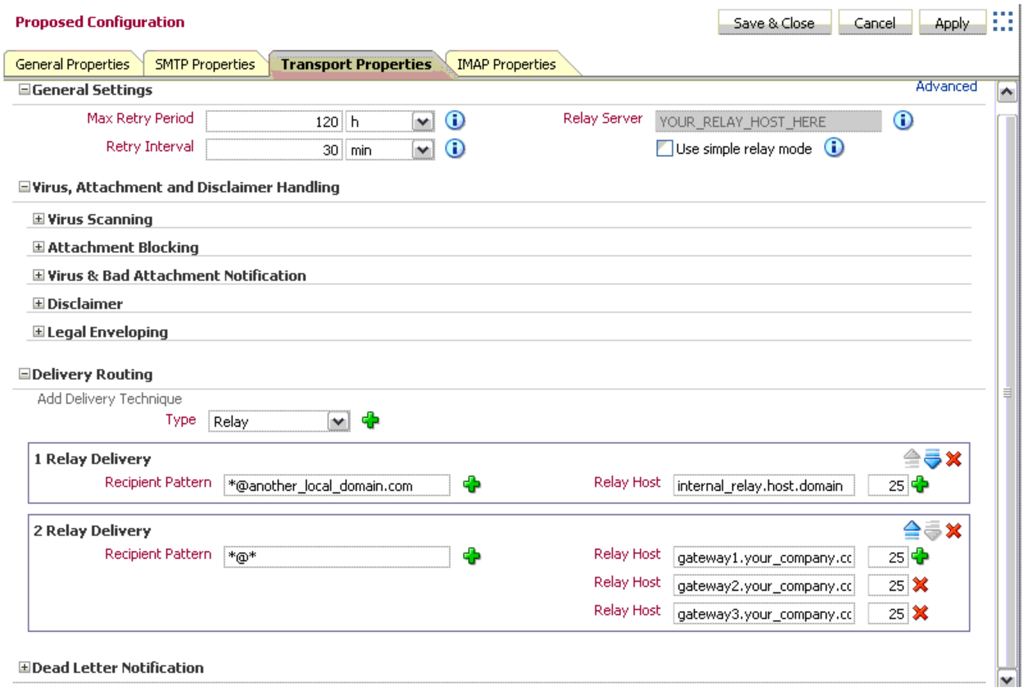
Messages sent using a client that connects to Oracle Beehive using SMTP are routed through the VMS Evaluator.

You can set up a rule that selects between different servers depending on a recipient pattern.

To set up relay servers, perform the following procedure:

1. Log in to Oracle Beekeeper, select the **E-mail Service**, select the **Configuration** tab, and click **Edit**.
2. Select the **Transport Properties** tab.
3. If you have only a single Relay server, you can specify it in the **Relay Server** field. Leave the **Use simple relay mode** check box checked, and skip to Step 6.
4. If you want to specify multiple servers, deselect the **Use simple relay mode** check box, and then expand the **Delivery Routing** section.
5. Click the **+** button next to the **Type** dropdown box to add additional Relays. Click the **+** button next to the **Recipient Pattern** field to add additional patterns to match for a given Relay. Click the **+** button next to the **Relay Host** field to specify additional fail-over hosts for a given Relay. If you have multiple Relays, you can arrange the order in which the rules will be processed by promoting or demoting the Relay order with the blue up and down buttons.

Figure 8–7 Configuring E-mail Relay Routing



General Settings

Max Retry Period: 120 h

Retry Interval: 30 min

Relay Server: YOUR_RELAY_HOST_HERE

□ Use simple relay mode

Virus, Attachment and Disclaimer Handling

- Virus Scanning
- Attachment Blocking
- Virus & Bad Attachment Notification
- Disclaimer
- Legal Enveloping

Delivery Routing

Add Delivery Technique

Type: Relay

1 Relay Delivery

Recipient Pattern: *@another_local_domain.com

Relay Host: internal_relay.host.domain 25

2 Relay Delivery

Recipient Pattern: *@*

Relay Host: gateway1.your_company.cc 25

Relay Host: gateway2.your_company.cc 25

Relay Host: gateway3.your_company.cc 25

Dead Letter Notification

Note: In addition to the Relay type, you can also specify a **Delivery Technique** of type DNS-MX. DNS-MX is intended for the use of a DMZ or Internet-based deployment, in which Oracle Beehive will attempt to look up DNS and direct delivery of messages to the corresponding address directly. This configuration is inappropriate for most deployments of Oracle Beehive.

---

6. When you have finished making your configuration changes, click **Apply** to apply the changes to the proposed configuration without closing the configuration window, or **Save and close** to apply the changes to the proposed configuration and close the window.
7. Activate the configuration by clicking **Configuration Control** in the **System** box, and then clicking **Activate**.

Configuring E-mail Archiving Rules

You can set up rules so that e-mail sent from or to specified addresses will be sent to one or more special archiving addresses. With archiving, Oracle Beehive attaches the original message to a new e-mail, includes meta-data about the original message, and sends it to one or more additional destinations for compliance or archiving purposes. For archiving email messages, you can use any third-party archiving server which uses RFC standards and has the capacity to store all your messages.

Archiving rules are a simple way to retain sent and received e-mail. This makes integration with third-party archiving products and services relatively simple. Alternatively, you can use any e-mail client to access and archive forwarded messages all at once.

Configure e-mail archiving rules by performing the following steps:

1. Log in to Oracle Beekeeper, select the **E-mail Service**, select the **Configuration** tab, and click **Edit**.
2. Select the **Transport Properties** tab.
3. Expand the **Legal Enveloping** section.
4. Choose between the two options, **Single Destination** or **Append to Email**. To choose **Append to Email**, de-select the **Use single destination** box:
   - With Single Destination, all archive messages are sent to the same address, such as `SoxCompliance@archive.example.com`.
   - With Append to e-mail, you define an append string, and the system will send a copy of each message that matches the pattern to the <recipient>_ appendstring. For example, if your append string is _archive, and a matched recipient is `user.name@example.com`, the system will send a copy of the message to `user.name_archive@example.com`.
5. Choose the e-mail address pattern or patterns you want to generate an archive message; for example, a specific e-mail address (`user@example.com`) or a pattern (`*@sales.example.com`). The system will generate an archive message for any sender or recipient that matches this pattern. Click the **plus** icon to add additional patterns (an address will generate an archive message if it matches any pattern)

   > **Caution:** Make sure that the Single Destination address, or the modified addresses produced by the Append process, do not result in addresses that match the e-mail address pattern you specify. If any such messages match, a logical loop will be created as the message is repeatedly appended or forwarded.

6. If the destination address or addresses you specify using Single Destination or Append to e-mail is a third-party archive system, make sure that the general routing rules route the messages to the correct destination. For example, you may need to configure DeliveryRules to ensure proper routing.

   > **Caution:** You can configure the DeliveryRules to archive a message after the delivery of a message, or to archive it after a post resolution is done. If the DeliveryRules are set to archive Post Delivery, the Oracle Beehive server will archive the messages when they are delivered to the users Inboxes. If a message does not reach the Inbox, it will not be archived. If the DeliveryRules are set to archive Post Resolution, the message is sent to the archiving system after the Oracle Beehive server has verified that it is a valid and local address.
   >
   > You can set the DeliveryRules in either the Post Resolution rules or the Post Delivery Rules, but **NOT** in both. Setting DeliveryRules in both can cause duplicates and even errors on the archiver or MTA used to archive the messages.

   > **Note:** The email messages that are archived depend on the filter you use. If you set it to a wildcard (*), all email messages will be archived. DSNs are also archived as they are considered to be messages.

7. When you have finished making your configuration changes, click **Apply** to apply the changes to the proposed configuration without closing the configuration window, or **Save and close** to apply the changes to the proposed configuration and close the window.
8. Activate the configuration by clicking **Configuration Control** in the **System** box, and then clicking **Activate**.

Adding a Virus Engine to Oracle Beehive

Oracle Beehive allows you to specify one or more virus scan engines for use in scanning e-mail messages.

For instructions on adding a virus engine to Oracle Beehive, see Chapter 11, "Integrating Symantec Scan Engine with Oracle Beehive" in the *Oracle Beehive Integration Guide*.

Managing Attachment Blocking and Virus Scanning

After you have added a virus scanning engine, you can enable virus scanning using Oracle Beekeeper.

For instructions on managing attachment blocking and virus scanning, see Chapter 11, "Integrating Symantec Scan Engine with Oracle Beehive" in the *Oracle Beehive Integration Guide*.

Managing Dead Letter

"Dead Letter" is the notification returned for undeliverable messages. Use the following procedure to set up and enable Dead Letter in Oracle Beehive:

1. Log in to Oracle Beekeeper, select the **E-mail Service**, select the **Configuration** tab, and click **Edit**.
2. Select the **Transport Properties** tab.
3. Expand the **Dead Letter Notification** section.
4. You can configure whether to send a notification to local senders, external senders, and administrators, and you can customize the notification message that will be sent. To send notification to an administrator, set the **Notifier Email** attribute to the e-mail address of an administrator, and check the **Notify administrators** check box.

> **Note:** By default, local and remote senders, but not administrators, are notified of undeliverable messages. If you deselect all three boxes, this effectively disables dead letter functionality.

5. When you have finished making your configuration changes, click **Apply** to apply the changes to the proposed configuration without closing the configuration window, or **Save and close** to apply the changes to the proposed configuration and close the window.
6. Activate the configuration by clicking **Configuration Control** in the **System** box, and then clicking **Activate**.

Setting and Modifying Multiple Endpoints

You can define multiple endpoints for an Oracle Beehive SMTP server. An endpoint scheme contains a specification of protocol, a string or wildcard used as a filter, and a port. For example, `MX:*:2226`.

> **Caution:** When creating or modifying endpoints, take care that you do not create a port conflict. On UNIX and Linux systems, if you make use of a privileged port, you must perform additional configuration steps: See "Modifying Oracle Beehive Ports using Privileged Port Numbers" in Chapter 4, "Oracle Beehive Property Reference" of the *Oracle Beehive Administrator's Reference Guide*.

Supported protocols are MX (unencrypted) and MXS (MX using SSL):

- `MX`: The server will use the BTI layer to request an MX server listen for TCP connections on the specified port.
- `MXS`: Similar to MX, except that the BTI layer will pre-negotiate an SSL session before any incoming connection requests are presented to the application (SMTP/IMAP) layer.

The filter is not currently used; you must use `*` for this value.

Once you have added additional endpoints, when you configure VMS Routing rules, you can add additional Endpoint Filter Rules which route specific endpoints to an identified VMS. See "Configuring VMS Routing to Relay Messages" on page 8-6 for details.

Perform the following steps to modify the default endpoint, and define additional endpoints:

1. Log in to Oracle Beekeeper, select the **E-mail Service**, select the **Configuration** tab, and click **Edit**.
2. Select the **SMTP Properties** tab.
3. Expand the **Endpoint Configuration** section.
4. If the **Assume single endpoint** check box is checked, a single port field is listed, which is the configured SMTP server port. All SMTP traffic must be directed to this port. Uncheck the **Assume single endpoint** check box to configure multiple endpoints.
5. If you want, you can modify the default endpoint name and scheme.
6. Create one or more new endpoints using the following steps:
   a. Click the **Add Endpoint** button.
   b. Enter a name and scheme for the additional endpoint.
7. When you have finished making your configuration changes, click **Apply** to apply the changes to the proposed configuration without closing the configuration window, or **Save and close** to apply the changes to the proposed configuration and close the window.
8. Activate the configuration by clicking **Configuration Control** in the **System** box, and then clicking **Activate**.

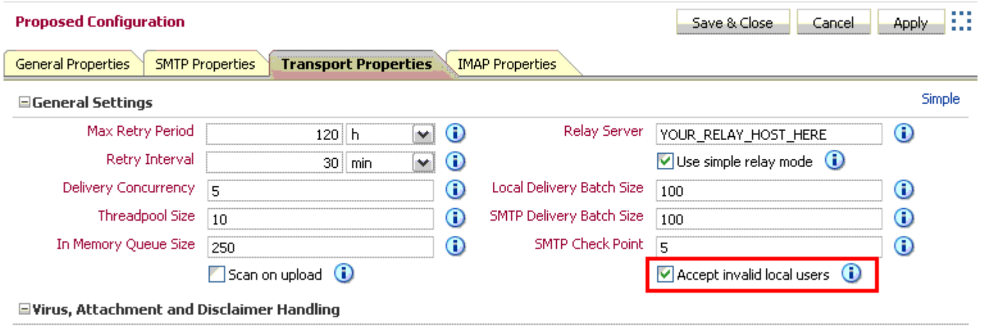
Delivering Remote E-mails with a Local E-mail Domain

In some cases, you may want to configure Oracle Beehive to accept inbound e-mails sent to addresses within an acceptable domain, but of users whose address is not within Oracle Beehive. For example, if your domain is example.com, you may wish to accept (and subsequently relay) a message for User1@example.com, even though there is no User1 in Oracle Beehive.

By default such "invalid local users" are blocked. To enable acceptance of such addresses:

1. Log in to Oracle Beekeeper, select the **E-mail Service**, select the **Configuration** tab, and click **Edit**.
2. Select the **Transport Properties** tab.
3. Click the **Advanced** link to show advanced properties.
4. Check the **Accept invalid local users** check box, as shown in Figure 8–8.

Figure 8–8 Selecting Accept Invalid Local Users Check box



Proposed Configuration

⊟General Settings

Max Retry Period: 120 h

Retry Interval: 30 min

Delivery Concurrency: 5

Threadpool Size: 10

In Memory Queue Size: 250

□ Scan on upload

Relay Server: YOUR_RELAY_HOST_HERE

☑ Use simple relay mode

Local Delivery Batch Size: 100

SMTP Delivery Batch Size: 100

SMTP Check Point: 5

☑ Accept invalid local users

⊟Virus, Attachment and Disclaimer Handling

5. When you have finished making your configuration changes, click **Apply** to apply the changes to the proposed configuration without closing the configuration window, or **Save and close** to apply the changes to the proposed configuration and close the window.
6. Activate the configuration by clicking **Configuration Control** in the **System** box, and then clicking **Activate**.

To configure further relaying of such addresses, follow the instructions in "Setting Up E-mail Relay Routing" on page 8-14.

Configuring Oracle Beehive E-mail Logs

To enable logs for e-mail messages, use the `beectl modify_property` command for `ModuleLogLevel`. Set the log level to `INFO` for the `mail.service.transport` module and activate the configuration as follows:

```
beectl modify_property --component _CURRENT_SITE:LoggingProperties --name
ModuleLogLevel --value oracle.ocs.mail.service.transport:INFO --activate_
configuration
```

Filter the e-mail log messages as the `INFO` level consists of several other logs. The log is available at:

```
$ORACLE_HOME/beehive/logs/oc4j/<your BEEAPP container>/log
```

For more information on modifying the log levels, see Chapter 17, "Oracle Beehive Logging and Diagnosability".

Viewing Email Queues

A service instance is physically a OC4J container in the middle tier that deploys an email service application. The following are the different types of email queues:

- **Active queue** – Each service instance has an active queue(s) of messages waiting to be processed.
- **Overflow queue** – When the active queue grows over the tunable system limit, new messages received by that service instance are spilled into another global queue called the overflow queue.
- **Retry queue** – A retry queue is a global queue polled by all service instances.

Email messages in transit are messages that has not been fully delivered. These messages are associated with exactly one of the queues above. To display all email messages in transit and other vital information related to message delivery email queues, use the following `beectl` commands:

- `list_email_queues` – This command displays the different email queues: active queue(s), retry queue, and overflow queue.
- `list_email_messages --queue_id <queue_id> --max_count <max_count>` – This command displays all the email messages for a specified queue id.
- `list_recipients_status --internet_message_id` – This command displays the recipent status for a specified message id.

For more information about the `beectl` commands, see the *Oracle Beehive Administrator's Reference Guide*.

E-mail Coexistence in a Single Domain

You may be installing Oracle Beehive in an environment with some e-mail users using pre-existing e-mail domains. You can maintain a common e-mail domain for all users, allowing new Oracle Beehive users to use the same domain, without creating a conflict between the multiple, coexisting e-mail servers.

Example

If a pre-existing user's e-mail address is user1@example.com, and Oracle Beehive is installed on a server beehive.example.com, then configure the co-existing e-mail system to auto-forward user1's e-mail to user1@beehive.example.com. Then, add an additional e-mail address for user1 in Oracle Beehive as user1@beehive.example.com, and add `*@beehive.example.com` as a LocalEmailPattern.

> **Note:** This relies on the co-existing system's e-mail forwards being envelope forwards; e-mails will be forwarded and accepted as user1@beehive.example.com, but the message headers will still be user1@example.com, and visible that way to the Oracle Beehive user.

To allow this type of coexistence, perform the following steps:

1. Log in to Oracle Beekeeper, select the **E-mail Service**, select the **Configuration** tab, and click **Edit**.

2. Select the **Transport Properties** tab.
3. Click the **Advanced** link to show advanced properties.
4. Check the **Accept invalid local users** check box, as shown in Figure 8–8, "Selecting Accept Invalid Local Users Check box" on page 8-20. By checking this box, you will allow all e-mails sent to a local address, even if the user or specific e-mail address does not exist in Oracle Beehive.
5. Select the **General Properties** tab, and expand the **General Settings** section.
6. In the **List of valid local e-mail address patterns**, click the **Add Pattern** button to add patterns. Add the e-mail domain you want to use with Oracle Beehive.
7. When you have finished making your configuration changes, click **Apply** to apply the changes to the proposed configuration without closing the configuration window, or **Save and close** to apply the changes to the proposed configuration and close the window.
8. Activate the configuration by clicking **Configuration Control** in the **System** box, and then clicking **Activate**.
9. On the non-Oracle Beehive e-mail system, configure user accounts of Oracle Beehive users to automatically forward to the Oracle Beehive SMTP Server. For each Oracle Beehive user, add the additional e-mail address.

9

Managing Oracle Beehive Subscriptions and Notifications

This module describes how to manage subscriptions and notifications in Oracle Beehive. Read this module if you are an Oracle Beehive System Administrator, an Oracle Beehive Business Administrator, or if you are responsible for creating or managing Oracle Beehive subscriptions and notifications. This module includes the following sections:

- Configuring User Notifications
- Creating Server-Side Rules with User Subscriptions
- Configuring Notifications to use SMS
- Configuring Actionable Notifications

> **See Also:** For information on configuring the Notification Delivery Service, see "Notification Delivery Service" in Chapter 4, "Oracle Beehive Property Reference", of the *Oracle Beehive Administrator's Reference Guide*

Configuring User Notifications

By default, each newly-provisioned Oracle Beehive user is subscribed to receive notifications about the following events:

- The user is invited to an event.
- An event the user is invited to, is deleted.
- The user is assigned a task.
- A task assignment is withdrawn.

Users can set up new notifications and disable default notifications provided to them, and they can control where the notifications are delivered.

Users make changes to their notification preferences using the Oracle Beehive user preference pages, which can be accessed using Oracle Beehive Central: `http(s)://<beehive host>:<port>/bcentral`.

Users can enable or disable notifications, and can choose a particular delivery channel they would like notification sent to for each notification type. By default, notifications are sent to the e-mail inbox. Users can temporarily disable notifications to a particular delivery channel. For example, users could turn off SMS delivery if they are going to be unavailable for a day, and then re-enable it afterwards.

You can alter what a user is subscribed to after they are provisioned. The easiest way to do this is by using Oracle Beekeeper. You can also manage user subscriptions using `beectl` commands.

Managing User Subscriptions using `beectl`

You can add a user subscription by using the `beectl add_user_subscription` command. You must specify the type of entity subscription, a subscription rule, the container the subscription is being attached (applied) to, and the user:

```
beectl> add_user_subscription --source_entity_class <sourceEntityClass of
template> --rule <rule name> --attach <identifier of attached entity> --subscriber
<identifier of subscriber>
```

The following example shows how to subscribe a user to be notified whenever a new document is uploaded to a workspace:

```
beectl> add_user_subscription --source_entity_class Document --rule NOTIFY_ON_ANY_
NEW_DOCUMENT --attach <Workspace identifier> --subscriber <User identifier>

Subscription has been created successfully. Identifier of the newly created
subscription is: 05C1:7403:subs:D493EDCBB1B34A06B680C37A30288E8B000000000000
```

Note: Make a note of the subscription identifier. You may need it if you decide to modify the subscription in the future.

Table 9–1, " User Subscription Entities and Rules" lists the entities that can be subscribed to and the subscription rules available for each entity.

Table 9–1 User Subscription Entities and Rules

Entity	Rule
AddressBook	NOTIFY_ON_ANY_UPDATED_CONTACT NOTIFY_ON_ANY_DELETED_CONTACT NOTIFY_ON_ANY_NEW_CONTACT
Artifact	NOTIFY_ON_ANY_NEW_DISCUSSION_ARTIFACT
Assignment	NOTIFY_ON_ANY_UPDATED_ASSIGNMENT NOTIFY_ON_ANY_NEW_ASSIGNMENT NOTIFY_ON_ANY_DELETED_ASSIGNMENT NOTIFY_ON_ANY_COMPLETED_ASSIGNMENT
DiscussionsMessage	NOTIFY_ON_ANY_NEW_DISCUSSION_MESSAGE NOTIFY_ON_ANY_DELETED_DISCUSSION_MESSAGE NOTIFY_ON_ANY_UPDATED_DISCUSSION_MESSAGE
Document	NOTIFY_ON_ANY_CREATED_CONTENT NOTIFY_ON_ANY_MODIFIED_CONTENT NOTIFY_ON_ANY_DELETED_CONTENT
Forum	NOTIFY_ON_ANY_NEW_DISCUSSION_FORUM NOTIFY_ON_ANY_DELETED_DISCUSSION_FORUM NOTIFY_ON_ANY_RENAMED_DISCUSSION_FORUM NOTIFY_ON_ANY_UPDATED_DISCUSSION_FORUM

Table 9–1 (Cont.) User Subscription Entities and Rules

Entity	Rule
Invitation	NOTIFY_ON_ANY_NEW_INVITATION NOTIFY_ON_ANY_UPDATED_INVITATION NOTIFY_ON_ANY_DELETED_INVITATION
Resource	NOTIFY_ON_ANY_INVITATION_CREATED_FOR_RESOURCE_APPROVAL NOTIFY_ON_ANY_INVITATION_UPDATED_FOR_RESOURCE_APPROVAL
Topic	NOTIFY_ON_ANY_NEW_DISCUSSION_TOPIC NOTIFY_ON_ANY_UPDATED_DISCUSSION_TOPIC NOTIFY_ON_ANY_DELETED_DISCUSSION_TOPIC NOTIFY_ON_ANY_MOVED_DISCUSSION_TOPIC
Workspace	NOTIFY_ON_ANY_MEMBER_ADDED NOTIFY_ON_ANY_MEMBER_REMOVED NOTIFY_ON_WORKSPACE_LOCKED NOTIFY_ON_WORKSPACE_UNLOCKED

You can enable or disable an existing user subscription using the `beectl modify_user_subscription` command:

```
beectl> modify_user_subscription --enable {true|false} --subscription
<Subscription identifier>
```

Creating Server-Side Rules with User Subscriptions

In addition to notifications, you can create Server-Side Rules (SSRs) using RULE_STATEMENT syntax with the `beectl add_user_subscription` command.

Example 9–1 shows a SSR subscription with multiple conditions and multiple actions:

Example 9–1 Server-Side Rules in User Subscription

```
beectl> add_user_subscription --source_entity_class Message --attach
5457:7954:wspr:5B5DB5E3F6652295E040578C92165D03000000000456 --subscriber
user=user1 --name MyServerSideRule_1 --rule RULE_STATEMENT:CONDITION:created_any_
from_contains=beehive,created_any_subject_contains=build,ACTION:move_
to=5457:7954:afrh:5B5DB5E3F6652295E040578C92165D03000000000440,forward_
to=user1@example.com
```

Note: This example uses CollabIDs for the workspace specified with the `--attach` option and the folder specified with the `move_to` action; however, you can use regular BODN identifiers for these objects as well.

Example 9–1 is a rule that invokes the following actions, if the following conditions are met:

Conditions:

- The From: field contains the string 'beehive'
- AND, the Subject: field contains the string 'build'

Actions:

- Move the new message to a folder
- AND, forward the message to the address 'user1@example.com'

Table 9–2 presents the available conditions and attributes you can use when creating e-mail SSRs.

Table 9–2 E-mail Server Side Rule Condition Attributes

Condition Name	Attribute Name
CONTAINER_MESSAGE_CREATED	created_any
CONTAINER_MESSAGE_CREATED_SUBJECT_IS	created_any_subject_is
CONTAINER_MESSAGE_CREATED_SUBJECT_CONTAINS	created_any_subject_contains
CONTAINER_MESSAGE_CREATED_SUBJECT_IS_NOT	created_any_subject_is_not
CONTAINER_MESSAGE_CREATED_SUBJECT_CONTAINS_NOT	created_any_subject_contains_not
CONTAINER_MESSAGE_CREATED_FROM_IS	created_any_from_is
CONTAINER_MESSAGE_CREATED_FROM_CONTAINS	created_any_from_contains
CONTAINER_MESSAGE_CREATED_FROM_IS_NOT	created_any_from_is_not
CONTAINER_MESSAGE_CREATED_FROM_CONTAINS_NOT	created_any_from_contains_not
CONTAINER_MESSAGE_CREATED_TO_IS	created_any_to_is
CONTAINER_MESSAGE_CREATED_TO_CONTAINS	created_any_to_contains
CONTAINER_MESSAGE_CREATED_TO_IS_NOT	created_any_to_is_not
CONTAINER_MESSAGE_CREATED_TO_CONTAINS_NOT	created_any_to_contains_not
CONTAINER_MESSAGE_CREATED_CC_IS	created_any_cc_is
CONTAINER_MESSAGE_CREATED_CC_CONTAINS	created_any_cc_contains
CONTAINER_MESSAGE_CREATED_CC_IS_NOT	created_any_cc_is_not
CONTAINER_MESSAGE_CREATED_CC_CONTAINS_NOT	created_any_cc_contains_not
CONTAINER_MESSAGE_CREATED_PRIORITY_IS	created_any_priority_is
CONTAINER_MESSAGE_CREATED_PRIORITY_IS_NOT	created_any_priority_is_not
CONTAINER_MESSAGE_CREATED_TO_OR_CC_CONTAINS	created_any_to_or_cc_contains
CONTAINER_MESSAGE_CREATED_TO_OR_CC_IS	created_any_to_or_cc_is
CONTAINER_MESSAGE_CREATED_TO_OR_CC_CONTAINS_NOT	created_any_to_or_cc_contains_not
CONTAINER_MESSAGE_CREATED_TO_OR_CC_IS_NOT	created_any_to_or_cc_is_not

Table 9–3 presents the action names and attributes you can use when creating e-mail SSRs.

Table 9–3 E-mail Server Side Rule Action Attributes

Action Name	Attribute Name
NOTIFY	channel
MOVE	move_to
FORWARD	forward_to
BCC	bcc_to
SET_FLAGS	flag
DELETE	delete

You can create rules with multiple values (such as multiple destination addresses for a forwarded e-mail message) by specifying the attribute name-value pair multiple times. For example:

```
beectl> add_user_subscription --source_entity_class Message --attach
5457:7954:wspr:5B5DB5E3F6652295E040578C92165D03000000000456 --subscriber
user=user1 --name MyServerSideRule_1 --rule RULE_STATEMENT:CONDITION:created_any_
from_contains=beehive,created_any_subject_contains=build,ACTION:move_
to=5457:7954:afrh:5B5DB5E3F6652295E040578C92165D03000000000440,forward_
to=user2@example.com,forward_to=user3@example.com
```

This example is a rule that invokes the following actions, if the following conditions are met:

Conditions:

- The From: field contains the string 'beehive'
- AND, the Subject: field contains the string 'build'

Actions:

- Move the new message to a folder (identified by the folder's CollabID)
- AND, forward the message to the two addresses 'user2@example.com' and 'user3@example.com'

Configuring Notifications to use SMS

Oracle Beehive can send users notifications as email messages, instant messages, or as mobile messages in the form of SMS. SMS notifications are only possible if the Oracle Beehive SMS delivery channel is enabled and configured to point to an SMS Aggregator.

> **Note:** Some mobile operators will not deliver messages from a short code they do not recognize. You may need to work with your SMS aggregator to register your short code with such operators.

You can configure SMS notifications in any one of the following ways:

- Configuring SMS using SMPP
- Configuring SMS using XMS

See Also: For information on configuring the SMPP Delivery Service, see "Managing the SMPP Delivery Service" on page 5-20

Configuring SMS using SMPP

The SMS delivery channel is configured by default to use Short Message Peer-to-Peer (SMPP). It will not be able to deliver SMS notifications until a subscription from an SMPP-compliant aggregator is secured.

See Also: For a list of supported SMPP-compliant aggregators, see "Oracle Beehive Mobility Data Sheet" on the Oracle Technology Network website, at the following URL:

http://www.oracle.com/technology/products/beehive/pdf/oracle%20beehive%20mobility%20data%20sheet.pdf

Once you have received an SMS subscription from one of these two vendors, you must make the SMS delivery channel aware of your credentials and enable the delivery channel.

To configure SMS with SMPP, perform the following steps:

1. Set the Notification Delivery Service to SMPP mode:

   ```
   beectl> modify_property --component _NotificationDeliveryService:SmsSettings
   --name SmsMode --value SMPP
   ```

2. Update the Notification Delivery Service with the URL, system ID, and password for your SMS subscription:

   ```
   beectl> modify_property --component _SmppDeliveryService --name SmppServiceUrl
   --value <service url>

   beectl> modify_property --component _SmppDeliveryService --name SmppSystemID
   --value <system id>

   beectl> modify_secure_property --component _SmppDeliveryService --name
   SmppPassword -value <password>
   ```

3. If you are using Clickatell Gateway, you must also set the `SmppSystemType`. The value is your assigned ID for the SMPP configuration with Clickatell via the Web interface:

   ```
   beectl> modify_secure_property --component _SmppDeliveryService --name
   SmppSystemType -value <assigned ID>
   ```

4. Activate your proposed configuration using the `beectl activate_configuration` command:

   ```
   beectl> activate_configuration
   ```

Configuring SMS using XMS

You can configure SMS delivery with the Oracle iAS Wireless XMS Interface by using the Oracle Beehive SMS delivery channel's XMS mode.

To configure SMS with XMS, perform the following steps:

1. Set the Notification Delivery Service SMS mode to XMS:

   ```
   beectl> modify_property --component _NotificationDeliveryService:SmsSettings
   ```

```
--name SmsMode --value XMS
```

2. Update the Notification Delivery Service with the user name, password, proxy host name, and proxy port for your XMS interface by issuing the following `beectl` commands:

```
beectl> modify_property --component _NotificationDeliveryService:XmsSettings
--name XmsUserName --value <username>

beectl> modify_secure_property --component _
NotificationDeliveryService:XmsSettings --name XmsPassword --value <password>

beectl> modify_property --component _
NotificationDeliveryService:XmsSettings:WebProxySettings --name ProxyHostName
--value <host>

beectl> modify_property --component _
NotificationDeliveryService:XmsSettings:WebProxySettings --name ProxyPort
--value <port>
```

3. Activate your proposed configuration using the `beectl activate_configuration` command:

```
beectl> activate_configuration
```

Configuring Actionable Notifications

An actionable notification is an Oracle Beehive feature that allows users to accept or decline an invitation or assignment by replying to notifications they receive.

By default, however, notifications are one-way (from the server to the user). In order to enable actionable notifications, you must perform several post installation configuration steps.

To enable actionable notifications, perform the following procedure:

1. Create a user account that will be used to receive notification responses from users. For example:

```
beectl> add_user --given_name NotificationAdmin --family_name NotificationAdmin
--login_id NotificationAdmin --login_password <password> --address business_
1:mailto:NotificationAdmin@example.com --address business_
1:im:NotificationAdmin@example.com --scope <enterprise identifier>
```

In this example, the user is called NotificationAdmin, but you can use any name. You can also create this account using Oracle Beekeeper.

> **Note:** If you are synchronizing the User Directory with an external, LDAP-based directory, you must create this account in your external directory with e-mail and instant message addresses, just like creating any other user account. Make sure the account is subsequently synchronized to the Oracle Beehive User Directory.

2. Use the `beectl list_users` command to get the EID of the user you just created. For example:

```
beectl> list_users --user loginid=NotificationAdmin --entity_format id
```

Make a note of the EID included in the output from this command.

3. Set a property of the Notification Delivery Service using the EID of the new user account you just created, with the `beectl modify_property` command:

```
beectl> modify_property --component _
NotificationDeliveryService:ActionAskSetings --name ActionableNotificationUser
--value <eid of user>
```

4. Activate your proposed configuration using the `beectl activate_configuration` command:

```
beectl> activate_configuration
```

The Notification Delivery Service will automatically start sending actionable notifications.

Actionable Notifications with SMS

In order to allow replies to SMS Notifications, a two-way SMS connection is required. Follow the steps for enabling the SMS delivery channel found in "Configuring Notifications to use SMS" on page 9-5. In order to support two-way SMS you must use SMPP mode. All that is required is for a proper short code to be assigned as part of the `NotificationDeliveryService.SmsSettings` configuration object.

> **Note:** When Actionable Notifications with SMS are enabled, users can interact with the "ASK Service" interface, which allows them to send SMS commands to the Notification Delivery Service and retrieve a variety of e-mail, calendar, and contacts data.
>
> For more information about the ASK Service, see "Using ASK Commands" in "Configuring Mobile Devices Help," available on the Oracle Technology Network Website at the following URL:
>
> `http://www.oracle.com/technology/products/beehive/beehive_users/2_0/mobile.htm#BABDCJGH`

Disabling Actionable Notification Responses

By default, when a user responds to an actionable notification, the system will reply with a confirmation message. You can set the `ActionableNotificationReplyDisabled` property of the `NotificationDeliveryService.ActionAskSettings` subcomponent to `true` to disable actionable notification responses. Set this property back to `false` to re-enable actionable notification responses.

> **See Also:** "NotificationDeliveryService.ActionAskSettings" in Chapter 4, "Oracle Beehive Property Reference," of the *Oracle Beehive Administrator's Reference Guide*

10

Managing Oracle Beehive Time Management

This module includes information about various administration tasks relating to Oracle Beehive Time Management services.

It includes the following topics:

Notes:

- All example `beectl` commands in this module are shown using the `beectl` shell mode. For more information about `beectl` shell mode and how to invoke it, refer to Chapter 2, "Oracle Beehive Command-Line Utility" in the *Oracle Beehive Administrator's Reference Guide*.
- You can also perform some of the configuration tasks presented in this chapter using Oracle Beekeeper. See the *Oracle Beekeeper Online Help* for details.

Managing Holidays

Oracle Beehive provides a special type of calendar event called a holiday. Holidays are day events which are optimal for informing the user community of enterprise-wide events, such as public holidays not worked. Typically, an administrator creates a holiday calendar in a team workspace, broadcasts its availability to users, and users can then control whether or not the event will appear in their personal calendars.

Note: Holiday events do not expose the attendee list to users who view the event.

You can use the `beectl import_icalendar` command to import holiday events to a team workspace. You can import one or multiple holidays from one iCalendar file. Holidays must last for one full day. This type of operation is typically performed either by the Oracle Beehive administrator, or by a user who administers holidays for your organization.

Note: Meetings created in a team workspace with a "Holiday" or "Holidays" category using a CalDAV client (such as Mozilla Lightning) will be automatically converted to holidays.

This section includes the following topics:

- Creating an iCalendar File
- Sample Holiday Entries
- Importing Holiday iCalendar Files

Creating an iCalendar File

Before importing holidays, you will need to create an iCalendar file with one or more holiday events.

Oracle Beehive will only successfully import iCalendar files that have been formatted according to the Official Internet Protocol Standards for iCalendar.

See Also: For information about iCalendar standards, including formatting and representing iCalendar objects, refer to RFC5545, *Internet Calendaring and Scheduling Core Object Specification (iCalendar)* document available from the Internet Engineering Task Force:

`http://www.rfc-editor.org/rfc/rfc5545.txt`

You can use the following text as a template for creating your iCalendar holiday file:

```
BEGIN:VCALENDAR
VERSION:2.0
CALSCALE:GREGORIAN
PRODID:-//ORACLE//NONSGML Beehive Time Management - //EN
<holiday1>
<holiday2>
<holidayN>
END:VCALENDAR
```

Where *<holiday1>*, *<holiday2>*, and *<holidayN>* represent different holidays that you want to import.

Note: The *<holiday1>*, *<holiday2>*, and *<holidayN>* entries in the template must be replaced by `VEVENT`-type entries. For samples of `VEVENT` holidays that can be imported to Oracle Beehive refer to "Sample Holiday Entries".

Sample Holiday Entries

This section includes samples of `VEVENT`-type entries representing holidays that can be placed into a properly formatted iCalendar file. For information about creating an iCalendar file refer to "Creating an iCalendar File".

Example 10–1 Sample Holiday Event Recurring on a Fixed Date

```
BEGIN:VEVENT
DTEND;VALUE=DATE:20080702
```

```
SUMMARY:Canada Day
DTSTAMP:20080507T132210Z
UID:2e49f2d5-fcd5-4d19-b46b-5ece651a8f46@example.com
DTSTART;VALUE=DATE:20080701
LAST-MODIFIED:20080507T132210Z
RRULE:FREQ=YEARLY
CATEGORIES:HOLIDAY
CREATED:20080507T132205Z
END:VEVENT
```

In Example 10–1, the `VEVENT` representing the holiday has an initial occurrence on July 1, 2008. The title of the holiday is "Canada Day", and has a yearly recurrence rule denoted by the `RRULE` parameter: the holiday occurs every July 1st.

Example 10–2 Sample Holiday Event Recurring on a Variable Date

```
BEGIN:VEVENT
DTEND;VALUE=DATE:20080527
SUMMARY:Spring Bank Holiday
DTSTAMP:20080507T132214Z
UID:544c9369-3a0a-42d6-be25-0fc84b8091fd@example.com
DTSTART;VALUE=DATE:20080526
LAST-MODIFIED:20080507T132214Z
RRULE:FREQ=YEARLY;BYMONTH=5;BYDAY=4MO
CATEGORIES:HOLIDAY
CREATED:20080507T132210Z
END:VEVENT
```

In Example 10–2, the `VEVENT` representing the holiday has an initial occurrence on May 26, 2008. The title of the holiday is "Spring Bank Holiday", and has a yearly recurrence rule denoted by the `RRULE` parameter: the holiday occurs on the fourth Monday, every month of May.

Importing Holiday iCalendar Files

Once you have created an iCalendar file and populated it with `VEVENT`-type holiday entries, use the `beectl import_icalendar` command to import the holiday events in Oracle Beehive.

For information about creating an iCalendar file, or sample `VEVENT`-type entries, refer to "Creating an iCalendar File", and "Sample Holiday Entries".

Follow these steps to import holidays from an iCalendar file:

1. Save the iCalendar file with the holiday entries in an accessible location on the computer running Oracle Beehive.

2. Find the team workspace calendar into which the holidays should be imported by using the `beectl list_calendars` command:

   ```
   beectl> list_calendars --select_by_address teamworkspace@example.com
   ```

3. Run the `beectl import_icalendar` command to import your iCalendar file:

   ```
   beectl> import_icalendar --file <icalendar_file> --do_as_authuser <user id>
   --calendar <calendar> --holiday
   ```

   Where <*icalendar_file*> represents the absolute path to the iCalendar file saved in Step 1, <*calendar*> represents the calendar identifier you located in Step 2, and <*user id*> is the user identifier (a principal or the login ID) of the user importing the holiday entries.

Example 10–3 Importing Holiday Events

```
beectl>list_calendars --select_by_address testtws@caldav.example.com

-------------------------------------------------------+--------------+---------
Calendar                                               | IsDefault    | Name
                                                       |              |
-------------------------------------------------------+--------------+---------
clnd=MyCalendar,wksp=test_team_workspace,enpr=ent1     | Yes          | Calendar
-------------------------------------------------------+--------------+---------

beectl> import_icalendar --file /tmp/holiday.ics --do_as_authuser user=jsmith
--calendar "clnd=MyCalendar,wksp=test_team_workspace,enpr=ent1" --holiday

Imported invitation series, unique
identifier=3449:5915:ocrs:A6E2F29FEB7A49DB9F27C0C3E322641300000000002, iCalendar
UID=2e49f2d5-fcd5-4d19-b46b-5ece651a8f46@example.com.
Imported invitation series, unique
identifier=3449:5915:ocrs:A6E2F29FEB7A49DB9F27C0C3E322641300000000006, iCalendar
UID=544c9369-3a0a-42d6-be25-0fc84b8091fd@example.com.
```

In Example 10–3, a file named `holiday.ics` is being imported from the `/tmp` directory. The events will be imported as user `jsmith` in a team worskpace default calendar with the email address `testtws@caldav.example.com`, and the **--holiday** option denotes that the events within the file are holiday-type entries. The resulting output on the command line indicates the two unique invitation series have been imported.

Modifying User Time Management Preferences

This section contains information related to changing specific user preferences related to Oracle Beehive Time Management services, and includes the following topics:

- Changing a User's Time Zone Preference
- Changing a User's Defined Working Hours

Changing a User's Time Zone Preference

When a user temporarily or permanently changes geographic locations, their preferred time zone may change. This section includes information about changing a user's time zone, including determining available time zone identifiers.

Once a user's time zone preference has been changed, events will appear in the user's calendar -- offset by the appropriate number of hours -- relative to their new geographic location.

To change a user's defined time zone preference:

1. Determine the time zone identifier of the time zone that you want to assign to a user. Use the `beectl list_timezones` command to obtain a list of available time zones and their associated identifiers:

   ```
   beectl> list_timezones
   ```

   > **Note:** To list all available time zones on your Oracle Beehive deployment specify the **--all** option. When the **--all** option is not specified only common time zones will be returned.

2. Run the `beectl modify_user` command to assign a new time zone for the specified user:

   ```
   beectl> modify_user --email <address> --timezone <ID>
   ```

   Where <*address*> represents the e-mail address of the user, and <*ID*> represents the identifier of the time zone.

   > **Note:** You can use the **--user** option instead of the **--email** option to identify the user. For more information, refer to `modify_user`, in the *Oracle Beehive Administrator's Reference Guide*.

Example 10–4 Changing a User's Time Zone

```
beectl> modify_user --email jsmith@example.com --timezone tmzn=Europe/Berlin

Successfully modified user: jsmith@example.com
```

In Example 10–4, the user with e-mail address jsmith@example.com had their time zone preference changed to `Europe/Berlin`. The resulting output on the command-line indicates the modification was successful.

Changing a User's Defined Working Hours

Working hours are set in a user's property preference profile, and can be changed using the `beectl add_preference_property` command.

The values set for a user's working hours helps other users determine when the person is most likely to be available for meetings. By default, a user's working hours are set to begin at 9:00 AM, and end at 6:00 PM, in the user's defined time zone.

> **Note:** When an administrator changes a user's working hours, it can take up to 24 hours for the information to take effect across the system.

To change a user's defined working hours:

1. Use the `beectl list_preference_profiles` command to list the user's preference properties:

   ```
   beectl> list_preference_profiles --consumer <userid>
   ```

   Where <*userid*> represents the identifier of the user.

2. Using the output returned from Step 1, locate the `TimeManagement` section. Within the section, locate the `working_hours` preference property, then locate the text below the `value` field. Copy the entire XML code block, within and including the XML declaration element (for example, `<?xml>`). The string will resemble the following text:

   ```
   <?xml version = '1.0' encoding = 'UTF-8'?><WeekBusinessHours
   xmlns="http://xmlns.oracle.com/2006/Beehive/BOM/business-hours"><WeekShift
   StartDay="MONDAY" StartTime="09:00:00" EndDay="MONDAY" EndTime="18:00:00"
   Type="REGULAR"/><WeekShift StartDay="TUESDAY" StartTime="09:00:00"
   EndDay="TUESDAY" EndTime="18:00:00" Type="REGULAR"/><WeekShift
   StartDay="WEDNESDAY" StartTime="09:00:00" EndDay="WEDNESDAY" EndTime="18:00:00"
   Type="REGULAR"/><WeekShift StartDay="THURSDAY" StartTime="09:00:00"
   EndDay="THURSDAY" EndTime="18:00:00" Type="REGULAR"/><WeekShift
   ```

```
StartDay="FRIDAY" StartTime="09:00:00" EndDay="FRIDAY"
EndTime="18:00:00"Type="REGULAR"/></WeekBusinessHours>
```

3. Paste the text copied in Step 2 into a text editor. Modify the `StartTime` and `EndTime` values of each work day to reflect the new working hours for the user.
4. Save the file you created in Step 3.
5. Use the `beectl add_preference_property` command with the --file option to import the new working hours for the user:

   ```
   beectl> add_preference_property --set prfs=TimeManagement,<userID> --name
   working_hours --type BUSINESS_HOURS --file <filename>
   ```

   Where <*userID*> represents the user identifier of the user, and <*filename*> represents the text file you created in Step 4.

Example 10–5 Changing a User's Working Hours

```
beectl> add_preference_property --set prfs=TimeManagement,user=jsmith --name
working_hours --type BUSINESS_HOURS --file new_business_hours.xml
```

In Example 10–5, the working hours for the user with user identifier `user=jsmith` was modified based on the contents of the `new_business_hours.xml` file.

Managing Calendars and Task Lists

Users, resources, and workspaces can all own calendars and task lists. Users can create calendars and task lists using various Oracle Beehive clients, and users can modify calendars and task lists they own or have permission to manage using those clients as well.

As an administrator, you can also modify calendars and task lists using `beectl` commands. In addition to changing calendar and task lists attributes (such as name, permissions, time zone, and so forth), you can also modify who is permitted to access a calendar or task list, and who is enrolled with a given calendar.

This section contains the following topics:

- Managing Calendar and Task List Attributes
- Managing Calendar and Task List Permissions
- Managing Calendar Enrollments

Managing Calendar and Task List Attributes

Calendars and task lists are usually created when user accounts and workspaces are created, and as users perform their work, using the various Oracle Beehive clients. You can modify the attributes of an existing calendar or task list using beectl.

Using the `beectl modify_calendar` and `beectl modify_tasklist` commands, you can modify the attributes of calendars. Table 10–1, " Oracle Beehive Calendar Attributes" and Table 10–2, " Oracle Beehive Task List Attributes" lists the attributes you can modify with each command.

These commands provide similar options for selecting the specific calendar or task list you want to modify:

- **--calendar or --tasklist**: Use this option to select a calendar or task list explicitly by using its unique Identifier.

- **--calendarowner or --tasklistowner**: Use this option to select a calendar or task list using its owner's unique Identifier.
- **--select_by_<owner>**: Use one of the 'select by' options to select a calendar or task list based on its owner's name (a user, resource, or workspace) or its unique URI.

> **Note:** When you use one of these command options and Oracle Beehive finds more than one calendar or task list based on your query, the *default* calendar or task list will be modified.

Table 10–1 Oracle Beehive Calendar Attributes

Option	Description	Accepted Values
--name	Specifies the display name for the calendar.	An alphanumeric string
--timezone	Specifies the primary time zone for this calendar.	A valid time zone string
--booking_ characteristics	Specifies the booking behavior of the resource calendar; either open or first-come-first-served. The value is not case-sensitive. This option applies to Resource calendars only.	O (for open) F (for first-come-first-served)
--include_in_freebusy	Specifies whether this calendar is used when determining the owner's free/busy data. The value is not case-sensitive.	Y (to include in free/busy) N (to not include in free/busy)
--priority	Specifies the default priority of the calendar (the priority that, by default, is assigned to new calendar entries). The value is not case-sensitive.	HIGH MEDIUM LOW NONE
--sensitivity	Specifies the default sensitivity of the calendar. The value is not case-sensitive.	PUBLIC NORMAL CONFIDENTIAL PERSONAL PRIVATE
--enrollment_type	Specifies the enrollment type of the team workspace calendar. The value is not case-sensitive. This option applies to team workspace calendars only.	PUBLIC PRIVATE
--self_enrollment	Specifies a team workspace's permission for self-enrollment in the calendar. The value is not case-sensitive. When set to OPEN, any Oracle Beehive user can enroll in the team workspace calendar.	OPEN CLOSED
--caldav_resource_ name	Specifies the CalDAV resource name of the calendar.	A string
--derive_timezone	Specifies whether the calendar will derive its time zone from the calendar owner's configuration.	Y (to derive time zone from the owner) N

Table 10–1 (Cont.) Oracle Beehive Calendar Attributes

Option	Description	Accepted Values
--derive_available_hours	Specifies whether the calendar will derive its available hours from the calendar owner's configuration.	Y (to derive available hours from the owner) N
--enable_presence	Specifies whether the calendar is enabled for presence integration with Oracle Beehive's presence services. This option is not valid for team workspace calendars.	Y (to enable presence integration) N
--enroll_members	Specifies whether the members of a team workspace are automatically enrolled in this calendar or not. This option is only valid for team workspace calendars.	Y (to automatically enroll members) N

Table 10–2 Oracle Beehive Task List Attributes

Option	Description	Accepted Values
--name	Specifies the display name for the task list.	An alphanumeric string
--timezone	Specifies the primary time zone for this task list.	A valid time zone string
--priority	Specifies the default priority of the task list (the priority that, by default, is assigned to new task lists). The value is not case-sensitive.	HIGH MEDIUM LOW NONE
--sensitivity	Specifies the default sensitivity of the task list. The value is not case-sensitive.	PUBLIC NORMAL CONFIDENTIAL PERSONAL PRIVATE
--caldav_resource_name	Specifies the CalDAV resource name of the task list.	A string
--derive_timezone	Specifies whether the task list will derive its time zone from the task list owner's configuration.	Y (to derive time zone from the owner) N

Managing Calendar and Task List Permissions

Oracle Beehive provides the ability for one user to delegate to other users the ability to manage their calendar or task list. You can list, modify, and clear these permissions lists.

To list the permissions list for a calendar or task list, use the `beectl list_calendar_permissions` or `beectl list_tasklist_permissions` commands:

```
beectl> list_calendar_permissions [...]
beectl> list_tasklist_permissions [...]
```

These commands provide similar options for selecting the specific calendar or task list you want to review:

- **--calendar or --tasklist**: Use this option to select a calendar or task list explicitly by using its unique Identifier.
- **--calendarowner or --tasklistowner**: Use this option to select a calendar or task list using its owner's unique Identifier.
- **--select_by_<owner>**: use one of the 'select by' options to select a calendar based on its owner's name (a user, resource, or workspace) or its unique URI.

> **Note:** When you use one of these command options and Oracle Beehive finds more than one calendar or task list based on your query, the *default* calendar or task list will be listed.

To add or remove users to a calendar or task list's permissions list, use the `beectl modify_calendar_permissions` or `beectl modify_tasklist_permissions` commands:

```
beectl modify_calendar_permissions [...] --user <userid> | --group <groupid>
[--user_principal | --delegated_principal]  [--can_invite | --cannot_invite]
[--manage <sensitivity>] [--read <sensitivity>] [--discover <sensitivity>] [--deny
<sensitivity>]
```

> **Note:** In this example, the options for selecting a calendar or task list to modify have been omitted. They are the same as those described for listing calendar and task list permissions.

You must combine an option to select a calendar or task list, an option to specify a user or group to apply permissions to, and what is granted and on which sensitivities (manage, read, discover, or deny).

For a given calendar or task list combined with a given grantee, the rights specified in the command *add* to any pre-existing or previously set rights for that grantee on that artifact.

For example, suppose you run the `beectl modify_calendar_permissions` command and specify a calendar `--calendar C`, a user `--user user=john.doe`, and rights `--read CONFIDENTIAL`.

Later, you run the `beectl modify_calendar_permissions` command again, specifying the same calendar `--calendar C` and the same user `--user user=john.doe`, with the new rights `--read NORMAL`.

After running the second command, the user will have Read rights on both the NORMAL sensitivity and the CONFIDENTIAL sensitivity.

To grant the same right on multiple sensitivities to a user for a calendar or task list, you can also specify all of the rights you want to grant in a single command:

```
beectl> modify_calendar_permissions [...] --calendar C --user user=john.doe --read
CONFIDENTIAL --read NORMAL
```

Table 10–3 lists the options for allowing and denying various types of access to users and groups, for calendars and task lists.

Table 10–3 Oracle Beehive Calendar and Task List Permissions Options

Option	Description	Accepted Values
--user	A single user which you want to add, modify, or remove from the calendar or task list's permissions.	A valid Oracle Beehive user identifier
--group	An Oracle Beehive UDS group whose members you want to add, modify, or remove from the calendar or task list's permissions. Group members will automatically inherit these permissions.	A valid Oracle Beehive group identifier
--user_principal	Specifies that you want to modify a calendar or task list's owner's permissions only (Oracle Beehive will not modify any access given to this user as a delegate).	N/A
--delegated_principal	Specifies that you want to modify a calendar or task list's delegated principal only (Oracle Beehive will not modify the access given to this user directly).	N/A
--can_invite	Specifies the user or group may invite this calendar. (This option is not available for task lists.)	N/A
--cannot_invite	Specifies the user or group may not invite this calendar. (This option is not available for task lists.)	N/A
--can_assign	Specifies the user or group may assign this task list. (This option is not available for calendars.)	N/A
--cannot_assign	Specifies the user or group may not assign this task list. (This option is not available for calendars.)	N/A

--manage	Grants manage access (delegate) to the grantee (user only) on specified sensitivities of this calendar or task list. You can provide multiple values by using the --manage option multiple times. The value is not case-sensitive.	`PUBLIC`
		`NORMAL`
		`CONFIDENTIAL`
		`PERSONAL`
		`PRIVATE`

--read	Grants read access to the grantee (user or group) on specified sensitivities of this calendar or task list. You can provide multiple values by using the --read option multiple times. The value is not case-sensitive.	`PUBLIC`
		`NORMAL`
		`CONFIDENTIAL`
		`PERSONAL`
		`PRIVATE`

--discover	Grants discover access to the grantee (user or group) on specified sensitivities of this calendar. You can provide multiple values by using the --discover option multiple times. (This option is not available for task lists.) The value is not case-sensitive.	`PUBLIC`
		`NORMAL`
		`CONFIDENTIAL`
		`PERSONAL`
		`PRIVATE`

--deny	Denies access of the grantee (user or group) to content with the specified sensitivities in this calendar or task list. You can provide multiple values by using the --deny option multiple times. The value is not case-sensitive.	`PUBLIC`
		`NORMAL`
		`CONFIDENTIAL`
		`PERSONAL`
		`PRIVATE`

You can clear the list of permissions for a calendar or task list by using the `beectl clear_calendar_permissions` or `beectl clear_tasklist_permissions` commands:

```
beectl> clear_calendar_permissions [...]
beectl clear_tasklist_permissions [...]
```

Refer to Table 10–1, " Oracle Beehive Calendar Attributes", Table 10–2, " Oracle Beehive Task List Attributes" and Table 10–3, " Oracle Beehive Calendar and Task List Permissions Options" for explanations of the various options for these commands.

Managing Calendar Enrollments

The Beehive server offers flexibility with calendaring and scheduling operations, particularly in the context of team workspace environments. Team workspace calendars support the following features:

- Appointments, without any attendees to the event
- Traditional invitations in which attendees are explicitly listed
- Enrollment invitations, in which a team workspace is invited to an event, automatically inviting users who are enrolled

The various Oracle Beehive clients each present calendaring operations in different ways; not all of these functions are supported in every client. Enrollments apply to team workspaces only. By default all team workspace members are enrolled in the team workspace calendar. They can modify this behavior in some of the Oracle Beehive clients.

> **Note:** Enrollments apply to the team workspace, not individual calendars. When multiple calendars exist in a workspace, only the default team workspace calendar is affected by the workspace enrollment list.

Team workspace members can also 'unenroll' from the workspace. Unenrolled users will not receive invitations when the workspace is invited to meetings or when meetings are created in the team workspace.

For example, you could create a public team workspace about local sports events, with 'SelfEnrollment=OPEN'. You can populate the workspace calendar with the dates and times of sports events. Users who want to attend such events can self-enroll in the calendar, and the events will appear in their personal calendars.

As another example, an executive might be interested in accessing documents stored in a team workspace, but does not want to be included in the team workspace’s calendar events. The executive can join the team workspace but unenroll in its calendar.

You can list the users that are directly enrolled or unenrolled in a team workspace calendar by using the `beectl list_calendar_enrollments` command:

```
beectl> list_calendar_enrollments { --calendar <calendarid> | --calendarowner
<unique_identifier> | --select_by_address <uri> | --select_by_workspace_name
<workspace_name> }
```

The options for this command provide you with a variety of ways to specify a calendar.

To enroll or unenroll users from a team workspace calendar, use the `beectl modify_calendar_enrollments` command:

```
beectl> modify_calendar_enrollments { --calendar <calendarid> | --calendarowner
<unique_identifier> | --select_by_address <uri> | --select_by_workspace_name
<workspace_name> } { --enroll <loginid> | --unenroll <loginid>}
```

Specify the calendar you want to modify using one of the options within the first set of curly-braces {}, and then specify either --enroll or --unenroll with the identifier of the user you want to enroll or unenroll in the calendar.

Managing Oracle Beehive Time Zone Definitions

This section includes information about time zone definitions, obtaining new time zone packages, and importing new time zone files into Oracle Beehive. This section contains the following topics:

- Overview
- Managing Common Time Zones
- Obtaining a New Oracle Beehive Time Zone Package
- Refreshing the Oracle Beehive Time Zone Package

Overview

Every installation of Oracle Beehive includes a set of time zones with associated rules. As time zones change as a result of political decisions, Oracle Beehive time zones will be updated as part of the regular upgrade process, or by applying a more recent Oracle Beehive Time Zones Package.

The contents of the Oracle Beehive Time Zones Package is based on the time zone definitions provided by the Public-Domain Time Zone Database Web site maintained at the National Institute of Health. For more information, consult the external Web site at the following address:

http://www.twinsun.com/tz/tz-link.htm

Managing Common Time Zones

Oracle Beehive uses a list of "common" time zones, which are those most frequently used by various clients as the preferred time zone choices to offer to users. In addition, Oracle Beehive supports a comprehensive list of time zones, from which the common time zones are drawn.

By default, Oracle Beehive provides over 400 time zone definitions, of which about 75 are designated as common.

When you use the `beectl list_timezones` command *without* the `--all` option, only the common time zones are listed. When you use the command *with* the `--all` option, all time zones are listed.

You can manage which time zones are included in the common time zone list, by using the `beectl modify_timezones` command with the `--common` option. For example, to set the Africa/Kigali time zone to common:

```
beectl> modify_timezones select_by_name Africa/Kigali --common
```

You can reset the common time zone list to the default settings (as provided by a fresh Oracle Beehive installation) by using the `beectl modify_timezones` command with the `--reset_to_default` option:

```
beectl modify_timezones --reset_to_default
```

Obtaining a New Oracle Beehive Time Zone Package

The most recent Oracle Beehive Time Zone Package is usually included with the most recent Oracle Beehive patch.

Consult Oracle Support for information about obtaining time zone packages that are made available between Oracle Beehive releases.

> **Note:** You can find the version of the Oracle Beehive Time Zone Package currently on your system by looking at the first few lines of the `beectl list_timezones` command.

Refreshing the Oracle Beehive Time Zone Package

Once you have obtained the newest time zone package, you will need to import the file using the `beectl import_timezones` command.

To import a time zone package:

1. Save the time zone package XML file in an accessible location on the computer running Oracle Beehive.
2. Run the `beectl import_timezones` command to import the Oracle Beehive Time Zone Package file:

   ```
   beectl> import_timezones --file <timezone_package>
   ```

   Where <*timezone_package*> represents the absolute path to the time zone package file saved in Step 1.

Example 10–6 Importing a Time Zone Definition Package

```
beectl> import_timezones --file /tmp/timezones/tzdata2008a-085.xml

Time zones are successfully imported to database.
```

In Example 10–6, the time zone definition package file named `tzdata2008a-085.xml` is being imported from the `/tmp/timezones` directory. A line is returned to the command-line indicating that the time zones were successfully imported.

Enabling Cross-Scheduling Between Oracle Beehive Deployments with iSchedule

Oracle Beehive supports the Internet Calendar Scheduling Protocol (iSchedule). iSchedule enables interoperability between different calendaring and scheduling systems. With iSchedule, users in connected systems can perform common calendaring and scheduling tasks, such as scheduling and rescheduling meetings, responding to meeting requests, and searching for free-busy time of other users, regardless of which system a user resides. Oracle Beehive 2.0 supports iSchedule interoperability between Oracle Beehive deployments only.

To enable cross-scheduling between two different Oracle Beehive deployments, perform the following steps:

1. Verify that the Oracle Beehive deployments are capable of sending and receiving HTTPS requests to and from each other. In some deployment scenarios, Oracle

Beehive servers are deployed behind a firewall that prevents HTTPS requests to be sent.

2. On the first Oracle Beehive instance, use `beectl` to configure the iSchedule server to point to that of the second Oracle Beehive instance.

   For example, in a scenario where the first Oracle Beehive instance is called "instance1.net" and the second instance is called "instance2.com", run the following commands on instance1.net:

   ```
   % beectl
   beectl> add_ischedule_server --name BEEHIVE-INSTANCE2-COM \
        --domain_regexp ".*@instance2\.com$" \
        --outgoing_url "https://beehive.instance2.com/ischedule" \
        --outgoing_auth_id ischedule \
        --outgoing_auth_key Welcome123 \
        --incoming_auth_id ischedule \
        --incoming_auth_key Beehive123 \
        --activate_configuration
   ```

3. On the second Oracle Beehive instance, use `beectl` to configure the iSchedule server to point to that of the first Oracle Beehive instance.

   For example, using the above scenario, run the following commands on instance2.com:

   ```
   beectl> add_ischedule_server --name BEEHIVE-INSTANCE1-NET \
        --domain_regexp ".*@instance1\.net$" \
        --outgoing_url "https://beehive.instance1.net/ischedule" \
        --outgoing_auth_id ischedule \
        --outgoing_auth_key Beehive123 \
        --incoming_auth_id ischedule \
        --incoming_auth_key Welcome123 \
        --activate_configuration
   ```

Note: If the `add_ischedule_server` command is executed directly on the command line, that is, not in the `beectl` shell, the value of the `--outgoing_auth_key` and `--incoming_auth_key` options will first need to be obfuscated using the `beectl obfuscate` command.

11

Managing Oracle Beehive Events and Policies

This module describes how to view business events, and create and manage policies including the default Oracle Beehive policies. This module contains the following sections:

- Introduction to Beehive Events and Policies
- Managing Beehive Events
- Managing Beehive Policies

Introduction to Beehive Events and Policies

Oracle Beehive allows you to control how the system will react to a wide variety of user and system-generated events, collectively called "business events". Virtually every type of user interaction with the system, such as logging in, sending a message, performing a search, or editing a file, is "trapped," meaning, Oracle Beehive generates a business event. Events are loggable (and are logged according to the current log level), and any event may be used to trigger a policy. Oracle Beehive exposes about 400 business events.

Policies in Oracle Beehive are sets of ordered rules. Rules are if/then statements, which determine a response to a given condition. Policies are designed to be triggered from events, which determine how the system should behave. For example, the password policy determines, based on a set of criteria, whether a supplied new password is acceptable, or should be rejected. The provisioning policy determines which objects are created, by default, in a new user’s personal workspace. You can make use of auditing policies to cause various user actions to be logged to an audit trail, and made available for analysis by privileged auditor users.

This section includes the following topics:

- Introduction to Beehive Events
- Introduction to Beehive Policies

Introduction to Beehive Events

Oracle Beehive provides about 400 business events on which you can base policies and generate notifications. In addition to these "non-blocking" events (asynchronous events), there are a number of "blocking" events (synchronous events). Synchronous events are not available for use in custom policies.

Synchronous Events

A synchronous event invocation is a "blocking call". What this means is that the event itself is prevented from completing until all policies have been evaluated to TRUE. However, in current Oracle Beehive versions, only the password policy, provisioning policy, and deprovisioning policy may make use of a synchronous event. Default password, provisioning, and deprovisioning policies are provided. You may not use synchronous events in your custom policies.

Asynchronous Events

Asynchronous events are used to customize what happens after an event completes. Policies may do something that has an impact on an entity or artifact involved the event (such as a file being updated), but the event first completes and then invokes these custom actions. For example, an asynchronous event raised after a document has been updated could be used for a policy that sends a notification to the document owner. In this case, first the file is updated, and then the event triggers the notification. You can define system-wide server-side rules by using asynchronous message-related events.

In fact, the Oracle Beehive subscriptions and notifications functionality makes extensive use of asynchronous events to trigger notifications about meeting updates, document updates, and so forth.

Asynchronous events may also be used for sending alerts. For example, a policy could send an urgent message to the mobile device of an administrator whenever a serious system fault occurs.

Asynchronous events are handled by an event queue. The event (and corresponding event payload) is accepted by the event management system, put in an event queue for later processing, and then the control is returned to the caller. This activity is transparent to users and administrators, although it may be logged for system troubleshooting purposes.

Event Subscriptions

Event subscriptions are the actions that can be attached to an event with an optional condition. When the subscription condition evaluates to true, the action attached to an event get executed.

The actions are defined by internal Oracle Beehive services and included in all deployments automatically. These actions are exposed through the policies.

Disabled Events

Certain events are disabled by default. All of the disabled events are part of the Time Management Service. The following events are not generated by the system by default:

- ASSIGNMENT_ADDED
- ASSIGNMENT_REMOVED
- ASSIGNMENT_UPDATED
- CALENDAR_ADDED
- CALENDAR_REMOVED
- CALENDAR_UPDATED
- DEFAULT_REMINDER_ADDED
- DEFAULT_REMINDER_REMOVED

- DEFAULT_REMINDER_UPDATED
- INVITATION_ADDED
- INVITATION_REMOVED
- INVITATION_UPDATED
- OCCURRENCE_ADDED
- OCCURRENCE_REMOVED
- OCCURRENCE_UPDATED
- REMINDER_ADDED
- REMINDER_REMOVED
- REMINDER_UPDATED
- RESOURCE_CREATED
- RESOURCE_UPDATED
- RESOURCE_DELETED
- TASKLIST_ADDED
- TASKLIST_REMOVED
- TASKLIST_UPDATED
- TODO_ADDED
- TODO_REMOVED
- TODO_UPDATED

If you want to use any of these events, you must enable them by changing the `EnableGenericClassOfTMBusinessEvents` property of the Time Management Service. See Chapter 4, "Oracle Beehive Parameter Reference," in the *Oracle Beehive Administrator's Reference Guide*.

Introduction to Beehive Policies

Policies are triggered by events. They establish rules for how the system should behave when certain events occur, based on evaluating the truth of a set of conditions, and then allowing or disallowing a resulting action.

Each policy is triggered by events.

A policy has one or more rules, each of which is triggered by one event.

Each rule contains one or more conditions, which are evaluated as true or false.

Each rule may activate an action, depending on the results of the evaluated conditions.

For example, the password policy is triggered whenever a user modifies their password; the password is evaluated (by a rule) by testing whether various conditions, such as minimum length, whether it was already used previously, whether it has numbers or special characters in it, and so forth, are true or false; and then, an action allows the password change if all of the conditions are successfully met.

Oracle Beehive is shipped and installed with three default policies:

- Provisioning policy
- Deprovisioning policy

- Password policy

Additionally, Oracle Beehive includes default audit policy templates created during the installation process. However, you cannot access auditing functionality through the policy framework.

If you configure Oracle Beehive with Oracle Universal Records Manager (URM), you can use the policy framework to create records management policies. See "Managing Records Management" on page 6-26 for details.

You can use these default policies without changes if you wish, or you can modify them to suit the requirements of your organization. In addition, you can create new, custom policies.

Managing Beehive Events

Oracle Beehive provides about 400 events for use by policies, logging, auditing, and other functions. Events are divided into two categories: synchronous events, and asynchronous events. Synchronous events are used internally and in default policies by Oracle Beehive. Asynchronous events are available for you to work with when creating custom policies.

You can list all available events using the `beectl list_events` command:

```
beectl> list_events
```

Each event is listed, along with an indication of whether it is synchronous (Y or N), and a short description of the event. Only asynchronous events are listed.

You can get detailed information about any event by using the `beectl list_events` command with the `--event_name` option. For example:

```
beectl> list_events --event_name DOCUMENT_UPDATED
```

This command produces output similar to the following:

```
Event Name: DOCUMENT_UPDATED
Event Description: Raised when an update to a document is about to be committed.
Is Synchronous: N

------------------------------------

Event Subscriptions:
------------------------------------
There are no event subscriptions to be listed.
------------------------------------

------------------------------------

Event Attributes:
------------------------------------

     Name: COMMON_ATTRIBUTES              Type: BEE_CODE.ECA_COMMON_EVENT_ATTRIBS_
T

         Name: ENTITY_ID                      Type: BEE_CODE.OCS_COLLAB_ID_T

         Name: CONTAINER                      Type: BEE_CODE.OCS_COLLAB_ID_T

         Name: ACTOR_ID                       Type: BEE_CODE.OCS_COLLAB_ID_T
         Name: OPERATION                      Type: STRING

         Name: STATUS                         Type: STRING

         Name: MESSAGE                        Type: STRING

         Name: EVENT_NAME                     Type: STRING

         Name: LOGON_RECORD_ID                Type: INTEGER

         Name: EVENT_ID                       Type: INTEGER

     Name: CUSTOM_ATTRIBUTES              Type: BEE_CODE.WS_DOCUMENT_EVENT_
ATTRIBS_T

         Name: ARTIFACT_ATTRIBUTES            Type: BEE_CODE.AM_COMMON_EVENT_
ATTRIBS_T

             Name: SIZE_CHANGE                    Type: INTEGER

             Name: NEW_CONTAINER                  Type: BEE_CODE.OCS_COLLAB_ID_T
```

In addition to the name and description of the event, any subscriptions to the event are listed, and the event's attributes are detailed. The attributes include the event payload.

You specify events (using an event name) in custom polices that you create, and their attributes are made available to consuming policies.

Managing Beehive Policies

Oracle Beehive is pre-seeded during installation with three default policies:

- Provisioning policy
- Deprovisioning policy
- Password policy

You can modify each of these policies, and in addition, you can create new policies to suit the needs of your organization. Although you can manage policies using the beectl command-line tool, it is easier to manage policies using Oracle Beekeeper. Beekeeper has several modules that allow you to work with visual models of policies, policy templates, and policy schemas.

See Also:

- "Policy Definitions" in Chapter 1, "Oracle Beehive XML File Reference" of the *Oracle Beehive Administrator's Reference Guide*
- "Managing Policies" in Chapter 3, "Managing Enterprises" of the *Oracle Beekeeper Online Help*

This section contains the following topics:

- Managing the Provisioning Policy
- Managing the Deprovisioning Policy
- Managing the Password Policy
- Managing Audit Policies
- Creating and Managing Custom Policies

- Using Dynamic Policy Attributes

Managing the Provisioning Policy

The provisioning policy is a definition of rules and actions that take affect when you create a user account. When Oracle Beehive is installed, a default user provisioning policy is seeded.

Provisioning policy rules can be applied based on any of the following:

- Account type: enterprise user or extended enterprise user
- Organizations to which the user belongs
- Manager
- Location
- Job Title

Like all policies, the provisioning policy couples rules with actions. The provisioning policy always use the same action:

```
<ActionInfo>
   <name>ProvisioningPLSQLAction</name>
</ActionInfo>
```

The provisioning action can specify the following:

- Which workspace template to use to create the user's personal workspace
- Which groups the user should be added to

Note: The user provisioning policy can only be applied during user account creation. As a result, it is preferable to add the user to dynamic groups, based on user attributes. Doing so eliminates the need for administrator action in the future, when user attributes such as job title or manager change.

To view the default provisioning policy, use the `beectl export_policy` command to export the provisioning policy to an XML-formatted file:

```
beectl> export_policy --policy_name UserProvisioningPolicy --scope
enpr=<enterprise alias> --destination /tmp
```

You must provide the name of the policy, and the scope container of the policy (in this case, the enterprise alias). This command creates a file in the `/tmp` folder called UserProvisioningPolicy.xml. You can edit this file and then use the `beectl modify_policy` command to upload your changes.

Alternatively, you can create an entirely new provisioning policy, and then use the `beectl modify_policy` command to overwrite the existing policy:

```
beectl> modify_policy --file <full path to the policy xml file>
```

Note: Your new policy must specify the collabID of the policy to be replaced or modified, in the `<collabId>` element at the beginning of the XML file.

Table 11–1, " User Attributes in Provisioning and Deprovisioning Policy Conditions" summarizes the field names you can use for user attributes in your provisioning (and deprovisioning) policy conditions. For each user attribute, enter the field name listed, and provide the value shown in the table.

Table 11–1 User Attributes in Provisioning and Deprovisioning Policy Conditions

User Account Attribute	Field Name in Policy XML File	Valid Values
Office Location	custom_attributes.office_location	Value of office location
Job Title	custom_attributes.job_title	Value of job title
Is external user	custom_attributes.external_user	Y or N
Organization	custom_attributes.organization_cid	CollabID of organization
Manager	custom_attributes.manager_cid	CollabID of manager

In the action portion of the XML file, you can specify personal workspace templates and groups for the user account, by supplying the collabID of the template and each group.

You can only specify a single personal workspace template:

```
<actionPreferenceInfo>
    <key>template_cid</key>
    <value>collabID</value>
 </actionPreferenceInfo>
```

Enter the collabID of the personal workspace template inside the `<value>` element.

To specify multiple groups, use the following format:

```
<actionPreferenceInfo>
    <key>group_cids</key>
    <group_cid>collabID1</group_cid><group_cid>collabID2</group_cid>...
 </actionPreferenceInfo>
```

Enter the collabID inside each `<group_cid>` element. You can specify any number of groups.

Example 11–1, "Provisioning Policy with Customized Conditions" demonstrates a modified provisioning policy XML file with added conditions and modified action. In this example, a simple condition is tested, in the `<RuleInfo priority="1">` element: the policy checks if the user account being created has a job title of "MANAGER". If so, then an action is triggered, which is to select the "manager_template" personal workspace template. Otherwise, if the user account does not match that job title, a personal workspace template called "developer_template" is selected instead.

See Also: For more information about creating custom personal workspace templates, see "Managing Oracle Beehive Workspaces".

Note: The CollabID values shown are examples; you must replace them with correct CollabIDs from your own deployment.

Example 11–1 Provisioning Policy with Customized Conditions

```
<?xml version = '1.0' encoding = 'UTF-8'?>
<PolicyInfo isExtensible="true">
```

```
<collabId>4DA5:0F49:plcy:355E4C9DBE6147FDE040578C0309297700000000042A</collabId>

<containerId>4DA5:0F49:enpr:355E4C9DBE6147FDE040578C030929770000000001D0</containe
rId>
   <templateId></templateId>
   <name>UserProvisioningPolicy</name>
   <RuleInfos>
      <RuleInfo priority="1">
         <name>Provisioning_rule1</name>
         <eventTypeName>USER_PROVISIONING</eventTypeName>
         <ruleId></ruleId>
         <templateRuleIds/>
         <ConditionInfo>
            <conditionExpression>
               <simple>
                  <leftSide>CUSTOM_ATTRIBUTES.JOB_TITLE</leftSide>
                  <operator>=</operator>
                  <rightSide>'MANAGER'</rightSide>
               </simple>
            </conditionExpression>
         </ConditionInfo>
         <ActionInfos>
            <name>ProvisioningPLSQLAction</name>
            <description>Provisioning action</description>
            <actionTypeName>PLSQL</actionTypeName>
            <actionString>uds_user_provisioning.apply_provisioning_
policy</actionString>
         </ActionInfos>
         <ActionPreferenceInfos>
            <actionPreferenceInfo>
               <key>template_cid</key>
                 <value>manager_template_collabID</value>
            </actionPreferenceInfo>
            <actionPreferenceInfo>
               <key>group_cids</key>
               <value><group_cid>managers_group_collabID</group_cid></value>
            </actionPreferenceInfo>
         </ActionPreferenceInfos>
      </RuleInfo>
      <RuleInfo priority="1">
         <name>Provisioning_rule2</name>
         <eventTypeName>USER_PROVISIONING</eventTypeName>
         <ruleId></ruleId>
         <templateRuleIds/>
         <ConditionInfo>
            <conditionExpression>
               <simple>
                  <leftSide>CUSTOM_ATTRIBUTES.JOB_TITLE</leftSide>
                  <operator>!=</operator>
                  <rightSide>'DEVELOPER'</rightSide>
               </simple>
            </conditionExpression>
         </ConditionInfo>
         <ActionInfo>
            <name>ProvisioningPLSQLAction</name>
         </ActionInfo>
         <ActionPreferenceInfos>
            <actionPreferenceInfo>
               <key>template_cid</key>
```

```
                <value>developer_template_collabID</value>
            </actionPreferenceInfo>
        </ActionPreferenceInfos>
    </RuleInfo>
  </RuleInfos>
</PolicyInfo>
```

As this example demonstrates, a policy can string multiple `<RuleInfo>` elements together, each one coupling a set of rules with an action.

> **Note:** Rules have priority numbers. Rules which are mutually exclusive, such as the two rules in this example, can have the same priority because they will never both occur. If you specify several rules which can evaluate as true at the same time, each such rule should have a different priority number to indicate which rule is evaluated first.

Example 11–2, "Provisioning Policy with Complex Customized Conditions" demonstrates the use of a more complex condition in a provisioning policy. In this example, two conditions are tested: if the user has a job title of "MANAGER", and if the user belongs to an organization called "SALES". If both conditions are true, the "sales_manager_template" personal workspace template is assigned, and the user is added to both the "all_managers" and "abcdefg_managers" groups.

Example 11–2 Provisioning Policy with Complex Customized Conditions

```
<?xml version = '1.0' encoding = 'UTF-8'?>
<PolicyInfo isExtensible="true">

<collabId>4DA5:0F49:plcy:355E4C9DBE6147FDE040578C0309297700000000042A</collabId>

<containerId>4DA5:0F49:enpr:355E4C9DBE6147FDE040578C03092977000000001D0</containe
rId>
   <templateId></templateId>
   <name>UserProvisioningPolicy</name>
   <RuleInfos>
      <RuleInfo priority="1">
         <name>Provisioning_rule1</name>
         <eventTypeName>USER_PROVISIONING</eventTypeName>
         <ruleId></ruleId>
         <templateRuleIds/>
         <ConditionInfo>
            <conditionExpression>
              <conjunction>
                <LeftBooleanExpression>
                  <leftSide>custom_attributes.job_title</leftSide>
                  <operator>=</operator>
                  <rightSide>'MANAGER'</rightSide>
                </LeftBooleanExpression>
                <RightBooleanExpression>
                  <leftSide>custom_attributes.organization_cid</leftSide>
                  <operator>=</operator>
                  <rightSide>'SALES"</rightSide>
                </RightBooleanExpression>
              </conjunction>
            </conditionExpression>
         </ConditionInfo>
         <ActionInfo>
```

```
            <name>ProvisioningPLSQLAction</name>
         </ActionInfo>
         <ActionPreferenceInfos>
            <actionPreferenceInfo>
               <key>template_cid</key>
               <value>sales_manager_template _collabID</value>
            </actionPreferenceInfo>
            <actionPreferenceInfo>
               <key>group_cids</key>
               <value><group_cid>all_managers_group_collabID</group_cid><group_
cid>abcdefg_manager_group_collabid</group_cid></value>
            </actionPreferenceInfo>
         </ActionPreferenceInfos>
      </RuleInfo>
   </RuleInfos>
</PolicyInfo>
```

As this policy demonstrates, you can provide multiple `<actionPreferenceInfo>` elements in a policy action, each one directing the system to perform a task.

Managing the Deprovisioning Policy

The deprovisioning policy is the policy that is activated when an account is set to be deleted. Oracle Beehive is pre-seeded with an "empty" deprovisioning policy; by default it does not trigger any system actions when invoked. You may modify the deprovisioning policy to suit the needs of your organization.

Deprovisioning policy rules can be based on any of the following user attributes:

- Organization
- Manager
- Extended enterprise user
- Location
- Job title

Like all policies, the deprovisioning policy couples rules with actions. The deprovisioning policy always uses the same action:

```
<ActionInfo>
    <name>DeprovisioningPLSQLAction</name>
</ActionInfo>
```

The deprovisioning action can specify any of the following:

- A user to be the new owner for groups the deleted user owned
- A rule reassigning all owned groups to the user’s manager
- A rule specifying a new assistant, when the user account being deleted is an assistant to another user

For example, the deprovisioning policy might indicate that when a user is removed, if this user has a title of "director," all of the user's documents will be reassigned to the user's manager, and all groups that the user owned will be reassigned to the system administrator.

To view the default deprovisioning policy, use the `beectl export_policy` command to export the provisioning policy to an XML-formatted file:

```
beectl> export_policy --policy_name UserDeprovisioningPolicy --scope
```

```
enpr=<enterprise alias> --destination /tmp
```

You must provide the name of the policy, and the name of the scope container of the policy (in this case, the enterprise alias). This command creates a file in the `/tmp` folder called UserDeprovisioningPolicy.xml. You can edit this file and then use the `beectl modify_policy` command to upload your changes.

Alternatively, you can create an entirely new deprovisioning policy, and then use the `beectl modify_policy` command to overwrite the existing policy:

```
beectl> modify_policy --file <full path to the policy xml file>
```

Note: Your new policy must specify the collabID of the policy to be replaced or modified, in the `<collabId>` element at the beginning of the XML file.

Table 11–1, " User Attributes in Provisioning and Deprovisioning Policy Conditions" summarizes the field names you can use for user attributes in your deprovisioning (and provisioning) policy conditions. For each user attribute, enter the field name listed, and provide the value shown in the table.

In the action portion of the XML file, you can specify a new owner for the user's owned groups by supplying the collabID of the new owner:

```
<actionPreferenceInfo>
    <key>new_owner_cid</key>
    <value>new_owner_collabID</value>
 </actionPreferenceInfo>
```

You can indicate whether ownership should be re-assigned to the user's manager, using a value of `Y` or `N` in the `assign_ownership_to_manager` action preference:

```
<actionPreferenceInfo>
    <key>assign_ownership_to_manager</key>
    <value>Y</value>
  </actionPreferenceInfo>
```

You can indicate a new assistant if the user being deleted is an assistant to another user, by supplying the collabID of the new assistant:

```
<actionPreferenceInfo>
    <key>new_assistant_cid</key>
    <value>new_assistant_collabID</value>
 </actionPreferenceInfo>
```

Example 11–3, "Deprovisioning Policy with Customized Conditions" demonstrates how to modify the deprovisioning policy XML file to add customized conditions. In this example, a simple condition is tested, in the `<RuleInfo priority="1">` element: the policy checks if the user account being deleted belonged to the "SALES" organization. If so, then an action is triggered, which is to assign all groups the user owns to a specific user.

Note: The RuleID and CollabID values shown are examples; you must replace them with correct CollabIDs from your own deployment. If you export an existing policy, it will contain the correct ruleIDs and collabIDs.

Example 11–3 Deprovisioning Policy with Customized Conditions

```
<?xml version = '1.0' encoding = 'UTF-8'?>
<PolicyInfo isExtensible="true">

<collabId>4DA5:0F49:plcy:355E4C9DBE6147FDE040578C0309297700000000042E</collabId>

<containerId>4DA5:0F49:enpr:355E4C9DBE6147FDE040578C030929770000000001D0</containe
rId>
   <templateId></templateId>
   <name>UserDeprovisioningPolicy</name>
   <RuleInfos>
      <RuleInfo priority="1">
         <name>Deprovisioning PLSQL rule</name>
         <eventTypeName>USER_DEPROVISIONING</eventTypeName>

<ruleId>4DA5:0F49:rule:355E4C9DBE6147FDE040578C03092977000000000430</ruleId>
         <templateRuleIds/>
         <ConditionInfo>
            <conditionExpression>
               <simple>
                  <leftSide>CUSTOM_ATTRIBUTES.organization_cid</leftSide>
                  <operator>=</operator>
                  <rightSide>'SALES'</rightSide>
               </simple>
            </conditionExpression>
         </ConditionInfo>
         <ActionInfo>
            <name>DeprovisioningPLSQLAction</name>
         </ActionInfo>
         <ActionPreferenceInfos>
            <actionPreferenceInfo>
               <key>new_owner_cid</key>
               <value>new_owner_collabID</value>
            </actionPreferenceInfo>
         </ActionPreferenceInfos>
      </RuleInfo>
   </RuleInfos>
</PolicyInfo>
```

Managing the Password Policy

The password policy is a definition of rules and actions that take affect when a password for a user account is created or modified. When Oracle Beehive is installed, a default password policy is seeded.

Password policy rules can be applied based on any of the following:

- Minimum length
- Maximum length
- Contains alphabetic characters
- Contains upper-case alphabetic characters
- Contains non-alphanumeric characters
- Contains the user name
- Was previously used as a password by the user

Like all policies, the password policy couples rules with actions. The password policy always use the same action:

```
<ActionInfo>
   <name>Password modification action</name>
</ActionInfo>
```

The password policy action only specifies that the password modification action will be allowed. If all of the policy rules are met, the action allows the password to be modified; otherwise, it prevents the password modification.

To view the default password policy, use the `beectl export_policy` command to export the provisioning policy to an XML-formatted file:

```
beectl> export_policy --policy_name "Validate Password" --scope  enpr=<enterprise
alias> --destination /tmp
```

You must provide the name of the policy, and the name of the scope container of the policy (in this case, the enterprise alias). This command creates a file in the `/tmp` folder called PasswordPolicy.xml. You can edit this file and then use the `beectl modify_policy` command to upload your changes.

Alternatively, you can create an entirely new password policy, and then use the `beectl modify_policy` command to overwrite the existing policy:

```
beectl> modify_policy --file <full path to the policy xml file>
```

Note: Your new policy must specify the collabID of the policy to be replaced or modified, in the `<collabId>` element at the beginning of the XML file.

Example 11–4, "Default Password Policy" shows the content of the default password policy. You can use this example as a reference for how password conditions are constructed in the `<RuleInfos>` section of the policy. As you can see, the password policy rule is triggered by the `ON_AUTH_USER_PASSWD_MODIFICATION` event. This is a synchronous (blocking) event. When you modify the password policy, you should not change this value.

Each of the conditions in the password policy is written as a condition of exclusion. This means, if the password matches any condition, it is disallowed. If it does not match any condition, it is allowed. This functionality is exposed by the password policy blocking the calling event if any condition is true.

Example 11–4 Default Password Policy

```
<?xml version = '1.0' encoding = 'UTF-8'?>
<PolicyInfo isExtensible="true">
   <policy>plcy=PasswordPolicy,enpr=oracle</policy>
   <scope>enpr=enterprise_alias</scope>
   <template></template>
   <name>PasswordPolicy</name>
   <description>Password policy desc</description>
   <RuleInfos>
      <RuleInfo priority="1">
         <name>Password Auth PLSQL Rule</name>
         <description>Password Auth rule description</description>
         <eventTypeName>ON_AUTH_USER_PASSWD_MODIFICATION</eventTypeName>
```

```
<ruleId>713E:6031:rule:374B57D9F3BDC9A1E040578C211A7A46000000000419</ruleId>
         <toRemove>false</toRemove>
         <templateRuleIds/>
         <ConditionInfo>
             <DisjunctionInfo>
               <LeftBooleanExpression>
                  <DisjunctionInfo>
                     <LeftBooleanExpression>
                        <DisjunctionInfo>
                           <LeftBooleanExpression>
                              <DisjunctionInfo>
                                 <LeftBooleanExpression>
                                    <DisjunctionInfo>
                                       <LeftBooleanExpression>
                                          <Simple>
                                             <leftSide>LENGTH(CUSTOM_
ATTRIBUTES.USER_PASSWD)</leftSide>
                                             <operator>&lt;</operator>
                                             <rightSide>8</rightSide>
                                          </Simple>
                                       </LeftBooleanExpression>
                                       <RightBooleanExpression>
                                          <Simple>
                                             <leftSide>LENGTH(CUSTOM_
ATTRIBUTES.USER_PASSWD)</leftSide>
                                             <operator>></operator>
                                             <rightSide>128</rightSide>
                                          </Simple>
                                       </RightBooleanExpression>
                                    </DisjunctionInfo>
                                 </LeftBooleanExpression>
                                 <RightBooleanExpression>
                                    <Simple>
                                       <leftSide>AUTH_POLICY_FUNC_PKG.VERIFY_
ALPHANUM(CUSTOM_ATTRIBUTES.USER_PASSWD)</leftSide>
                                       <operator>=</operator>
                                       <rightSide>0</rightSide>
                                    </Simple>
                                 </RightBooleanExpression>
                              </DisjunctionInfo>
                           </LeftBooleanExpression>
                           <RightBooleanExpression>
                              <Simple>
                                 <leftSide>AUTH_POLICY_FUNC_PKG.CONTAINS_
USERNAME(CUSTOM_ATTRIBUTES.USER_NAME,CUSTOM_ATTRIBUTES.USER_PASSWD)</leftSide>
                                 <operator>=</operator>
                                 <rightSide>0</rightSide>
                              </Simple>
                           </RightBooleanExpression>
                        </DisjunctionInfo>
                     </LeftBooleanExpression>
                     <RightBooleanExpression>
                        <Simple>
                           <leftSide>AUTH_POLICY_FUNC_PKG.VERIFY_CASE(CUSTOM_
ATTRIBUTES.USER_PASSWD)</leftSide>
                           <operator>=</operator>
                           <rightSide>0</rightSide>
                        </Simple>
                     </RightBooleanExpression>
                  </DisjunctionInfo>
               </LeftBooleanExpression>
               <RightBooleanExpression>
                  <Simple>
                     <leftSide>AUTH_POLICY_FUNC_PKG.IN_HISTORY(CUSTOM_
ATTRIBUTES.USER_NAME,CUSTOM_ATTRIBUTES.ENCRYPTED_PASSWD)</leftSide>
                     <operator>=</operator>
                     <rightSide>0</rightSide>
                  </Simple>
               </RightBooleanExpression>
            </DisjunctionInfo>
         </ConditionInfo>
         <ActionInfo>
            <name>Password modification action</name>
         </ActionInfo>
         <ActionPreferenceInfos/>
      </RuleInfo>
   </RuleInfos>
</PolicyInfo>
```

This password policy sets the following rules for passwords:

- Must not be fewer than 8 characters in length
- Must not be more than 128 characters in length
- Must not have 0 alphanumeric characters
- Must contain both alphabetic and numeric characters
- Must not contain the user name
- Must not be a previously-used password (verified by checking the password history of the user)

Managing Audit Policies

Auditing allows you to track and record the activities of users and administrators as they perform actions in the system. These activities include logging on and off, creating, modifying, or deleting content, altering system configuration parameters, starting and stopping processes, and so forth. The goal is to provide a framework for keeping tabs on who does what to the system.

Audit policies are treated separately from other policy types. For example, they are not listed when you use the `beectl list_policies` command.

For full details about creating and managing auditing policies, see Chapter 14, "Managing Oracle Beehive Auditing".

Creating and Managing Custom Policies

In addition to the default policies, you can create custom policies of your own, to suit your organization’s needs. Generally, it is much easier to create and manage custom policies using Oracle Beekeeper. In this section, procedures for creating and managing custom policies are presented using the `beectl` command-line tool. The *Oracle Beekeeper Online Help* includes instructions and assistance with creating and managing policies.

See Also:

- "Policy Definitions" in Chapter 1, "Oracle Beehive XML File Reference" of the *Oracle Beehive Administrator's Reference Guide*
- "Managing Policies" in Chapter 3, "Managing Enterprises" of the *Oracle Beekeeper Online Help*

This section contains the following topics:

- Creating New Policies
- Editing Existing Policies
- Deleting Custom Policies

Creating New Policies

There are two types of actions: Java and PLSQL. The default provisioning, deprovisioning, password, auditing, and record management policies make use of PLSQL actions which are pre-defined in the system. When you modify any of the default policies, or create your own provisioning, deprovisioning, auditing, or record management policies, you do not need to make any changes to the actions.

You cannot create custom actions. You can create custom policies using the pre-existing actions.

To create a new policy using beectl, upload an XML formatted policy document using the `beectl add_policy` command:

```
beectl> add_policy --file <full path to the policy xml file>
```

You can also create and use policy templates. A policy template defines a set of rules containing conditions, and an action, but does not specify the actor or scope for the triggering event of the policy. For example, you could create a policy template that is triggered when a document in a workspace is deleted. You could then later apply this policy template by using it to create a policy applying it to a specific organization, or for a specific user's documents.

Policy templates may make use of schemas. Policy Schemas are pre-defined sets of rule and attribute definitions. When you use a schema to create a template, you are combining the schema's rules with actions defined in the template.

Oracle Beehive comes with two default schemas: the Audit schema and the Records Management schema. Additionally, you can create your own schemas: new Audit or Records Management schemas, or General schemas (for all other uses).

Creating New Policy Templates

To create a new policy template using beectl, create a template XML file and import it using the `beectl add_policy_template` command:

```
beectl> add_policy_template --file <full path to the policy template xml file>
```

An example policy template may be found in your Oracle Beehive installation at `$ORACLE_HOME/beehive/templates/policy/TestAddPolicyTemplate.xml`.

Creating New Policy Schemas To create a new policy schema using beectl, create a schema XML file and import it using the beectl add_policy_schema command:

```
beectl> add_policy_schema --file <full path to the policy schema xml file>
```

An example policy schema may be found in your Oracle Beehive installation at `$ORACLE_HOME/beehive/templates/policy/TestAddPolicySchema.xml`.

Creating New Policies

To create a new policy based on a policy template using beectl, use the `beectl add_policy` command, referring to an XML-formatted file containing the policy information:

```
beectl> add_policy --file <full path to the policy xml file>
```

Example policies may be found in your Oracle Beehive installation at `$ORACLE_HOME/beehive/templates/policy`.

The following is an example of a policy file. To specify a template to use for a policy, put the template identifier into the template element tag in the policy XML file:

```
<?xml version="1.0" encoding="UTF-8" ?>
<PolicyInfo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xsi:noNamespaceSchemaLocation="policy_xml/policyInfo.xsd">
  <policy></policy>
  <scope></scope>
  <template></template>
  <name>FooAddPolicyTest_simple</name>
  <description>This is a test.</description>
...
```

Editing Existing Policies

You can edit any existing policy. First, locate the policy and export it to an XML file. Then, edit the file, and import it back to the system to modify the existing policy.

To list policies, use the `beectl list_policies` command:

```
beectl> list_policies --policy_name <name of the policy to be listed> --scope
<container where the policy is deployed> --children <whether to include policies
defined at child containers> --enforced <whether to include enforced policies>
```

To export a policy, use the `beectl export_policy` command:

```
beectl> export_policy --policy_name <name of the policy to download> --scope
<identifier of container where the policy is defined> --destination <destination
directory>
```

After editing the policy XML, use the `beectl modify_policy` command to import your changes:

```
beectl> modify_policy --file <full path to the policy xml file>
```

Deleting Custom Policies

You can delete an existing policy. You should never delete the provisioning, deprovisioning, or password policies.

If you delete a policy that prevents some action that you are prohibiting users from performing, the users will be able to perform that action. Thus, there is a potential security risk whenever you delete a policy. Use caution.

To delete a policy, use the `beectl delete_policy` command:

```
beectl> delete_policy --policy <id or name of the policy to be deleted>
```

You can also delete a custom policy template you have created. If the policy template is currently in use by any policy, when you attempt to delete it you will see an error message.

To delete a policy template, use the `beectl delete_policy_template` command:

```
beectl> delete_policy_template --policy_template <id or name of the policy
template to be deleted>
```

Using Dynamic Policy Attributes

Beginning in Oracle Beehive 1.5, you can use dynamic attributes in your policies. This allows you to get data from the payload of a triggering event and use it in the policy. For example, if you built a policy template that triggers when a new document is created in a workspace, you could get the document name and use it in the generated task title.

To use a dynamic attribute, include XML in your policy (or policy template) similar to the following:

```
  <Attributes>
   <attribute>
     <name>attribute1</name>
     <defaultValue>attribute1 default value</defaultValue>
     <prompted>true</prompted>
     <required>false</required>
 </attribute>
.
.
.
  <attribute>
   <name>attribute n</name>
   <defaultValue>attribute n default</defaultValue>
   <prompted>true</prompted>
   <required>false</required>
 </attribute>
</Attributes>
```

You can include any number of attribute elements.

Dynamic policy attributes are listed in "Dynamic Policy Attributes," in Chapter 1, "Oracle Beehive XML File Reference" of the *Oracle Beehive Administrator's Reference Guide*, along with an example. Attribute names must be from either the dynamic policy attribute list in that section, or the list of attributes for the given event that will trigger the policy. You can list all of the attributes of an event by using the `beectl list_events` command with the `--internal_name` option. For example:

```
beectl> list_events --internal_name WORKSPACE_UPDATED
```

12 Oracle Beehive Client Customization

This module describes the customization capabilities of the Oracle Beehive clients: Oracle Beehive Central and Oracle Beehive Zimbra.

This module includes the following sections:

- Customizing Oracle Beehive JavaSSO Login Page
- Customizing Oracle Beehive Central
- Customizing Oracle Beehive Webmail

Customizing Oracle Beehive JavaSSO Login Page

This section decribes the steps to customize the Oracle Beehive JavaSSO login page to display a footer that can be used as a privacy or legal disclaimer on the login page.

To customize the Oracle Beehive JavaSSO login page:

1. In your Oracle Beehive Oracle Home, create the following directory:

   ```
   $ORACLE_HOME/langpack/xliff/custom/
   ```

2. Create a file `oracle.ocs.authentication.service.AuthServiceEndUserResourceBundle.xlf` in the `langpack/xliff/custom` directory.

   The file created should be an `xliff` file containing only the translation token OCSSSO-00003, as shown in the following example:

   ```
   <?xml version="1.0" encoding="UTF-8"?>
   <xliff version='1.1' xmlns='urn:oasis:names:tc:xliff:document:1.1'>
   <file
   original='oracle.ocs.authentication.service.AuthServiceEndUserResourceBundle'
   source-language='en-us' datatype='beehive'>
   <header>
     <prop-group name="ora_reconstruction">
        <prop prop-type="beehive-version">0.0.0.0</prop>
        <prop prop-type="beehive-filetype">custom</prop>
        <prop prop-type="beehive-scope">enduser-webui</prop>
     </prop-group>
   </header>
   <body>
     <trans-unit id='OCSSSO-00003'>
     <source>Insert customer legal disclaimer or privacy statement here</source>
     </trans-unit>
   </body>
   </file>
   </xliff>
   ```

3. Add any other translations for additional languages in the `langpack/xliff/custom` directory. For example, the French translation file would be `oracle.ocs.authentication.service.AuthServiceEndUserResourceBundle_fr.xlf`.

4. Use the following command to create a custom language pack (in the form of a `.jar` file) containing the `xlf` file created in step 2. This command creates a `zip` file that includes all the files in the `langpack` directory and its subdirectories.

   ```
   jar cvf javasso-disclaimer.jar langpack
   ```

5. Run the following two commands to upload the language pack.

   ```
   $ORACLE_HOME/beehive/bin/beectl upload_language_pack --source
   /var/tmp/javasso-disclaimer.jar
   $ORACLE_HOME/beehive/bin/beectl activate_configuration
   ```

6. Then, on every midtier where JavaSSO is running, run the following command:

   ```
   $ORACLE_HOME/beehive/bin/beectl modify_local_configuration_files --restart_
   needed false
   ```

7. Restart the **BEEMGMT** component service.

Customizing Oracle Beehive Central

This section describes the ways in which Oracle Beehive administrators can customize Oracle Beehive Central for their users. This section includes the following topics:

- Introduction to Customizing Oracle Beehive Central
- Customizing the Oracle Beehive Central Public Home Page
- Customizing the Oracle Beehive Central Private Home Page
- Customizing the Oracle Beehive Central Help Page
- Customizing Oracle Beehive Central Download Center

Introduction to Customizing Oracle Beehive Central

Oracle Beehive Central provides Web-based single-point access to efficiently manage your Beehive delegation and sharing privileges, set preferences, download supported Beehive clients, and access other Oracle Web-based clients.

Oracle Beehive Central enables you to manage deployment-specific content on the Public, and Private Home pages, the Help page, and the Download Center areas.

The customizable files for the Public and Private Home pages as well as the Help page are available at the following location:

```
$ORACLE_HOME/j2ee/BEEAPP/applications/bcentral/bcentral-web/custom
```

The customization files use the following directory structure:

```
custom
|
|- - - en
         |
         |- - -publichome.html.template
         |- - -privatehome.html.template
```

```
              |- - -help.html.template
```

Note: Oracle recommends that you follow the template provided in the file structure without using any additional graphics, decoration or fixed length components while making customizations. It is also recommended that you adhere to the user interface (UI) look and feel of Oracle Beehive Central.

The customization files used by the Download Center are describes later in the "Customizing Oracle Beehive Central Download Center" section.

For information about Security considerations for Oracle Beehive Central, refer to "Security Considerations for Oracle Beehive Central" in the *Oracle Beehive Deployment Guide*.

Customizing the Oracle Beehive Central Public Home Page

The Oracle Beehive Central public home page contains general information about Beehive Central with links to other Beehive clients. For example, you may add additional information with links to your own IT portal.

You can customize the Introduction and Download sections of public home pages.

To customize the Oracle Beehive Central public home page:

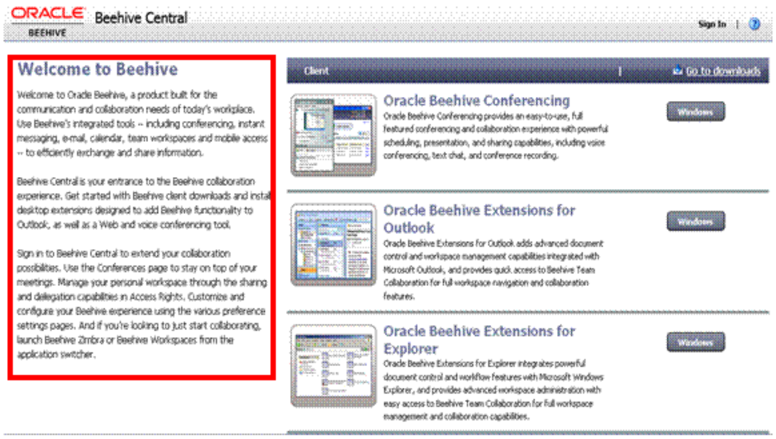
1. Edit the `publichome.html.template` file in a text editor.
2. Rename the `publichome.html.template` file to `publichome.html`.
3. Save and exit the `publichome.html` file.

If you want to extend the existing content, first retrieve it and then edit it as required. You can apply one of the following methods to retrieve existing content:

Using the Beehive Central Public Home page

1. Using a Web browser, access the Beehive Central public home page. This page is shown in Figure 12–1.

Figure 12–1 Oracle Beehive Central Public Home Page - Customizable Area



Welcome to Beehive

Welcome to Oracle Beehive, a product built for the communication and collaboration needs of today's workplace. Use Beehive's integrated tools -- including conferencing, instant messaging, e-mail, calendar, team workspaces and mobile access -- to efficiently exchange and share information.

Beehive Central is your entrance to the Beehive collaboration experience. Get started with Beehive client downloads and instal desktop extensions designed to add Beehive functionality to Outlook, as well as a Web and voice conferencing tool.

Sign in to Beehive Central to extend your collaboration possibilities. Use the Conferences page to stay on top of your meetings. Manage your personal workspace through the sharing and delegation capabilities in Access Rights. Customize and configure your Beehive experience using the various preference settings pages. And if you're looking to just start collaborating, launch Beehive Zimbra or Beehive Workspaces from the application switcher.

Oracle Beehive Conferencing

Oracle Beehive Conferencing provides an easy-to-use, full featured conferencing and collaboration experience with powerful scheduling, presentation, and sharing capabilities, including voice conferencing, text chat, and conference recording.

Oracle Beehive Extensions for Outlook

Oracle Beehive Extensions for Outlook adds advanced document control and workspace management capabilities integrated with Microsoft Outlook, and provides quick access to Beehive Team Collaboration for full workspace navigation and collaboration features.

Oracle Beehive Extensions for Explorer

Oracle Beehive Extensions for Explorer integrates powerful document control and workflow features with Microsoft Windows Explorer, and provides advanced workspace administration with easy access to Beehive Team Collaboration for full workspace management and collaboration capabilities.

2. Right-click any blank area of the Beehive Central public home page.

3. Select **View Source** from the shortcut menu.
4. Copy the contents of the file into an HTML editor.

Using the default_home_content.jsp File

1. Open the following file in an HTML editor:

   ```
   ORACLE_HOME/j2ee/BEEAPP/applications/bcentral/bcentral-web/default_
   custom/default_home_content.jsp
   ```

2. Copy the contents of this file into a new window in the HTML editor.
3. Replace all Java XLFF strings with their language equivalents.

Customizing the Oracle Beehive Central Private Home Page

The `privatehome.html.template` file contains the information about the customizable part of the private home page. For example, you may add additional information about service offered by your enterprise.

To customize the Oracle Beehive Central private home page:

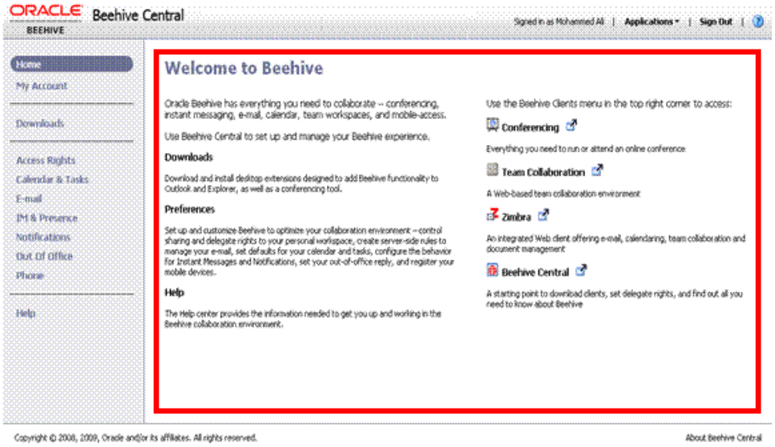
1. Edit the `privatehome.html.template` file in a text editor.
2. Rename the `privatehome.html.template` file to `privatehome.html`.
3. Save and exit the `privatehome.html` file.

If you want to extend the existing content, first retrieve it and then edit it as required. You can apply one of the following methods to retrieve existing content:

Using the Beehive Central Private Home page

1. Using a Web browser, access the Beehive Central private home page. This page is shown in Figure 12–2.

Figure 12–2 Oracle Beehive Central Private Home Page - Customizable Area



Welcome to Beehive

Oracle Beehive has everything you need to collaborate -- conferencing, instant messaging, e-mail, calendar, team workspaces, and mobile-access.

Use Beehive Central to set up and manage your Beehive experience.

Downloads

Download and install desktop extensions designed to add Beehive functionality to Outlook and Explorer, as well as a conferencing tool.

Preferences

Set up and customize Beehive to optimize your collaboration environment -- control sharing and delegate rights to your personal workspace, create server-side rules to manage your e-mail, set defaults for your calendar and tasks, configure the behavior for Instant Messages and Notifications, set your out-of-office reply, and register your mobile devices.

Help

The Help center provides the information needed to get you up and working in the Beehive collaboration environment.

Use the Beehive Clients menu in the top right corner to access:

Conferencing

Everything you need to run or attend an online conference

Team Collaboration

A Web-based team collaboration environment

Zimbra

An integrated Web client offering e-mail, calendaring, team collaboration and document management

Beehive Central

A starting point to download clients, set delegate rights, and find out all you need to know about Beehive

Copyright © 2008, 2009, Oracle and/or its affiliates. All rights reserved.

2. Right-click any blank area of the Beehive Central private home page.
3. Select **View Source** from the shortcut menu.
4. Copy the contents of the file into an HTML editor.

Using the default_personal_home_content.jsp File

1. Open the following file in an HTML editor:

ORACLE_HOME/j2ee/BEEAPP/applications/bcentral/bcentral-web/default_custom/default_personal_home_content.jsp

2. Copy the contents of this file into a new window in the HTML editor.
3. Replace all Java XLFF strings with their language equivalents.

Customizing the Oracle Beehive Central Help Page

You can access the Oracle Beehive Central Help pages by clicking **Help** from the Beehive Central navigation menu. The Oracle Beehive Help page contains links to online help pages hosted by the Oracle Technology Network (OTN). You can replace the default Help page with custom help pages prepared for your deployment.

The `help.html.template` file contains the information about the customizable part of the Help page.

To customize the Oracle Beehive Central Help page:

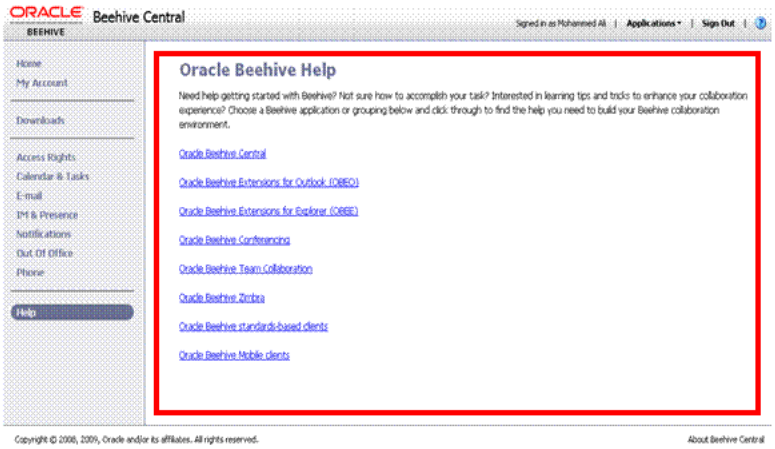
1. Edit the `help.html.template` file in a text editor.
2. Rename the `help.html.template` file to `help.html` file.
3. Save and exit the `help.html` file.

If you want to extend the existing content, first retrieve it and then edit it as required. You can apply one of the following methods to retrieve existing content:

Using the Beehive Central Help page

1. Using a Web browser, access the Beehive Central Help page. This page is shown in Figure 12–3.

Figure 12–3 Oracle Beehive Central Help Page - Customizable Area



Oracle Beehive Help

Need help getting started with Beehive? Not sure how to accomplish your task? Interested in learning tips and tricks to enhance your collaboration experience? Choose a Beehive application or grouping below and click through to find the help you need to build your Beehive collaboration environment.

Oracle Beehive Central

Oracle Beehive Extensions for Outlook (OBEO)

Oracle Beehive Extensions for Explorer (OBEE)

Oracle Beehive Conferencing

Oracle Beehive Team Collaboration

Oracle Beehive Zimbra

Oracle Beehive standards-based clients

Oracle Beehive Mobile clients

Copyright © 2008, 2009, Oracle and/or its affiliates. All rights reserved.

About Beehive Central

2. Right-click any blank area of the Beehive Central Help page.
3. Select **View Source** from the shortcut menu.
4. Copy the contents of the file into an HTML editor.

Using the default_help_content.jsp File

1. Open the following file in an HTML editor:

ORACLE_HOME/j2ee/BEEAPP/applications/bcentral/bcentral-web/default_custom/default_help_content.jsp

2. Copy the contents of this file into a new window in the HTML editor.
3. Replace all Java XLFF strings with their language equivalents.

Customizing Oracle Beehive Central Download Center

The Oracle Beehive Central Download Center is the central place for Oracle Beehive users to download various Oracle Beehive client applications such as Oracle Beehive conferencing client, and Oracle Beehive Extension for Outlook.

The Oracle Beehive Central Download Center exposes applications available from the Oracle Beehive Device Management (DM) Application Repository. You can extend the list of downloadable applications and control the set of applications available to end users based on their enterprise and organization memberships.

This section contains the following topics:

- Oracle Beehive Central Download Center Concepts
- Setting Up a Tightly Controlled Environment
- Setting Up an Open Environment
- Enabling and Disabling Oracle Beehive Central Download Center
- Removing Applications from Oracle Beehive Central Download Center
- Adding a new Application to Oracle Beehive Central Download Center
- Customizing the Application List Templates
- Customizing the Download Landing Page Templates
- Customizing the Oracle Beehive Central Individual Client Pages
- Packaging Applications for Upload to Device Management Repository
- Configuring Oracle Beehive Central Download Center

Oracle Beehive Central Download Center Concepts

This section describes the concepts that you should know to manage the Oracle Beehive Central Download Center.

This section includes the following topics:

- DM Application Repository
- Client Application
- Bootstrap Application
- Third-Party Application
- Platform Application
- Tightly Controlled Environments
- Open Environments
- Provisioning

DM Application Repository

The DM Application Repository is part of the Device Management Service and is the central location where the client applications are stored. The DM Application Repository can be managed through Oracle Beekeeper or `beectl`. This includes uploading new applications, deleting applications, managing installable application configurations, and provisioning.

Client Application

You use a client application to work with Oracle Beehive. Oracle Beehive Conferencing, Oracle Beehive Extension for Outlook (OBEO), and Oracle Beehive Extension for Explorer (OBEE) are examples of client applications. A client application is characterized by a version, patch set number, and platform characteristics, such as the certified device class, operating system, and processor.

Bootstrap Application

A bootstrap application is an application that is used to install a client application directly or download the correct version from the DM Repository prior to installing it.

Third-Party Application

A third-party application is a client application that can be stored in and offered for download from the Oracle Beehive DM Application Repository (for example, open source clients such as Mozilla Thunderbird, and Mozilla Sunbird). Third-party applications are characterized by platform characteristics (device-class, operating system, and processor). Unlike other client-applications, third-party applications are not subject to provisioning. That is, third-party applications are always offered for download to anonymous users.

Platform Application

A platform application is a client application with the `platform` flag set. Platform applications are libraries or applications required by one or more client applications.

Tightly Controlled Environments

Some enterprises may have may have their own Desktop Management System. In a tightly controlled environment such as this, you (Administrator) can limit the available downloads to some form of Desktop Management Console or hide the Download Center.

Open Environments

For enterprises that do not have their own Desktop Management System, the list of application available from the Oracle Beehive Central Download Center can be expanded to include any application the enterprise would like to make available to its users.

All provisioned client applications available in the DM Application Repository are available for download to the end user. The Oracle Beehive administrators can extend the list of client applications by uploading new applications to the DM Repository and provisioning the applications to all or some organizations or enterprises.

Provisioning

Provisioning is the process of defining which applications are available to which enterprises or organizations. If a client application is provisioned to an enterprise, then the application is available to all users (including anonymous users). Applications provisioned to an organization are available only to users of that organization who are logged in.

Setting Up a Tightly Controlled Environment

To set up a tightly controlled environment, apply one of the following approaches:

- If you want to use your own desktop management system, then you can hide Oracle Beehive Central Download Center. For instructions on how to hide Oracle Beehive Central Download Center, see "Enabling and Disabling Oracle Beehive Central Download Center".
- Hide the applications that Oracle Beehive Central Download Center offers by default. Offer users one download which is some form of custom desktop management console for use with your desktop management system. The Download center then becomes another place where your users can get started. See "Removing Applications from Oracle Beehive Central Download Center" for information about removing the default applications. Then, proceed to "Adding a new Application to Oracle Beehive Central Download Center" to make your desktop management console available to the users.

Setting Up an Open Environment

In an open environment, you can add applications such as Mozilla Thunderbird and Pidgin to Oracle Beehive Download Center. See "Adding a new Application to Oracle Beehive Central Download Center" for instructions.

In addition, you can customize the content displayed for the default applications. See "Customizing the Oracle Beehive Central Individual Client Pages" for instructions.

Enabling and Disabling Oracle Beehive Central Download Center

One of the ways to support a tightly controlled enterprise scenario is by hiding the Oracle Beehive Central Download Center altogether with the configuration property `EnableDownlodCenter`. You can modify this setting through either Oracle Beekeeper or the `beectl` command line utility.

To disable the Oracle Beehive Download Center using Oracle Beekeeper:

1. Log on to Oracle Beekeeper.
2. Click **BeeCentral** under Services.
3. In the Beehive Central window, click the **Configuration** tab.

   The active configuration is displayed and the EnableDownloadCenter option is selected.
4. Click **Edit** to modify the existing configuration values.
5. Set the value to `False`.
6. To activate the new settings, click **Configuration Control** and then click **Activate**.

To enable or disable the Oracle Beehive Download Center using the `beectl` command line utility, use the following commands:

```
beectl modify_property --name EnableDownloadCenter
                       --component _BeeCentralService
                       --value false
beectl activate_configuration
```

After a short delay, Oracle Beehive Central picks up the configuration change. Therefore, restarting your computer is not required. The Download Center link will no longer be available and Download Center information in the public and private home pages will also disappear.

Removing Applications from Oracle Beehive Central Download Center

The client applications are displayed in a list in Oracle Beehive Central Download Center.

You can remove an application from this list by using Oracle Beekeeper to remove the installable package of the application from Oracle Beehive DM Application Repository.

To remove an application from Oracle Beehive Download Center:

1. Log in to Oracle Beekeeper as `beeadmin` or a user with privileges to access the Client Applications module.
2. Under System on the left navigation panel, open or expand the Client Applications module.
3. Expand the node for the application that you want to remove. For example, expand the node for Oracle Beehive Conferencing Bootstrap.
4. Select the latest version or patch set for the application.
5. On the Detail pane, click the **Provisioning** tab.
6. Select the enterprise.
7. On the tool bar, click **Remove**.
8. In the upper-right area of the Detail pane, click **Apply**.

Adding a new Application to Oracle Beehive Central Download Center

In the Oracle Beehive Central Download Center, the available client applications are shown in a list.

You can configure the content and layout of the items in the list through the templates provided in the Oracle Beehive Central deployment. When no template is defined for an application, a default template is used. The default template displays the following information:

- Application name as defined in the DM Application Repository
- Application platform information, which is the target operating system
- Application version
- Description as defined in the DM Application Repository

The contents of the application listings in the list view and the download landing page can be customized through templates. There are two templates, one for displaying the list and one for the download landing page. The templates are localized and are provided for each language that is supported. If there is no template provided in the user's language, then the English template is used by default. If no template is provided at all, then a default template to display the minimal application characteristics, such as name, platform details, processor details, operating system, and the version, is used.

The templates are located in the `web-deployment` folder in the Beehive Application Tier where Oracle Beehive Central is deployed. The file location is:

```
$ORACLE_HOME/j2ee/<BEEAPP>/applications/bcentral/bcentral-web/custom
```

In the preceding example, *`$ORACLE_HOME`* represents the file location of the Beehive installation and `CNT_NAME` represents the Web container name in which Oracle Beehive Central is deployed (typically, `BEEAPP`). This folder contains sub-folders for the supported locales. The sub-folders are named with the ISO 639 country code of the locale.

The templates use the following file naming conventions:

```
<loc>/<BeehiveAppId>.list.html
<loc>/<BeehiveAppId>.details.html
```

In the preceding example, <loc> stands for the locale-specific folder and <BeehiveAppId> is a unique identifier for the downloadable application defined in the application configuration properties in Oracle Beekeeper. The templates that have names ending with .list.html are used to render a table-row in the list view. The templates that have names ending with .details.html are used for the download landing page.

Each installable application can define its own BeehiveAppId parameter.

Customizing the Application List Templates

The application list templates have a file name of the form <BeehiveAppId>.list.html and are used to render the first two columns of a table-row in the application list.

The template contains two <td> tags for the first and second columns of the download list. The first column is used to display an application thumbnail. The image is 130 pixels wide and 111 pixels high.

In this folder location, *$ORACLE_HOME* refers to the installation location of the Beehive Application Tier to which the Oracle Beehive Central is deployed, <CNT_NAME> refers to the name of the Web container (BEECLIENT or BEEAPP), and <BUILD> refers to the build number of Oracle Beehive Central.

The first two columns of the application list table consume 450 pixels horizontally. To reference any pictures in the download-images folder, use the following template variable:

```
%DOWNLOAD_IMGS%
```

A custom template to display an entry for the Mozilla Thunderbird application in the English locale appears as follows:

```
<!-- image column -->
<td class="downloadHorizontalLine" style="vertical-align: top;">
    <img src="%DOWNLOAD_IMGS%/btn_thunderbird.png" alt="Download Thunderbird">
</td>

<!-- description column -->
<td class="downloadHorizontalLine" style="vertical-align: top;">
    <span class="downloadTableRowHeader">
        Thunderbird for Windows
    </span>
    <br/>
    <span class="downloadTableRowDescription">
Thunderbird for windows is a powerful email-application and seamlessly
works with Oracle Beehive.
    </span>
    <br/>
</td>
```

Customizing the Download Landing Page Templates

The templates for the download landing page have a file name of the form `<BeehiveAppId>.details.html`. These templates are used to render the customizable area of the download landing page.

The image of the download landing page must consume 256 pixels horizontally and the text-area should be 300 pixels high.

To refer to images stored in the download-images folder, use the template variable `%DOWNLOAD_IMGS%` as described in the previous section.

The following is an example of a generic template to show application-specific details on the download landing page:

```
<div style="float:left;width:256px;margin-right:30px">
<img src="%DOWNLOAD_IMGS%/download1.png">
</div>

div style="float:left;width:300px;">
<div class="pageTitle">
Application Name
</div>
<br/>

<span id="detailedDownloadPage|data|overviewTitle">
<b>Overview</b>
</span>
<br/>
<span id="detailedDownloadPage|data|overview">
Synopsis of the application
<br/>
</span>
<br/>
<span id="detailedDownloadPage|data|installationTitle">
<b>Installation</b>
</span>
<br/>
<span id="detailedDownloadPage|data|installation">
Detailed installation instructions
<br/>
</span>
<br/>
<span id="detailedDownloadPage|data|configTitle">
<b>Configuration</b>
</span>
<br/>
<span id="detailedDownloadPage|data|configuration">
Detailed configuration instructions
<br/>
</span>
<br/>
<span id="detailedDownloadPage|data|sysReqTitle">
<b>System Requirements</b>
</span>
<br/>
<span id="detailedDownloadPage|data|systemReq">
detailed system requirements (hardware / software)
<br/>
</span>
</div>
```

Customizing the Oracle Beehive Central Individual Client Pages

You can customize the following individual client pages:

- Oracle Beehive Conferencing Client Windows

  The `BeehiveAppId` of this client page is `confclientwin`.

- Oracle Beehive Conferencing Plug-In for Windows Media-Player

  The `BeehiveAppId` of this client page is `confcodecwin`.

  The `BeehiveAppId` of this client page is

- Oracle Beehive Conferencing Client for Mac

  The `BeehiveAppId` of this client page is `confclientmac`.

- Oracle Beehive Extension for Windows Explorer

  The `BeehiveAppId` of this client page is `obeewin`.

- Oracle Beehive Extension for Microsoft Outlook

  The `BeehiveAppId` of this client page is `obeowin`.

Customizing the Oracle Beehive Central individual client pages is similar to customizing Application List template and the Download Landing Page template described in "Customizing the Application List Templates" and "Customizing the Download Landing Page Templates". You can customize the `<BeehiveAppId>.list.html` and `<BeehiveAppId>.details.html` templates files for each of the individual client pages.

Packaging Applications for Upload to Device Management Repository

To upload a custom application to the device-management repository, you must create a Beehive-specific application archive (a ZIP file).

For example, a Beehive customer who wants to add the Mozilla Thunderbird application for Windows to the device management repository can use an application archive, which is a normal ZIP file. This ZIP file contains the installation binary (for example, Thunderbird Setup 2.0.0.21.exe) and an XML file with additional information. The XML file is called metadata.xml and includes following content:

```
<?xml version="1.0" encoding="UTF-8"?>
<application>
       <property>
              <name>Thunderbird 2 for Windows</name>
              <description>Thunderbird 2.0.0.21</description>
              <os>windows</os>
              <processor>x86</processor>
              <deviceclass>desktop</deviceclass>
              <language>en</language>
              <version>2.0.0.21</version>
              <versionnumber>2</versionnumber>
              <patchsetnumber>0</patchsetnumber>
              <vendor>Mozilla</vendor>
             <application_type>THIRDPARTY</application_type>
        </property>
        <modules>
                <module>
                       <name>Thunderbird Setup 2.0.0.21.exe</name>
                       <src>.</src>
                       <dest>/</dest>
                       <contenttype>application/octet-stream</contenttype>
                </module>
        </modules>
        <configuration>
              <param name="VIRTUAL_HOST" value="beehive.oracle.com"/>
              <param name="VIRTUAL_PORT" value="80" />
              <param name="VIRTUAL_SSL" value="false"/>
              <param name="BTI_PORT" value="1532"/>
              <param name="BTI_SSL" value="false"/>
              <param name="FILE_NAME" value="thunderbird-setup.exe"/>
              <param name="BeehiveAppId" value="thunderbirdwin"/>
        </configuration>
</application>
```

The `metadata.xml` file contains the following three sections:

- Property Section
- Modules Section
- Configuration Section

Property Section

The Property section contains various application-specific attributes, such as application version, platform, supported processor, and languages.

The value of the <os> element is used to match the application to the operating system of the user's computer. The permitted values for <os> element are: windows, mac and linux (all lowercase).

The <application_type> element specifies the type of the application. Supported values for the <application_type> element are `BOOTSTRAP`, `REGULAR`, `PLATFORM` or `THIRDPARTY`. When the type is set to `BOOTSTRAP`, a matching application (with regards to device class, operating system, and processor) with type `REGULAR` or `PLATFORM` must be present in order to list the bootstrap application in the Oracle Beehive Central Download Center.

Modules Section

The content of the application archive is listed in the Modules section. Every file in the archive (except the `metadata.xml` file itself) has a module entry defining the file name, MIME type, and other attributes.

Configuration Section

In the Configuration section, the `BeehiveAppId` parameter is defined to select a custom template that will render the application. Furthermore, installation specific parameters, such as `FILE_NAME`, `VIRTUAL_HOST`, `VIRTUAL_PORT`, `VIRTUAL_SSL`, `BTI_PORT`, and `BTI_SSL` are defined.

You can edit the values of the configuration parameters in Oracle Beekeeper when an installable client application is created from the uploaded application.

To create an application archive:

1. Add the installation binary file and the metadata.xml files without any sub-directories to a new ZIP archive.
2. Add and customize the application list and download landing page templates in the directory for your language.
    - switch to `$ORACLE_HOME/j2ee/BEEAPP/applications/bcentral/bcentral-web/custom`

`/en` for English or to another language directory. Prefix the templates with the BeehiveAppId parameter value. You can copy an existing template and make the necessary modifications. For example, to create an application archive for the pidgin application, copy `obeewin.details.html` and `obeewin.list.html` and rename them to `pidgin.details.html` and `pidgin.list.html` respectively.

- In both files `pidgin.details.html` and `pidgin.list.html`, append your icon filename to the `%DOWNLOAD_IMGS%` variable.

  ```
  img src="%DOWNLOAD_IMGS%/pidgin_logo.png
  ```
- Icons are stored under the `%DOWNLOAD_IMGS%` template variable. The location of this variable is stored in the `$ORACLE_HOME/j2ee/BEEAPP/applications/bcentral/bcentral-web/WEB-INF/web.xml` file. Use the following command to grep for the `download-images` folder value.

  ```
  egrep download_imgs $ORACLE_
  HOME/j2ee/BEEAPP/applications/bcentral/bcentral-web/WEB-INF/web.xml
  ...
  <param-value>skins/17379/default/download_imgs</param-value>
  <url-pattern>/skins/17379/default/download_imgs/*</url-pattern>
  ```
  For example, for the pidgin application, copy `pidgin_logo.png` to `$ORACLE_HOME/j2ee/BEEAPP/applications/bcentral/bcentral-web/skins/17379/default/download_imgs`

  See the sections "Customizing the Application List Templates" and "Customizing the Download Landing Page Templates" for more information.

3. Upload the archive file to the device management repository either through Oracle Beekeeper or through the command-line configuration tool `beectl`.

   Table 12–1 lists the configuration parameters supported by Oracle Beehive Central Download Center.

Table 12–1 Configuration Parameters Supported by Oracle Beehive Central Download Center

Parameter	Description
`BeehiveAppId`	Value used to select a template to render either a list or detail view.
`FILE_NAME`	When present, the value is used to name the downloaded file. When missing, the module name (as defined in module section of `metadata.xml`) is used instead. The `FILE_NAME` parameter can refer to any of the values of the other properties (`VIRTUAL_HOST`, `VIRTUAL_SERVER`, `VIRTUAL_SSL`, `BTI_PORT` and `BTI_SSL`) by using the pattern `%PROPERTY_NAME%`. For instance, if `VIRTUAL_HOST` is set to `beehive.oracle.com` and the `FILE_NAME` parameter is defined as `AppInstaller-%VIRTUAL_HOST%.exe`, then the resulting file name is resolved to `AppInstaller-beehive.oracle.com.exe`.
`VIRTUAL_HOST`	The host name of the Beehive installation (for example, DNS name of load-balancer). When missing, but used in `FILE_NAME`, the value is inferred from the current request.
`VIRTUAL_PORT`	The port of the Beehive installation. When missing, but used in `FILE_NAME`, the value is inferred from the current request.
`VIRTUAL_SSL`	Set to True or False to indicate whether to generate HTTPS / HTTP in resulting file name. When missing, but used in `FILE_NAME`, the value is inferred from the current request.

Table 12–1 (Cont.) Configuration Parameters Supported by Oracle Beehive Central Download Center

Parameter	Description
BTI_PORT	The port of the BTI. When missing, but used in FILE_NAME, the value is read from the current system configuration.
BTI_SSL	Set to True or False to indicate whether to generate SLL / NOSSL in resulting file name. When missing, but used in FILE_NAME, the value is read from the current system configuration.

Upload the client application .ZIP file to the repository using the following command:

```
beectl upload_client_application --file /refresh/home/scratch/<application
name>.zip
```

The following is an example of the result that is displayed if the pidgin application is uploaded successfully:

```
Successfully uploaded the client binary file /refresh/home/scratch/pidgin.zip
to repository. The identifier of the resulting patchset is
4CFD:1802:capp:898B2EC2C2ED0750E040AE0A3AE843970000001293A9.
```

4. Verify that the application is uploaded successfully to the repository using the folowing `beectl` command:

```
beectl list_client_applications
```

The following is an example of the result that is displayed when the pidgin application is uploaded to the repository:

```
pidgin
+-------------------------------------------------------+-----------------
| pidgin | pidgin
+---+-----------+----------+--------------------------+-----------------
| Windows_NT | x86 | desktop
+---+-----------+----------+-------------------------------------------
| 4CFD:1802:capn:898B2EC2C2ED0750E040AE0A3AE843970000001293A7
+-----------------------------------------------------------------------
```

5. Add the download privileges for the application.

- Go to `Oracle Beekeeper`, then `Client Applications`, and then `<application-name>`.
- Click **New**. The `New Installation Client Application` dialog appears. In the dialog, click **Ok** to create a new row `Latest version and patch set for <application-name>`.

You can also use the following command to create a client application configuration object:

```
beectl add_client_application_configuration --file <path to the application>
```

6. Once the application ZIP file is uploaded successfully to the repository, and the application is listed, you must provision the application to the community.

- Select `Latest version and patch set for <application-name>`(created in previous step) in the list view.
- In the bottom panel, go to the Provisioning tab, and click **Add**. The Enterprise dialog appears.
- In the dialog box, search for your enterprise name and add it to the right hand side table. Click **Ok**.
- Click **Apply** in the bottom panel.

You can also use the following command to provision the application to the community:

```
beectl add_client_application_provisioning --community enpr=<enterprise name>
--application 4CFD:1802:capn:898B2EC2C2ED0750E040AE0A3AE843970000001293A7
```

The following result is displayed:

```
Successfully provisioned client applications to the community.
```

Configuring Oracle Beehive Central Download Center

In addition to the system-management property `EnableDownloadCenter`, which is used to enable or disable the download center (as described in "Enabling and Disabling Oracle Beehive Central Download Center"), Table 12–2 lists the configuration properties that are defined in the deployment descriptor of Oracle Beehive Central.

The deployment descriptor file is located at:

```
$ORACLE_HOME/j2ee/<CNTR>/applications/bcentral/bcentral-web/WEB-INF/web.xml
```

Table 12–2 Configuration Properties Defined in the Deployment Descriptor of Oracle Beehive Central

Context Parameter	Description
`download.useragent.WIN32`	Regular expression used to match the user-agent HTTP header of the request to determine if the user's operating system is Windows (XP, Vista and 7). The default value is `.* Windows .*`.
`download.useragent.MAC`	Regular expression used to match the user-agent HTTP header of the request to determine if the user's operating system is Mac OS X. The default value is `.* Mac OS X.*`.
`download.useragent.LINUX`	Regular expression used to match the user-agent HTTP header of the request to determine if the user's operating system is Linux. The default value is `.* Linux .*`.
`download.applicationlist.cachetime`	This parameter specifies the life time (in milliseconds) of application listings in the public Download Center. Default value is 5 min = 300000. When a new application is uploaded to the DM Repository, you must wait for the time specified before the Beehive Central Download Center shows the new content.
`download.modulecache.location`	Location of the module cache. The module cache keeps a local copy of downloaded applications in the Beehive Application Tier of Oracle Beehive Central to minimize the number of streaming requests to the DM Repository. The file location is a path relative to the Oracle Beehive Central deployment. The default value is `downloads/modulecache`, referring to `$ORACLE_HOME/j2ee/<CNTR>/applications/bcentral/bcentral-web/downloads/modulecache`.

Table 12–2 (Cont.) Configuration Properties Defined in the Deployment Descriptor of Oracle Beehive Central

Context Parameter	Description
`download.modulecache.maxsize`	Size restriction of the module cache (in bytes). The module cache is an LRU cache with a size restriction. If the total size of cached files exceeds this number, files that were downloaded least recently are removed first from the cache. The default value is 250 MB (262144000). Caching of module files can be turned off completely by setting the value to 0.

Customizing Oracle Beehive Webmail

This section describes how to customize Oracle Beehive Webmail zimlets. This section includes the following topics:

- Introduction to Customizing Oracle Beehive Webmail Zimlets
- Customizing Oracle Beehive Webmail Zimlets

Introduction to Customizing Oracle Beehive Webmail Zimlets

Zimlets are useful widgets that enhance the functionality of Oracle Beehive Webmail. For example, zimlets enable Oracle Beehive Webmail users to view calendar data and add contact information to their address books directly from e-mail messages. Zimlets work by matching pre-defined string patterns in content, such as dates, addresses, and URLs in e-mail messages. Based on those matches, zimlets provide users with the appropriate features and options for maximal efficiency and convenience.

Customizing Oracle Beehive Webmail Zimlets

Oracle Beehive Webmail currently supports customization of the following zimlets:

- `com_zimbra_date`

  The com_zimbra_date zimlet activates Date string pattern matches in content to show calendar summaries and enable event creation.

- `com_zimbra_email`

  The `com_zimbra_email` zimlet recognizes e-mail address patterns and enables menus for activities such as replying and integration with Address Book.

- `com_zimbra_url`

  The com_zimbra_url zimlet recognizes URLs and Web locations and makes them navigable for users.

- `com_zimbra_dnd`

  The com_zimbra_dnd zimlet in conjunction with the Zimbra Drag and Drop Firefox extension (a separately available Firefox plug-in), enables users to attach files to e-mail by dragging the files from their local file system.

  For more information about the Zimbra Drag and Drop Firefox extension, refer to `http://www.zimbra.com`.

- `com_zimbra_bug`

The com_zimbra_bug zimlet enables quick links to an external bug tracking Web pages based on pattern recognition of strings such as `Bug 1234`.

These zimlets are customized by the `Preference` setting by using the following `beectl` command:

```
prfs=ZimbraDefaultCOS,enpr=...
```

For each of the zimlets, the `Preference` property is of type STRING with value `true` or `false`.

To disable the Oracle Beehive Webmail zimlets, assign the value `false` to the `Preference` property. Similarly, to enable Oracle Beehive Zimbra zimlets, assign the value `true` to the `Preference` property. This customization takes effect the next time the user logs in to the system. Note that restarting Oracle Beehive Services or the user's computer is not required for the customization to take effect.

13

Managing Oracle Beehive Access Control

Oracle Beehive uses a robust and highly-configurable model for controlling access to the various features of Oracle Beehive, and to the shared content stored by users, including files, folders, workspaces, calendars, and so forth.

During installation, a default setup is created providing a variety of pre-configured privileges, roles, and sensitivities. (Each of these special terms is defined in detail in the first section of this module.)

This module describes the various functions you can use, as an administrator, to customize how Oracle Beehive grants and limits access for your users. It contains the following topics:

- About Access Control
- Managing Privileges
- Managing Roles
- Creating and Managing Access Control Entities and Sensitivities

> **See Also:** With Oracle Beehive Version 2 (2.0.1.1) and later, you can use third-party single sign-on providers for user authentication. For more information, see "Integrating Third-Party Single Sign-On Providers with Oracle Beehive," in the *Oracle Beehive Integration Guide*.

About Access Control

In Oracle Beehive, you can use the `beectl` command line tool to manage and customize most aspects of users' access to services and stored objects. Throughout this module, you may find it convenient to refer to Module 2, "Oracle Beehive Command Line Utility" in *Oracle Beehive Administrator's Reference Guide* for syntax. All of the commands related to access control are categorized as "access control," and listed in the beginning of that module in Table 2-1. Alternatively, you can list the access control `beectl` commands, using the `beectl list_commands` command:

```
beectl> list_commands --category "access control"
```

In Oracle Beehive, there are two methods for controlling access to objects:

- Explicit Access Control: Access Control applied directly to controllable objects
- Implicit Access Control: Permissions granted or denied from users and groups

Explicit Access Control

Explicit access control is accomplished by creating "Access Control Entities (ACEs)", which are logical pairings of "Accessors" and "Access Privileges". An ACE may either grant or restrict any of five Access Types:

- READ: coded as +R or -R, this access type grants or revokes permission to open or read the object
- WRITE: coded as +W or -W, this access type grants or revokes permission to add, alter, or make changes to the object. Note that version control preserves previous versions, so WRITE access does not itself provide DELETE access
- DISCOVER: coded as +O or -O, this access type grants or revokes permission to see the object in lists, look it up in directories, find it in searches, and so forth. When DISCOVER privileges are revoked, a user cannot detect the existence of that object using any Oracle Beehive client process
- EXECUTE: coded as +E or -E, this access type grants or revokes permission to invoke or assign the object or cause it to perform its function
- DELETE: coded as +D or -D, this access type grants or revokes permission to delete the object

> **Note:** Not all access types have meaning for all entity types. For example, "delete" access is meaningless in the context of access to a service; "execute" access is meaningless in the context of access to a text file. Oracle Beehive ignores access type settings that are meaningless in a given context.

In an ACE, access types are referenced as `+-RWOED` (for example, "`RW-D`", or "`-D+RW`").

Any Oracle Beehive object may have any number of ACEs, collected into an Access Control List (ACL). ACLs are simply tables of ACEs, all of which apply to a single controllable entity.

Sensitivities

You may find that you need to grant a similar collection of ACEs repeatedly to various different objects. To facilitate this, Oracle Beehive provides Sensitivities. A sensitivity is simply an ACL, given a label, and made available for users to apply to any object. A sensitivity on its own is not yet active; users apply sensitivities to entities, and thereby create an ACL containing the ACEs encapsulated by the sensitivity.

Sensitivities are always created at the workspace level of scope. A workspace manager can view or search for sensitivities within the workspace, using a client application.

For example, a “Confidential” sensitivity could be applied to a set of artifacts. This sensitivity would prevent access from ALL_USERS, but grant access only to the creator of the object. Later, an instruction like “share all Confidential documents with Bob” can be used by a workspace manager, to alter the sensitivity (and thus all objects in the workspace having that sensitivity) to grant Bob READ access to them.

Implicit Access Control

Implicit access control is accomplished by granting or revoking various levels of access (privileges) to users. You may do this by applying privileges directly to user accounts, or to any container which contains user accounts (groups, organizations, or the enterprise).

Privileges

Oracle Beehive includes about 47 pre-defined privileges. Privileges are used to define access when there is no relevant entity. Examples include EMAIL_USER, AUDITOR, and WORKSPACE_MGR. The EMAIL_USER privilege grants access to use the E-mail functionality in Oracle Beehive. The AUDITOR privilege grants access to Oracle Beehive auditing functionality. You can list all privileges using the `beectl list_privileges` command. (Note: you cannot create custom privileges.)

Roles

The easiest and most flexible way to manage implicit access control is through the use of Roles.

A role is similar to a sensitivity; it encapsulates a collection of granted and revoked privileges, under a single label. However, you assign roles to users, thereby granting those users implicit access. Oracle Beehive is packaged with default roles, which you may modify, and you can also create new roles to meet the requirements of your organization.

Managing Privileges

A privilege is an assignable entity that grants access to some part of Oracle Beehive. For example, the EMAIL_USER privilege grants access to Oracle Beehive e-mail functions. Table 13–1, " Default Oracle Beehive Privileges" lists the default privileges pre-defined in every Oracle Beehive deployment.

Table 13–1 Default Oracle Beehive Privileges

Privilege	Access Granted
ARCHIVE_MGR	Business administrator access to archiving functionality
AUDITOR	Access to read auditing logs
AUDIT_ADMIN	Allows assignee to configure audit policy
BYPASS	Global superuser privilege allows assignee to perform any possible function
CALENDAR_MGR	Business administrator access to time management functionality
CALENDAR_USER	User-level access to the calendar functionality
CONF_MGR	Business administrator access to web conferencing
CONF_USER	User-level access to the web conferencing functionality[1]
CONTENT_MGR	Business administrator access to file management
CONTENT_USER	User-level access to the file management functionality within workspaces[1]
DELEGATE	
DIAGNOSE	
DM_MGR	Business administrator access to mobile device management functionality
EMAIL_MGR	Business administrator access to e-mail settings
EMAIL_USER	User-level access to the e-mail functionality
EXCEED_QUOTA	
FORUM_MGR	Administrator access to discussions functionality

Table 13–1 (Cont.) Default Oracle Beehive Privileges

Privilege	Access Granted
FORUM_USER	User-level access to the discussions functionality[1]
IM_MGR	Business administrator access to instant messaging functionality
IM_USER	User-level access to the instant messaging functionality[1]
LOGIN	
MARKER_MGR	
MODIFY_ACL	System administrator access to add to, or modify, access control lists on objects
NOTIFICATION_MGR	Administrator access to notifications functionality
NOTIFICATION_USER	User-level access to the notifications functionality
ORGANIZATION_MGR	Administrator access to
POLICY_MGR	Administrator access to policies
PREFERENCE_MGR	Administrator access to setting default preferences
PROTOCOL_USER	User-level access to WebDAV and FTP protocols[1]
QUOTA_MGR	Administrator access to quota settings within the specified scope
READALL	
RESOURCE_MGR	Administrator access to resource management
ROLE_MGR	Allows access to assign roles, within a specified scope
SECURITY	
SUBSCRIPTION_MGR	Administrator access to subscriptions functionality
SUBSCRIPTION_USER	User-level access to the subscription functionality
SYSTEM_OPER	
TASK_MGR	Administrator access to tasks functionality
TASK_USER	User-level access to the tasks functionality[1]
TIMEZONE_MGR	Administrator access to managing time zones
USER_MGR	Administrator access to managing users
VERSION_MGR	
VOICE_USER	User-level access to the voice messaging functionality[1]
WEBADMIN_USER	Allows access to Oracle Beehive Administration Console (Oracle Beekeeper). This privilege is only available in Oracle Beehive version 1.3 or later
WORKSPACE_ADD	Allows assignee to add workspaces within the specified scope
WORKSPACE_MGR	Administrator access within the specified workspace scope

[1] In Oracle Beehive Release 2, this privilege is inactive, meaning, the granted access is always available regardless of assignment of this privilege. The ability to disable this level of access by revoking this privilege is planned for a future release of Oracle Beehive.

Managing Roles

A role is a collection of privileges, which can be assigned to users. Roles are convenient because they allow you to provide several different layers of privileges to a heterogeneous population of users. A user may have any number of roles, granting the appropriate collection of privileges at a variety of scopes.

> **Note:** "Scope" refers to a logical level of organization within the Oracle Beehive deployment; the enterprise, or a specific organization, or a specific workspace.

An assigned role has two parts:

- An **Assignee** (a user or group)
- A **Role Definition** - Encapsulates access types and privileges for a given scope. It has no effect until assigned

This section includes the following topics:

- About Role Definitions
- Creating Role Definitions
- Creating Assigned Roles
- Modifying Roles
- Deleting Roles

About Role Definitions

As part of the initial install seeding, a collection of role definitions are created and assigned (through assigned roles) to the ALL_USERS dynamic group (which is also seeded at install time). This means that all users within your enterprise are eligible to be assigned any of the default roles. Additional default role definitions are available, but unassigned.

When you create custom role definitions, you can assign them to specific groups, and at scopes lower than the enterprise level, if you wish.

You can list the existing role definitions using the beectl list_role_definitions command:

```
beectl> list_role_definitions
```

This produces output similar to the following:

```
---------------------------------------------------------------------------------------------------
---------------------------------------
| role_definition      | name                  | description          | access_types         |
privileges           | always_enabled        |
---------------------------------------------------------------------------------------------------
---------------------------------------
| acrd=AUDIT-ADMIN,enpr=mycompany | AUDIT-ADMIN         |                     |
| [AUDIT_ADMIN]        | true                  |
| acrd=AUDITOR,enpr=mycompany | AUDITOR              |                      |
| [AUDITOR]            | true                  |
| acrd=DEFAULT_RESOURCE_ROLE_DEFINITION,enpr=mycompany | DEFAULT_RESOURCE_ROLE_DEFINITION |
ResourceMgrRoleDef   |                       | [CALENDAR_USER, TASK_USER] | true                 |
| acrd=DELEGATOR,enpr=mycompany | DELEGATOR           |                     |
| [DELEGATE]           | true                  |
| acrd=enterprise-administrator,enpr=mycompany | enterprise-administrator |                |
| [ARCHIVE_MGR, EXCEED_QUOTA, MARKER_MGR, ORGANIZATION_MGR, PREFERENCE_MGR, QUOTA_MGR, ROLE_MGR,
USER_MGR, VERSION_MGR, WORKSPACE_MGR] | true                   |
| acrd=enterprise-system,enpr=mycompany | enterprise-system   | enterprise-system   |
| [BYPASS]             | true                 |
| acrd=user-calendar,enpr=mycompany | user-calendar        |                      |
| [CALENDAR_USER]      | true                 |
| acrd=user-content,enpr=mycompany | user-content         |                      |
| [CONTENT_USER]       | true                 |
| acrd=user-core,enpr=mycompany | user-core            |                      |
| [LOGIN, PROTOCOL_USER, WORKSPACE_ADD] | true                 |
| acrd=user-discussions,enpr=mycompany | user-discussions     |                      |
| [FORUM_USER]         | true                 |
| acrd=user-email,enpr=mycompany | user-email           |                      |
| [EMAIL_USER]         | true                 |
| acrd=user-notification,enpr=mycompany | user-notification    |                      |
| [NOTIFICATION_USER]  | true                 |
| acrd=user-subscription,enpr=mycompany | user-subscription    |                      |
| [SUBSCRIPTION_USER]  | true                 |
| acrd=user-task,enpr=mycompany | user-task            |                      |
| [TASK_USER]          | true                 |
| acrd=workspace-coordinator,enpr=mycompany | workspace-coordinator |                | +RWDEO
| [CALENDAR_MGR, CONF_MGR, CONTENT_MGR, EMAIL_MGR, FORUM_MGR, IM_MGR, MARKER_MGR, MODIFY_ACL,
NOTIFICATION_MGR, POLICY_MGR, PREFERENCE_MGR, READALL, ROLE_MGR, SECURITY, SUBSCRIPTION_MGR, USER_
MGR, VERSION_MGR, WORKSPACE_MGR] | true                 |
| acrd=workspace-document-coordinator,enpr=mycompany | workspace-document-coordinator |
| +RWDEO           | [CONTENT_MGR, FORUM_MGR, MARKER_MGR, MODIFY_ACL, VERSION_MGR] | true
|
| acrd=workspace-member,enpr=mycompany | workspace-member     |                      | +RWDEO
|                      | true                 |
| acrd=workspace-participant-coordinator,enpr=mycompany | workspace-participant-coordinator |
| +RO                  | [MODIFY_ACL, ROLE_MGR, USER_MGR] | true                 |
| acrd=workspace-viewer,enpr=mycompany | workspace-viewer     |                      | +RO
|                      | true                 |
-----------------------------------------------------------------------------------------------
---------------------------------------
```

As an example, the role definition `user-email` has the privilege EMAIL-USER granted to it. Similarly, the `workspace-viewer` role definition has access types `+RO` (it allows READ and DISCOVER access) granted to it.

You can use the `beectl list_role_definitions` command to search for role definitions based on name, using the `%` wildcard. For example, to find all role definitions related to workspaces, issue the following command:

```
beectl> list_role_definitions --name workspa%
```

This command should return results including: `workspace-participant-coordinator`, `workspace-viewer`, `workspace-member`, `workspace-document-coordinator`, and `workspace-coordinator`.

Creating Role Definitions

A role definition is simply a collection of privileges and access types. While Oracle Beehive comes pre-seeded with a variety of default role definitions, you may find it convenient to create your own role definitions.

To create a custom role definition, use the `beectl add_role_definition` command, setting the scope as desired (the whole enterprise, a specific organization, or a specific workspace).

Note: For some privileges, scope does not make sense or is not required. For example, the LOGIN privilege allows or disallows user authentication with Oracle Beehive. Users do not log in at a level of scope, so a definition of scope makes no sense in combination with the LOGIN privilege.

For privileges that do take a scope, if you do not specify a scope, scope is set to the enterprise level by default.

For example:

```
beectl> add_role_definition --scope orgn=human_resources,enpr=mycompany --name
hr-administrator --description "A manager of users and roles" --privilege ROLE_MGR
--privilege USER_MGR --access_types ORWDE --always_enabled true
```

In this example, two privileges are granted: ROLE_MGR and USER_MGR. Additionally, all access types are granted. This role might be appropriate for a high-level administrator needing access to all objects in the organization, and the ability to grant roles to all users within that organization. Alternatively, if the scope were set at the level of a workspace, it might be useful for a director or manager needing access to manage users and assign roles only within that workspace. Note that in order to assign roles, a user must have the ROLE_MGR privilege, and in order to assign them to users, the USER_MGR privilege is needed.

As another example:

```
beectl> add_role_definition --scope enpr=mycompany --name workspace-viewer
--description "Can see everything in the workspace" --access_types OR --always_
enabled true
```

In this example, the role definition grants blanket Discover and Read access within the context in which it is assigned. This might be useful as a guest role, assignable at the level of a workspace, which would allow a guest user to find and read, but not modify, objects within the scope specified. This role definition is also set to "always enabled". This affects how the role definition works when it is used in an assigned role. See "Creating Assigned Roles" on page 13-8 for details.

This role definition example actually recreates one of the default role definitions provided in Oracle Beehive: the workspace-viewer role.

Notes: To see a list of all privileges, use the `beectl list_privileges` command.

In Oracle Beehive, the convention is that all role definitions are in all-lowercase, while privileges are in all-caps. If you follow this convention when creating role definitions, you may find it easier to distinguish them from privileges when constructing your commands.

Exposing a Role Definition to Users

Once you have created a role definition, you have the option of creating an ACE which will expose the role definition to users. If you do not take this step, the role definition will only be manageable from the command-line. Users within Oracle Beehive will not be able to see, or make use of, the custom role.

To expose a role definition, add an ACE using the `beectl add_local_ace` command. For example, to expose the role definition to all members of a privileged group called "ADMINS":

```
beectl> add_local_ace --entity acrd=workspace-viewer,enpr=mycompany --accessor
grup=ADMINS,enpr=mycompany --access_types RE
```

This command grants Read and Execute access types, enabling the members of the ADMINS group to execute (assign) the role definition specified in <Role>. Note that the entity is defined by listing the role definition by name, the organization by name, and the enterprise by name, separated by commas (no spaces). The accessor is specified by name, using the same fully-qualified syntax.

All access control role definitions have the four-letter code `acrd`, organizations are `orgn`, and enterprises are `enpr`.

Creating Assigned Roles

You may tie a role definition to an object called an Assigned Role. Assigned roles grant role definitions directly to users or groups. An assigned role always specifies assignees, and may also specify scope.

An Assigned role ties a role definition to a set of accessors in a given scope.

If the role definition that is associated with the assigned role is marked "Always Enabled", then the assigned role is always enabled for all of its accessors; it may not be disabled. If the role definition is *not* marked "Always Enabled", then by default it is disabled for all of its accessors unless and until it is enabled (by a user).

Note that if an assigned role is disabled (that is, its role definition is not marked "Always Enabled", and it has not been enabled in the current user context), privileges and access types that are specified in the role definition have no effect. However, if the role definition *denies* any access types, those denied access types still take effect.

For example, to assign the workspace-viewer role to a user, with the scope of a workspace (so the role definition applies for that user to that specific workspace only), use the `beectl add_assigned_role` command:

```
beectl> add_assigned_role --scope enpr=mycompany --name my_assigned_role
--assigned_scope wksp=myworkspace,orgn=human_resources,enpr=mycompany --role_
definition acrd=workspace-viewer,enpr=mycompany --accessor <USERID>
```

Provide the identifier of the user for `<USERID>`.

Note that some users have privileges allowing them to assign roles (such as the ROLE_MGR privilege). Whenever a user assigns a role definition to another user, they are effectively creating an assigned role. If the user can see (discover) and assign (execute) permissions for a role definition, they can assign it to other users for which they have management privileges (such as USER_MGR).

Modifying Roles

You can modify any role definition by using the `beectl modify_role_definition` command:

```
beectl> modify_role_definition --role_definition <role_definition_id> [--name
<name>] [--description <description>] [[--privilege <privilege_name>]...]
[--access_types <access_types_string>] [--always_enabled <boolean_value>]
```

Only use options for parts of the role definition you wish to change. For example, if you only wish to change the name, identify the role definition using the `--role_definition` and its identifier, and then use the `--name` option to specify a new name.

You can change the assignee(s) of any assigned role, adding or removing accessors, using the `beectl modify_assigned_role` command:

```
beectl> modify_assigned_role --assigned_role <assigned_role_id> [--name <name>]
[--description <description>] [--assigned_scope <assigned_scope_id>] [--role_
definition <role_definition_id>] [[--add_accessor <accessor_id>]...] [[--remove_
accessor <accessor_id>]...]
```

Use the `--add_accessor` and `--remove_accessor` options to list accessors from whom you want to grant or revoke the assigned role.

Deleting Roles

You can delete roles using the `beectl delete_assigned_role` or command:

```
beectl> delete_assigned_role --assigned_role <assigned_role_id>
```

When you delete an assigned role, you are effectively removing it from all assignees. This changes the privileges for all affected users, so caution is advised.

You can also delete a role definition, using the `beectl delete_role_definition` command:

```
beectl> delete_role_definition --role_definition <role_definition_id>
```

Note: If you attempt to delete a role definition that is currently assigned, Oracle Beehive will return an error. You must unassign the role definition from all actors in the system before you can delete it.

Creating and Managing Access Control Entities and Sensitivities

This section describes how to create and manage ACEs and sensitivities.

Any Oracle Beehive object may have any number of ACEs, collected into an Access Control List (ACL). ACLs are simply tables of ACEs, all of which apply to a single controllable entity. You never need to "create" an ACL: creating an ACE on any object automatically creates an ACL for it. If an object has one or more ACEs, it has an ACL, and if all ACEs are removed from an object, it no longer has an ACL. Oracle Beehive manages this task for you. All you need to do is create, modify, and delete ACEs according to your needs.

Sensitivities are just like ACLs, except that they are not attached to an object. Instead, a sensitivity is an ACL-template, which may be used again and again to assign the same ACL to many different objects.

This section contains the following topics:

- Creating and Managing ACEs
- Creating and Managing Sensitivities

Creating and Managing ACEs

You can create an ACE for most types of entities in Oracle Beehive.

Each ACE contains three values: the entity, which specifies where the ACE will be applied (a workspace, folder, calendar, etc.); the accessor, which specifies for whom this ACE applies (a user, members of a specified group, etc.); and a string defining access types (+-RWOED).

This section contains the following topics:

- Viewing ACEs
- Creating New ACEs
- Modifying ACEs
- Deleting ACEs

Viewing ACEs

To view the ACL of an object (listing all ACEs currently applied to it), use the beectl list_local_acl command:

```
beectl> list_local_acl --entity <entity_id>
```

Specify the entity using its identifier. Each ACE is listed, including its accessor and access type list.

Creating New ACEs

To create a new ACE (adding it to the ACL of an existing entity), use the beectl add_local_ace command:

```
beectl add_local_ace --entity <entity_id> --accessor <accessor_id> [--access_types
<access_types_string>]
```

An ACL may only have a single ACE for each accessor. If an ACE for an accessor already exists, you must modify it to alter the access types, by modifying the ACE.

Modifying ACEs

To modify an existing ACE, use the beectl modify_local_ace command:

```
beectl> modify_local_ace --entity <entity_id> --accessor <accessor_id> [--access_
types <access_types_string>]
```

Specify both the entity and the accessor, using their identifiers, to identify the ACE you wish to edit. The access type string you provide will replace the current access type string of the specified ACE.

Deleting ACEs

To delete an ACE, use the beectl delete_local_ace command:

```
beectl> delete_local_ace --entity <entity_id> --accessor <accessor_id>
```

Specify both the entity and the accessor, using their identifiers, to identify the ACE you wish to delete.

Creating and Managing Sensitivities

A sensitivity is a template ACL, containing ACEs. Users can assign sensitivities repeatedly to different objects, saving the effort of having to create the same ACEs again and again. A sensitivity also exposes the ability to set access control at the object level (explicit access control) to privileged users who do not have access to the command line.

> **Note:** Sensitivities are always created and managed at the workspace level of scope.

This section contains the following topics:

- Viewing Sensitivities
- Creating New Sensitivities
- Modifying Sensitivities
- Deleting Sensitivities

Viewing Sensitivities

To see a list of all sensitivities available in a workspace, first, get the workspace's unique identifier by using the `beectl list_workspaces` command:

```
beectl> list_workspaces --scope enpr=Example --type p
```

This produces output similar to the following:

```
----------------------------------------------------------------------
| Workspace Name | Workspace Type | Identifier |
----------------------------------------------------------------------
| SystemWorkspace | PERSONAL | wksp=SystemWorkspace,enpr=mycompany|
| beeadmin's Personal Workspace | PERSONAL | wksp=beeadmin's Personal
Workspace,enpr=mycompany |
| example.user's Personal Workspace | PERSONAL | wksp=example.user's Personal
Workspace,enpr=mycompany |
----------------------------------------------------------------------
```

To list the sensitivities in a workspace, use the `beectl list_sensitivities` command:

```
beectl> list_sensitivities --workspace "wksp=example.user's Personal
Workspace,enpr=mycompany"
```

This produces output similar to the following:

```
----------------------------------------------------------------------------------
----------------------------------
| sensitivity | name | description | sensitivity_only | delegatable |
----------------------------------------------------------------------------------
----------------------------------
| acsn=Confidential,wksp=example.user's Personal Workspace,enpr=mycompany |
Confidential | confidential sensitivity | false | true |
| acsn=Normal,wksp=example.user's Personal Workspace,enpr=mycompany | Normal |
normal sensitivity | false | true |
| acsn=Private,wksp=example.user's Personal Workspace,enpr=mycompany | Private |
private sensitivity | true | false |
| acsn=Public,wksp=example.user's Personal Workspace,enpr=mycompany | Public |
public sensitivity | false | true |
-------------------------------------------------------------------------------
---------------------------------
```

You can use the `--name` command to search for only sensitivities matching the provided string. Use the `%` symbol as a wildcard.

To review the ACL (all ACEs) of a specific sensitivity, use the `beectl list_sensitivity_acl` command:

```
beectl> list_sensitivity_acl --sensitivity " acsn=Public,wksp=example.user's
Personal Workspace,enpr=mycompany"
```

This produces output similar to the following:

```
---------------------------------------------
| accessor              | access_types        |
---------------------------------------------
| grup=ALL_USERS,enpr=mycompany | +RO                  |
---------------------------------------------
Listed SensitivityACL for Sensitivity 'acsn=Confidential,wksp=example.user's
Personal Workspace,enpr=mycompany'
```

As you can see, the default personal workspace contains a default sensitivity called "Public". This sensitivity grants Read and Discover access to the members of the ALL_USERS group, which contains all users in the enterprise. Therefore, this sensitivity allows any other Oracle Beehive user to discover and read (but not modify or delete) items in the workspace marked with this sensitivity.

Creating New Sensitivities

To create a new sensitivity, use the `beectl add_sensitivity` command:

```
beectl> add_sensitivity --workspace <workspace_id> --name <name> [--description
<description>] [--sensitivity_only <boolean_value>] [--delegatable <boolean_
value>]
```

Specify the workspace scope using the `--workspace` option, and give the sensitivity a unique name using `--name`. Optionally, you can give the sensitivity a description, using the `--description` option; the description will be readable by users when they are choosing and assigning sensitivities.

To create a new sensitivity ACE, use the `beectl add_sensitivity_ace` command:

```
beectl> add_sensitivity_ace --sensitivity <entity_id> --accessor <accessor_id>
[--access_types <access_types_string>]
```

Specify the sensitivity using its identifier. As with normal ACEs, sensitivity ACEs combine an accessor with an access type string. You may only have one ACE in a given ACL with the same accessor. You can use the `beectl modify_sensitivity_ace` command if you want to change the access type string for an existing accessor on a sensitivity ACE.

For more details on ACEs, see "Creating and Managing ACEs" on page 13-10.

Modifying Sensitivities

To modify an existing sensitivity, use the `beectl modify_sensitivity` command:

```
beectl> modify_sensitivity --sensitivity <sensitivity_id> [--name <name>]
[--description <description>] [--sensitivity_only <boolean_value>] [--delegatable
<boolean_value>]
```

Specify the sensitivity using its identifier. Note that you add, modify, and remove ACEs from a sensitivity using the `beectl add_sensitivity_ace`, `beectl modify_sensitivity_ace`, and `beectl delete_sensitivity_ace` commands.

To modify an existing sensitivity ACE, use the `beectl modify_sensitivity_ace` command:

```
beectl> modify_sensitivity_ace --sensitivity <entity_id> --accessor <accessor_id>
[--access_types <access_types_string>]
```

This command effectively replaces an existing sensitivity ACE with the new one you specify, based on the `--accessor`. Hence, you must specify both the sensitivity identifier and the accessor, and then optionally provide a new access type string.

Deleting Sensitivities

To delete a sensitivity ACE, use the `beectl delete_sensitivity_ace` command:

```
beectl> delete_sensitivity_ace --sensitivity <sensitivity_id> --accessor
<accessor_id>
```

Specify the sensitivity using its identifier, and specify the accessor. The ACE corresponding to the accessor you identify will be removed.

To delete a sensitivity, use the `beectl delete_sensitivity` command:

```
beectl> delete_sensitivity --sensitivity <sensitivity_id>
```

Specify the identifier of the sensitivity you wish to delete.

Note: If a sensitivity is currently assigned to any objects in its workspace, Oracle Beehive will return an error message when you attempt to delete it. You must unassign the sensitivity from all objects before it can be deleted.

14

Managing Oracle Beehive Auditing

Oracle Beehive includes a comprehensive auditing framework which allows you to record the activities of users, the disposition of artifacts, and the operation of the system.

This module contains the following topics:

- About Oracle Beehive Auditing
- Creating and Managing Oracle Beehive Audit Policies
- Creating and Managing Oracle Beehive Audit Trails
- Audit Events Structure

About Oracle Beehive Auditing

Oracle Beehive includes an Audit Service, which performs the function of writing audit information to the Audit Repository, in the Oracle Beehive database. The Audit Service records a selection of information, as defined by audit policies and templates. Policies and templates, in turn, define which events will be recorded by the Audit Service.

In Oracle Beehive, auditing is for activity tracking and recording. Auditing allows you to track and record the activities of users and administrators as they perform actions in the system. These activities include logging on and off, creating, modifying, or deleting content, altering system configuration parameters, starting and stopping processes, and so forth. The goal is to provide a framework for keeping tabs on who does what to the system.

An audit record contains information about who (what user or users), what (what artifacts, services, or interfaces), where (what scope or context), when (a date/time stamp), and how (what client or interface, what command).

The Audit Service depends upon system events to trigger audit policies, causing a record to be written to the Audit Repository.

You can manage audit functions from either the `beectl` command line, or from Oracle Beekeeper. To manage auditing in Oracle Beekeeper, you must log in with sufficient privileges. The AUDIT_ADMIN privilege allows you to configure audit policy, while the AUDITOR privilege allows you to read audit logs.

By default, all Administrator-controlled auditing functions are turned off.

Note: Records Management related events are always audited. You cannot turn on or off auditing of these events. For more information about Records Management in Oracle Beehive, see Chapter 7, "Integrating Oracle Universal Records Management (Oracle URM) with Oracle Beehive" in the *Oracle Beehive Integration Guide*.

For more information about privileges, see "Managing Privileges" on page 13-3.

About Audit Events

The audit framework makes use of a subset of the Oracle Beehive business events, called the audit events. Audit events trigger auditing actions whenever they fit the criteria specified in an active audit policy.

You can review a list all of various categories of audit events by using the `beectl list_events` command:

```
beectl> list_events
```

This command lists the audit event categories, and their identifiers.

Note: By default, no event is raised when an Oracle Beehive user sends an e-mail message. You can enable sent e-mail events, and thereby enable auditing of sent e-mails. To do so, follow the instructions in "Configuring Sent E-mail Plugins" on page 8-5.

For more information about business events in Oracle Beehive, see: Chapter 11, "Managing Oracle Beehive Events and Policies".

Disabled Events

By default, certain events related to the Time Management Service are disabled. These events will not be audited when they occur unless you enable them. See "Disabled Events" on page 11-2 for a list of the disabled events.

If you want to audit any of these events, you must enable them by changing the `EnableGenericClassOfTMBusinessEvents` property of the Time Management Service. See Chapter 4, "Oracle Beehive Parameter Reference," in the *Oracle Beehive Administrator's Reference Guide*

About Audit Policies

An audit policy combines a collection of events to be audited with a scope, to define what will be audited. Scope can be user-focused, such as a user or group, or it can be a container, such as the enterprise, or one or more organizations, workspaces, or folders. Once you create an audit policy, the system begins to record events that match the policy in the Audit Repository.

About Audit Trails

Once you have created audit policies, data is written to the Audit Repository. An audit trail is a view or report of some portion of that data. You can create and configure audit trails to include only the specific data you are interested in. You can think of an audit trail as a query of the Audit Repository.

Creating and Managing Oracle Beehive Audit Policies

An audit policy combines audit templates (which specify events) with a context. In this case, context includes any combination of:

- The enterprise, or one or more organizations or workspaces or both
- One or more users
- One or more groups

An audit policy is a definition of rules and actions that determine which events should be recorded in the audit repository.

An audit policy dictates when the events in an audit template should be recorded in the Audit Repository. You can create many audit policies to suit your organization’s requirements.

This section contains the following topics:

- Creating Audit Policies
- Listing Audit Policies
- Modifying Audit Policies
- Enabling and Disabling Audit Policies
- Deleting Audit Policies
- Example Audit Policy

Creating Audit Policies

You can create a new audit policy as an XML file, and then use the `beectl add_audit_policy` command to upload it to the system:

```
beectl> add_audit_policy --file <full path to the policy xml file>
```

The Audit Policy XML file used for setting audit context references an audit template you specify, and allows you to set the policy to either a level of scope (user, organization, or enterprise), or a level of the content hierarchy, such as an individual entity, folder, or workspace. If you do not reference any context (no scope is referenced, and no actor or content directive is used), then the scope is assumed to be global, and the events referenced in the audit template will be raised for ALL contexts (user or content) where that activity occurs.

You can also create a new audit policy using Oracle Beekeeper:

1. Log in to Oracle Beekeeper, and under **Enterprises**, click **Audit**
2. Select the Policies tab. All existing audit policies are listed
3. Click New. The New Audit Policy window opens
4. Enter a name and description for the new audit policy
5. Choose an audit template to use from the Template picker
6. Optionally, choose a scope from the Scope picker:
   a. Click on the Scope picker icon to open the **Scope picker** window. The enterprise is shown by default
   b. Select the enterprise, and then optionally click the first icon to show organizations, or the second to show workspaces, at the enterprise level of scope.

When you select a displayed organization or workspace, the picker changes to that level of scope, and you can continue to use the icons to descend the scope hierarchy. Use the back button to step up a level of scope hierarchy, and use the **Filter by** field to search through the currently displayed results.

c. Once you have located the enterprise, organization, or workspace you want to use as the scope for this policy, select it and then click **OK**

7. Optionally, choose one or more users. Select the **Users** tab, and click **Add** to add a user. The Users picker opens. Use the **Search** field to find users based on name or e-mail address.

   Tip: Search with an empty field to return a list of all users.

   When you locate a user you want to add, select it and click **Add**. The user appears in the list on the **Users** tab. Repeat this process to add additional users. Select a user and click **Remove** on the Users tab to remove a user from the list

8. Optionally, choose one or more groups. Select the **Groups** tab, and click **Add** to add a group. The Groups picker opens. Use the **Search** field to find groups based on name.

   Tip: Search with an empty field to return a list of all users.

   When you locate a group you want to add, select it and click **Add**. The group appears in the list on the **Groups** tab. Repeat this process to add additional groups. Select a user and click **Remove** on the Groups tab to remove a group from the list

9. Click **Apply** to save your changes without closing the New Audit Policy window, or click **Save and Close** to save your changes and close the window.

   Your new policy appears in the list in the **Policies** tab

Once an Audit Policy has been put in place, events will be generated and recorded to the database Audit Repository. The act of creating an audit policy also enables that policy immediately.

Listing Audit Policies

To see active audit policies using `beectl`, use the `beectl list_audit_policies` command:

```
beectl> list_audit_policies [--name <Name of the audit policy>] [--container
<Container identifier>]
```

Optionally, you can provide a name, container, or both, to list only those policies with the name or applied to the context of the container.

Note: One audit policy is seeded at install: the `Audit Management` policy, which audits all Audit management related events.

Modifying Audit Policies

You can modify existing audit policies with `beectl` using the `beectl modify_audit_policy` command:

```
beectl> modify_audit_policy --policy <Audit policy identifier> --file <full path
to the policy xml file>
```

> **Note:** You may not change the audit template of an existing audit policy. You must create a new audit policy to apply the policy on a different container, or to use another audit template.
>
> If you make changes to an existing audit template, that will only apply to new policies you create with the template after it is updated. **Existing policies will not be updated with changes made to an audit template**.

You can modify existing audit policies using Oracle Beekeeper:

1. Log in to Oracle Beekeeper, and under **Enterprises**, click **Audit**
2. Select the **Policies** tab. All existing audit policies are listed
3. Select a policy from the list. Its details are shown in the lower pane. Using the **General**, **Users**, and **Groups** tabs, make your desired changes
4. Click **Apply** to apply your changes to the audit policy, or click **Reset** to revert to the currently-saved version of the policy

Enabling and Disabling Audit Policies

You can disable active audit policies, and re-enable inactive audit policies. This allows you to easily turn audit on and off at a granular level. To enable or disable an active audit policy using `beectl`, use the `beectl modify_audit_policy` command with the `--enable` option:

```
beectl> modify_audit_policy --policy <Audit policy identifier> --file <full path
to the policy xml file> --enable [true|false]
```

You still must provide a path to the policy file, but if you do not wish to modify the content of the audit policy, you should reference the file originally used to create the audit policy.

You can disable and enable audit policies using Oracle Beekeeper:

1. Log in to Oracle Beekeeper, and under **Enterprises**, click **Audit**
2. Select the **Policies** tab. All existing audit policies are listed
3. Select an audit policy, and in the lower pane, on the **General** tab, select or de-select the **Enabled** check box to enable or disable the policy. Click **Apply** to apply your change. The policy is enabled or disabled.

Deleting Audit Policies

You can delete an existing audit policy with `beectl` by using the `beectl delete_audit_policy` command:

```
beectl> delete_audit_policy --policy <Audit policy identifier>
```

You can get the audit policy's identifier by using the `beectl list_audit_policies` command.

You can delete an existing audit policy with Oracle Beekeeper:

1. Log in to Oracle Beekeeper, and under **Enterprises**, click **Audit**
2. Select the **Policies** tab. All existing audit policies are listed

3. Select the policy you want to delete, and click **Delete**. In the confirmation box, click **OK**. The policy is deleted

Example Audit Policy

Example 14–1, "Simple Audit Policy" demonstrates a simple audit policy XML file that creates a policy sourcing the `Audit management events` template, and raises events in the context of the two users listed.

Example 14–1 Simple Audit Policy

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<AuditPolicyInfo>
    <name>Audit Management Policy</name>
    <description>Sample test policy</description>
    <template>Audit management events</template>
    <actor add='true' id='user=user1'/>
    <actor add='true' id='user=user2'/>
</AuditPolicyInfo>
```

In this example, since no scope was specified, all events specified in the `Audit management events` audit template will be audited for both of the specified users. Actors can be users or groups.

Note that there is an attribute of the `<actor>` element called "`add`", which in this example is set to "`true`". When you modify an audit policy, you can provide an `<actor>` element and set this attribute to "`false`" to delete the actor from the modified audit policy. When you modify an audit policy, set this value to "true" to either add a new actor, or to modify an existing actor.

Creating and Managing Oracle Beehive Audit Trails

Once you have enabled auditing (by creating one or more audit policies), audit information accumulates in the Audit Repository. You can view selection of this data by running an audit trail. An audit trail is a query against the Audit Repository.

This section contains the following topics:

- Listing Audit Trails
- Creating Audit Trails
- Modifying Audit Trails
- Exporting Audit Trails
- Validating Audit Trails
- Deleting Audit Trails
- Example Audit Trail

Listing Audit Trails

You can list existing audit trails with `beectl` using the `beectl list_audit_trails` command:

```
beectl> list_audit_trails [--name <Name of the audit trail>]
```

You can list details about a specific audit trail by using the `--name` option to reference the audit trail.

You can list existing audit trails with Oracle Beekeeper:

1. Log in to Oracle Beekeeper, and under **Enterprises**, click **Audit**
2. Select the **Trails** tab. All existing audit trails are listed

Creating Audit Trails

To create an audit trail using `beectl`, begin by creating an XML file for your audit trail.

An example audit trail file is located in your Oracle Beehive install folder, in the templates subfolder `ORACLE_HOME/beehive/templates/audit/trail_ex.xml`.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<AuditTrailInfo>
    <name>Trail Name</name>
    <description>Trail description</description>

    <actor>Collab ID of actor</actor>
    <actor>user=sample.id</actor>

    <entity>Collab ID of entity</entity>

    <startTime>Start Time predicate</startTime>
    <endTime>End Time predicate</endTime>

    <serviceName>Service Name</serviceName>
    <userName>User Name</userName>
    <activity>Type of Activity (CREATE, DELETE, ETC)</activity>
    <eventType>Event Type</eventType>
    <predicate>Predicate Type (all, any)</predicate>
</AuditTrailInfo
```

Add a new audit trail by using the `beectl add_audit_trail` command, referencing your XML file:

```
beectl> add_audit_trail --file <Full path of the input file>
```

To create an audit trail using Oracle Beekeeper:

1. Log in to Oracle Beekeeper, and under **Enterprises**, click **Audit**
2. Select the **Trails** tab. All existing audit trails are listed
3. Click **New**. The New Audit Trail window opens
4. Enter a name and description for the new trail
5. Optionally, pick a **Start Time** and/or **End Time**. These options specify a range of dates and times for which you want to see audit records. If you leave these fields blank, records from all dates and times will be returned
6. Optionally, use the **Actor Filter** tab to specify one or more actors for whom you want to see audit records. Click the **Actor Filter** tab, click **Add** to open the **Users** window, and then search for actors. Select an actor and click **Add** to add it to the list in the **Actor Filter** tab. Only those records created by actions from the actors listed in the **Actor Filter** tab will be returned
7. Optionally, use the **Entity Filter** tab to specify one or more entities (audit templates and policies) for which you want to see audit records. Click the **Entity Filter** tab, click **Add** to open the Audited Entity Picker, and then search for

entities. Select an entity and click **Add** to add it to the list in the **Entity Filter** tab. Only audit records for the entities listed in the **Entity Filter** tab will be returned.

8. Optionally, use the **Cumulative Member Records** tab to specify *additional* individual audit records for the audit trail, or to remove records that are already matched by the audit trail.

   The **Cumulative Member Records** tab lets you select records that do not otherwise match the audit trail filter criteria, and add them to the audit trail. Click the **Cumulative Member Records** tab, click **Add** to open the Audit Record Picker, and then search for records. Select a record and click **Add** to add it to the list in the **Cumulative Member Records** tab.

   Remove records from the trail by selecting them in the **Cumulative Member Records** list, and then clicking Remove.

9. Click **Apply** to save your changes without closing the New Audit Trail window, or click **Save and Close** to save your changes and close the window.

   Your new trail appears in the list in the **Trails** tab

Modifying Audit Trails

To modify an existing audit trail using `beectl`, edit the XML file used to create the audit trail, and then reference it with the `beectl modify_audit_trail` command:

```
beectl> modify_audit_trail --trail <Audit trail identifier> --file <Full path of
the input file>
```

To modify an existing audit trail using Oracle Beekeeper:

1. Log in to Oracle Beekeeper, and under **Enterprises**, click **Audit**
2. Select the **Trails** tab. All existing audit trails are listed
3. Select a trail from the list. In the lower pane, you can make edits to the trail. When you have finished making changes, click **Apply** to update the audit trail, or click **Reset** to revert to the saved version of the audit trail without making changes to it.

Exporting Audit Trails

You can run an audit trail (like running a query), which extracts the data specified by the trail to a file.

To export the data using `beectl`, use the `beectl export_audit_trail` command, specifying the audit trail and a file for the output:

```
beectl export_audit_trail --trail <Audit trail identifier> --file <Full path of
the output file>
```

You can export the data using Oracle Beekeeper:

1. Log in to Oracle Beekeeper, and under **Enterprises**, click **Audit**
2. Select the **Trails** tab. All existing audit trails are listed
3. Select a trail from the list
4. Click the Export button, and then choose a filename and location. The audit trail records are saved as an XML-formatted file.

Validating Audit Trails

You can validate an audit trail to ensure that there are no errors.

To validate an audit trail using `beectl`, use the `beectl validate_audit_trail` command:

```
beectl validate_audit_trail --trail <Audit trail identifier> [--count <Maximum
number of audit records to print>]
```

To validate an audit trail using Oracle Beekeeper:

1. Log in to Oracle Beekeeper, and under **Enterprises**, click **Audit**
2. Select the **Trails** tab. All existing audit trails are listed
3. Select one or more trails from the list
4. Click the **Validate** button. A dialog box opens to indicate whether valid records were found for the selected audit trails.

Deleting Audit Trails

You can delete an audit trail using `beectl`, by using the `beectl delete_audit_trail` command:

```
beectl delete_audit_trail --trail <Audit trail identifier>
```

To delete an audit trail using Oracle Beekeeper:

1. Log in to Oracle Beekeeper, and under **Enterprises**, click **Audit**
2. Select the **Trails** tab. All existing audit trails are listed
3. Select one or more trails from the list
4. Click the **Delete** button, and click **OK** in the confirmation dialog box. The selected trails are deleted.

Example Audit Trail

The following is an XML file used to create an example audit trail, and an example of the exported audit data based on that trail.

Example 14–2 Example Audit Trail

Example 14–3 shows two example audit records included in an exported audit trail.

Example 14–3 Example Exported Audit Trail Data

```
<?xml version="1.0" encoding="utf-8"?>
<AuditTrail>
     <name>update_trail_name1221015531621640000</name>
     <description>Updated description</description>
     <createdOn>2008-09-10T02:58:49.212</createdOn>
     <modifiedOn>2008-09-10T02:58:53.004</modifiedOn>
     <recordCount>5251</recordCount>
     <records>
<event name="ACCOUNT_LOGIN_SUCCEEDED">
               <InstanceId>BEEFIX.srv.example.com</InstanceId>
               <HomeInstance>BEEFIX.srv.example.com</HomeInstance>
               <HostId>srv.example.com</HostId>
               <HostNwaddr>srv.example.com</HostNwaddr>
               <OracleHome>/private/jdoe/product/b1.0.4/beefix</OracleHome>
               <OrgId>26703</OrgId>
               <ComponentId>23333</ComponentId>
               <HostingClientId>null</HostingClientId>
               <ClientOS>null</ClientOS>
               <RemoteIP>null</RemoteIP>
               <ModuleId>ocs</ModuleId>
               <ProcessId>ocs</ProcessId>
               <ThreadId>0</ThreadId>
               <UpstreamComponentId>OCSAPP</UpstreamComponentId>
               <DownstreamComponentId>OCSCORE</DownstreamComponentId>

<ECID>684F:5B25:aurc:54131E861EC8CC82E040578C9B9A7310000000008621</ECID>
               <SessionId>476</SessionId>
               <LogonTime>2008-09-09T18:38:04.000</LogonTime>
               <AuthenticationMethod>PLAIN</AuthenticationMethod>
               <ApplicationName>LOGON</ApplicationName>
               <EventType>ACCOUNT_LOGIN_SUCCEEDED</EventType>
               <EventCategory>LOGIN</EventCategory>
               <EventStatus>SUCCESS</EventStatus>
               <TstzOriginating>2008-09-10T01:38:04.184</TstzOriginating>
               <ComponentName>LOGON</ComponentName>
               <Initiator>user=beeadmin</Initiator>
               <UserName>beeadmin</UserName>
               <MessageText>null</MessageText>
               <FailureCode>SUCCESS</FailureCode>
               <Target>enpr=Example</Target>
               <Resource>enpr=Example</Resource>
               <Roles>principal=beeadmin</Roles>

<UserSession>684F:5B25:pcpl:C57ACA07B48D48499CE221AA5F0F01E8000000000002</UserSess
ion>
               <PrincipalType>PRIM</PrincipalType>
               <Information>{{SOURCE: }}; {{TARGET: }}</Information>
          </event>
          <event name="ACCOUNT_LOGIN_SUCCEEDED">
               <InstanceId>BEEFIX.srv.example.com</InstanceId>
               <HomeInstance>BEEFIX.srv.example.com</HomeInstance>
               <HostId>srv.example.com</HostId>
               <HostNwaddr>srv.example.com</HostNwaddr>
               <OracleHome>/private/jdoe/product/b1.0.4/beefix</OracleHome>
               <OrgId>26703</OrgId>
               <ComponentId>23333</ComponentId>
               <HostingClientId>null</HostingClientId>
               <ClientOS>null</ClientOS>
               <RemoteIP>null</RemoteIP>
               <ModuleId>ocs</ModuleId>
               <ProcessId>ocs</ProcessId>
               <ThreadId>0</ThreadId>
               <UpstreamComponentId>OCSAPP</UpstreamComponentId>
               <DownstreamComponentId>OCSCORE</DownstreamComponentId>

<ECID>684F:5B25:aurc:54131E861EC8CC82E040578C9B9A7310000000008623</ECID>
               <SessionId>477</SessionId>
               <LogonTime>2008-09-09T18:38:05.000</LogonTime>
               <AuthenticationMethod>PLAIN</AuthenticationMethod>
               <ApplicationName>LOGON</ApplicationName>
               <EventType>ACCOUNT_LOGIN_SUCCEEDED</EventType>
               <EventCategory>LOGIN</EventCategory>
               <EventStatus>SUCCESS</EventStatus>
               <TstzOriginating>2008-09-10T01:38:04.570</TstzOriginating>
               <ComponentName>LOGON</ComponentName>
               <Initiator>user=beeadmin</Initiator>
               <UserName>beeadmin</UserName>
               <MessageText>null</MessageText>
               <FailureCode>SUCCESS</FailureCode>
               <Target>enpr=Example</Target>
               <Resource>enpr=Example</Resource>
               <Roles>principal=beeadmin</Roles>

<UserSession>684F:5B25:pcpl:C57ACA07B48D48499CE221AA5F0F01E8000000000002</UserSess
ion>
               <PrincipalType>PRIM</PrincipalType>
               <Information>{{SOURCE: }}; {{TARGET: }}</Information>
          </event>
     </records>
</AuditTrail>
```

Audit Events Structure

This reference section lists all of the audit events included in each audit event category.

Access Control Events

Table 14–1 lists auditable events related to access control.

Table 14–1 Access Control Events

Event Subcategory	Events
ASSIGNED_ROLE_ASYNC_EVENTS	ASSIGNED_ROLE_DELETED ASSIGNED_ROLE_UPDATED ASSIGNED_ROLE_CREATED
DELEGATED_ROLE_ASYNC_EVENTS	DELEGATED_ROLE_UPDATED DELEGATED_ROLE_CREATED DELEGATED_ROLE_DELETED
ROLE_DEFINITION_ASYNC_EVENTS	ROLE_DEFINITION_UPDATED ROLE_DEFINITION_CREATED ROLE_DEFINITION_DELETED
SENSITIVITY_ASYNC_EVENTS	SENSITIVITY_DELETED SENSITIVITY_CREATED SENSITIVITY_UPDATED

Address Book Events

Table 14–2 lists auditable events related to address books.

Table 14–2 Address Book Events

Event Subcategory	Events
ADDRESSBOOK_ASYNC_EVENTS	ADDRESSBOOK_MOVED ADDRESSBOOK_UNDELETED ADDRESSBOOK_CREATED ADDRESSBOOK_DELETED ADDRESSBOOK_UPDATED
PERSON_CONTACT_ASYNC_EVENTS	PERSON_CONTACT_DELETED PERSON_CONTACT_CREATED PERSON_CONTACT_UNDELETED PERSON_CONTACT_UPDATED PERSON_CONTACT_MOVED
RESOURCE_CONTACT_ASYNC_EVENTS	RESOURCE_CONTACT_UPDATED RESOURCE_CONTACT_CREATED RESOURCE_CONTACT_DELETED RESOURCE_CONTACT_MOVED RESOURCE_CONTACT_UNDELETED

Artifact Events

Table 14–3 lists auditable events related to artifacts.

Table 14–3 Artifact Events

Event Subcategory	Events
ANNOUNCEMENT_ASYNC_EVENTS	ANNOUNCEMENT_DELETED ANNOUNCEMENT_UPDATED ANNOUNCEMENT_UNDELETED ANNOUNCEMENT_ARCHIVED ANNOUNCEMENT_CREATED
BOND_ASYNC_EVENTS	BOND_DELETED BOND_CREATED BOND_UPDATED
CATEGORY_ASYNC_EVENTS	CATEGORY_REMOVED CATEGORY_APPLIED CATEGORY_DELETED CATEGORY_CREATED CATEGORY_UPDATED
DFDRAFT_ASYNC_EVENTS	DFDRAFT_MOVED DFDRAFT_UPDATED DFDRAFT_UNDELETED DFDRAFT_CREATED DFDRAFT_ARCHIVED DFDRAFT_DELETED
DOCUMENT_ASYNC_EVENTS	DOCUMENT_DELETED DOCUMENT_UPDATED DOCUMENT_CHECKEDIN DOCUMENT_MOVED DOCUMENT_WORKING_COPY_UPDATED DOCUMENT_CHECKOUT_CANCELLED DOCUMENT_CREATED DOCUMENT_UNDELETED DOCUMENT_CHECKEDOUT DOCUMENT_ARCHIVED
ENTITY_LOCK_ASYNC_EVENTS	ENTITY_LOCKED ENTITY_UNLOCKED
EXTERNAL_ARTIFACT_ASYNC_EVENTS	EA_CREATED EA_DELETED EA_UPDATED

Table 14–3 (Cont.) Artifact Events

Event Subcategory	Events
FOLDER_ASYNC_EVENTS	FOLDER_MOVED FOLDER_UNDELETED FOLDER_ARCHIVED FOLDER_UPDATED FOLDER_CREATED FOLDER_DELETED
FORUM_ASYNC_EVENTS	FORUM_MOVED FORUM_CREATED FORUM_UNDELETED FORUM_ARCHIVED FORUM_DELETED FORUM_UPDATED
LABEL_ASYNC_EVENTS	LABEL_APPLIED LABEL_REMOVED LABEL_CREATED LABEL_DELETED LABEL_UPDATED
LINK_ASYNC_EVENTS	LINK_DELETED LINK_COPIED LINK_CREATED LINK_MOVED LINK_UPDATED LINK_UNDELETED
LOCK_ASYNC_EVENTS	ENTITY_LOCKED LOCK_UPDATED ENTITY_UNLOCKED
NOTIFICATION_EVENTS	(See Table 14–4, " Notification Events")
TOPIC_ASYNC_EVENTS	TOPIC_ARCHIVED TOPIC_MOVED TOPIC_DELETED TOPIC_CREATED TOPIC_UNDELETED TOPIC_UPDATED

Table 14–4 lists auditable events in the sub-category of Notification events.

Table 14–4 Notification Events

Event Subcategory	Events
NOTIFICATION_ASYNC_EVENTS	NOTIFICATION_CREATED NOTIFICATION_UPDATED NOTIFICATION_DELETED

Table 14–4 (Cont.) Notification Events

Event Subcategory	Events
NOTIFICATION_SCHEMA_ASYNC_EVENTS	NOTIFICATION_SCHEMA_DELETED NOTIFICATION_SCHEMA_CREATED NOTIFICATION_SCHEMA_UPDATED

Audit Events

Table 14–5 lists auditable events related to each audit event category.

Table 14–5 Audit Events

Event Subcategory	Events
AUDIT_ASYNC_EVENTS	AUDIT_TRAIL_DELETED AUDIT_TEMPLATE_DELETED AUDIT_TRAIL_UPDATED AUDIT_TEMPLATE_CREATED AUDIT_POLICY_DELETED AUDIT_POLICY_CREATED AUDIT_TEMPLATE_UPDATED AUDIT_POLICY_ENABLED AUDIT_POLICY_UPDATED AUDIT_TRAIL_CREATED AUDIT_POLICY_DISABLED
AUDIT_ASYNC_FAILED_EVE	AUDIT_POLICY_CREATE_FAILED AUDIT_TRAIL_UPDATE_FAILED AUDIT_TRAIL_CREATE_FAILED AUDIT_TRAIL_DELETE_FAILED AUDIT_TEMPLATE_UPDATE_FAILED AUDIT_POLICY_DELETE_FAILED AUDIT_TEMPLATE_CREATE_FAILED AUDIT_POLICY_UPDATE_FAILED AUDIT_TEMPLATE_DELETE_FAILED AUDIT_POLICY_DISABLE_FAILED AUDIT_POLICY_ENABLE_FAILED

Calendar Events

Table 14–6 lists auditable events related to calendars.

Table 14–6 Calendar Events

Event Subcategory	Events
CALENDAR_ASYNC_EVENTS	CALENDAR_ADDED CALENDAR_REMOVED CALENDAR_UPDATED

Table 14–6 (Cont.) Calendar Events

Event Subcategory	Events
DEFAULT_REMINDER_ASYNC_EVENTS	DEFAULT_REMINDER_ADDED DEFAULT_REMINDER_REMOVED DEFAULT_REMINDER_UPDATED
INVITATION_ASYNC_EVENTS	INVITATION_ADDED INVITATION_REMOVED INVITATION_UPDATED
OCCURRENCE_ASYNC_EVENTS	OCCURRENCE_ADDED OCCURRENCE_REMOVED OCCURRENCE_UPDATED
REMINDER_ASYNC_EVENTS	REMINDER_ADDED REMINDER_REMOVED REMINDER_UPDATED
RESOURCE_ASYNC_EVENTS	RESOURCE_CREATED RESOURCE_DELETED RESOURCE_UPDATED
TASKLIST_ASYNC_EVENTS	TASKLIST_ADDED TASKLIST_REMOVED TASKLIST_UPDATED
TODO_ASYNC_EVENTS	TODO_ADDED TODO_REMOVED TODO_UPDATED

Client Application Events

Table 14–7 lists auditable events related to client applications.

Table 14–7 Client Application Events

Event Subcategory	Events
CLIENT_APPLICATION_ASYNC_EVENTS	CLIENT_APPLICATION_CREATED CLIENT_APPLICATION_DELETED
CLIENT_APPLICATION_PATCHSET_ASYNC_EVENTS	CLIENT_APPLICATION_PATCHSET_DELETED CLIENT_APPLICATION_PATCHSET_CREATED
CLIENT_APPLICATION_PROV_UPDATED	CLIENT_APPLICATION_PROV_UPDATED
CLIENT_APPLICATION_VERSION_ASYNC_EVENTS	CLIENT_APPLICATION_VERSION_DELETED CLIENT_APPLICATION_VERSION_CREATED

Device Management Events

Table 14–8 lists auditable events related to device management.

Table 14–8 Device Management Events

Event Subcategory	Events
DEVICE_ASYNC_EVENTS	DEVICE_CREATED DEVICE_DELETED DEVICE_UPDATED
DEVICE_PROFILE_ASYNC_EVENTS	DEVICE_PROFILE_UPDATED DEVICE_PROFILE_CREATED DEVICE_PROFILE_DELETED
DEVICE_TYPE_ASYNC_EVENTS	DEVICE_TYPE_DELETED DEVICE_TYPE_CREATED DEVICE_TYPE_UPDATED

Enterprise Events

Table 14–9 lists auditable events related to Enterprises.

Table 14–9 Enterprise Events

Event Subcategory	Events
ENTERPRISE_ASYNC_EVENTS	ENTERPRISE_ARCHIVEPURGED ENTERPRISE_DELETED ENTERPRISE_UPDATED ENTERPRISE_CREATED

LDAP Sync Profile Events

Table 14–10 lists auditable events related to LDAP sync profiles.

Table 14–10 LDAP Sync Profile Events

Event Subcategory	Events
LDAP_SYNC_PROFILE_ASYNC_EVENTS	LDAP_SYNC_PROFILE_DELETED LDAP_SYNC_PROFILE_CREATED

Message Events

Table 14–11 lists auditable events related to messages.

Table 14–11 Message Events

Event Subcategory	Events
DISCUSSION_MESSAGE_ASYNC_EVENTS	DISCUSSION_MESSAGE_ARCHIVED DISCUSSION_MESSAGE_DELETED DISCUSSION_MESSAGE_UPDATED DISCUSSION_MESSAGE_CREATED DISCUSSION_MESSAGE_MOVED

Table 14–11 (Cont.) Message Events

Event Subcategory	Events
ES_ASYNC_EVENTS	ES_MSG_MOVED ES_MSG_DELETED ES_MSG_UNDELETED ES_MSG_DELIVERED ES_MSG_UPDATED ES_MSG_ADDED
FAX_MESSAGE_ASYNC_EVENTS	FAX_MESSAGE_UPDATED FAX_MESSAGE_MOVED FAX_MESSAGE_DELETED FAX_MESSAGE_COPIED FAX_MESSAGE_CREATED
IMS_ASYNC_EVENTS	[IMS_OFFLINE_MSG_ADDED IMS_OFFLINE_MSG_DELETED IMS_OFFLINE_MSG_MOVED IMS_OFFLINE_MSG_UNDELETED
MESSAGE_DELIVERY_ASYNC_EVENTS	MESSAGE_DELIVERY_STATUS_UPDATED MESSAGE_DELIVERY_STATUS_DELETED MESSAGE_DELIVERY_STATUS_CREATED
NOTIFICATION_EVENTS	(See Table 14–4, " Notification Events" on page 14-14)
VOICE_MESSAGE_ASYNC_EVENTS	VOICE_MESSAGE_MOVED VOICE_MESSAGE_CREATED VOICE_MESSAGE_UPDATED VOICE_MESSAGE_DELETED VOICE_MESSAGE_COPIED

Organization Events

Table 14–12 lists auditable events related to Organizations.

Table 14–12 Organization Events

Event Subcategory	Events
ORGANIZATION_ASYNC_EVENTS	ORGANIZATION_ARCHIVED ORGANIZATION_UPDATED ORGANIZATION_CREATED ORGANIZATION_DELETED

Policy Subscription Events

Table 14–13 lists auditable events related to policies and subscriptions.

Table 14–13 Policy Subscription Events

Event Subcategory	Events
POLICY_ASYNC_EVENTS	POLICY_UPDATED POLICY_DELETED POLICY_CREATED
SUBSCRIPTION_ASYNC_EVENTS	SUBSCRIPTION_UPDATED SUBSCRIPTION_ENABLED SUBSCRIPTION_DELETED SUBSCRIPTION_DISABLED SUBSCRIPTION_CREATED
SUBSCRIPTION_TEMPLATE_ASYNC_EVENTS	SUBSCRIPTION_TEMPLATE_CREATED SUBSCRIPTION_TEMPLATE_DELETED SUBSCRIPTION_TEMPLATE_UPDATED

Records Management Events

Table 14–14 lists auditable events related to Records Management.

Table 14–14 Records Management Events

Event Subcategory	Events
RM_ASYNC_EVENTS	RECORD_UNFILED RECORD_FILED RECORD_DISP_PROC_STEP_SUCCEEDED RECORD_PURGED
RM_ASYNC_FAILED_EVENTS	RECORD_CREATE_FAILED RECORD_DELETE_FAILED RECORD_PURGE_FAILED RECORD_DISP_PROC_STEP_FAILED

Search Events

Table 14–15 lists auditable events related to search.

Table 14–15 Search Events

Event Subcategory	Events
SEARCH_ASYNC_EVENTS	SEARCH_FINISHED SEARCH_STARTED

Security Events

Table 14–17 lists auditable events related to security.

Table 14–16 Security Events

Event Subcategory	Events
ACCOUNT_ASYNC_EVENTS	ACCOUNT_LOGIN_SUCCEEDED ACCOUNT_LOGOUT_SUCCEEDED ACCOUNT_LOCKED

Table 14–16 (Cont.) Security Events

Event Subcategory	Events
ACCOUNT_ASYNC_FAILED_EVENTS	ACCOUNT_LOGIN_FAILED
CREDENTIAL_ASYNC_EVENTS	CREDENTIAL_DELETED CREDENTIAL_EXPIRED CREDENTIAL_RESET CREDENTIAL_UPDATED CREDENTIAL_CREATED
CREDENTIAL_ASYNC_FAILED_EVENTS	CREDENTIAL_DELETE_FAILED CREDENTIAL_CREATE_FAILED CREDENTIAL_UPDATE_FAILED CREDENTIAL_RESET_FAILED

Service Configuration Update Events

Table 14–17 lists auditable events related to Service configuration updates.

Table 14–17 Service Configuration Update Events

Event Subcategory	Events
SERVICE_CONFIG_UPDATED	SERVICE_CONFIG_UPDATED

System Events

Table 14–18 lists auditable events related to the core Oracle Beehive system.

Table 14–18 System Events

Event Subcategory	Events
INSTANCE_START_STOP_ASYNC_EVENTS	INSTANCE_STARTED INSTANCE_STOPPED
SYSTEM_START_STOP_ASYNC_EVENTS	SERVICE_STOPPED INSTANCE_STOPPED SERVICE_STARTED INSTANCE_STARTED

Time Management Events

Table 14–19 lists auditable events related to time management.

Table 14–19 Time Management Events

Event Subcategory	Events
TM_SUBSCRIPTION_ASSIGNMENT_ASYNC_EVENTS	TM_SUBSCRIPTION_ASSIGNMENT_INDIRECTLY_DELETED TM_SUBSCRIPTION_ASSIGNMENT_INDIRECTLY_UPDATED TM_SUBSCRIPTION_ASSIGNMENT_NEW_OR_TIME_UPDATED

Table 14–19 (Cont.) Time Management Events

Event Subcategory	Events
TM_SUBSCRIPTION_INVITATION_ASYNC_EVENTS	TM_SUBSCRIPTION_INVITATION_INDIRECTLY_DELETED TM_SUBSCRIPTION_INVITATION_INDIRECTLY_UPDATED TM_SUBSCRIPTION_INVITATION_NEW_OR_RESCHED
TM_SUBSCRIPTION_INVITATION_SERIES_ASYNC_EVENTS	TM_SUBSCRIPTION_INVITATION_SERIES_INDIRECTLY_DELETED TM_SUBSCRIPTION_INVITATION_SERIES_INDIRECTLY_UPDATED TM_SUBSCRIPTION_INVITATION_SERIES_NEW_OR_RESCHED
TM_SUBSCRIPTION_OCCURRENCE_ASYNC_EVENTS	TM_SUBSCRIPTION_OCCURRENCE_RESOURCE_PARTICIPANT_INDIRECTLY_UPDATED TM_SUBSCRIPTION_OCCURRENCE_USER_PARTICIPANT_INDIRECTLY_UPDATED
TM_SUBSCRIPTION_TODO_PARTICIPANT_INDIRECTLY_UPDATED	TM_SUBSCRIPTION_TODO_PARTICIPANT_INDIRECTLY_UPDATED
TM_TIMEZONE_DEFINITION_UPDATED	TM_TIMEZONE_DEFINITION_UPDATED

User Management Events

Table 14–20 lists auditable events related to user management.

Table 14–20 User Management Events

Event Subcategory	Events
EXTERNAL_PERSON_ASYNC_EVENTS	EXTERNAL_PERSON_PURGED EXTERNAL_PERSON_CREATED EXTERNAL_PERSON_DELETED EXTERNAL_PERSON_UPDATED
GROUP_ASYNC_EVENTS	GROUP_UPDATED GROUP_PURGED GROUP_DELETED GROUP_CREATED GROUP_UNDELETED
USER_ASYNC_EVENTS	USER_UPDATED USER_DELETED USER_PURGED USER_CREATED

Workspace Events

Table 14–1 lists auditable events related to Workspaces.

Table 14–21 Workspace Events

Event Subcategory	Events
VERS_CFG_ASYNC_EVENTS	VERS_CFG_DELETED VERS_CFG_UPDATED VERS_CFG_CREATED
WORKSPACE_ASYNC_EVENTS	WORKSPACE_PURGED WORKSPACE_CREATED WORKSPACE_ARCHIVED WORKSPACE_DELETED WORKSPACE_UPDATED
WORKSPACE_QUOTA_ASYNC_EVENTS	WORKSPACE_HQUOTA_OVERFLOW WORKSPACE_SQUOTA_OVERFLOW

XMPP Events

Table 14–22 lists auditable events related to XMPP messaging.

Table 14–22 XMPP Events

Event Subcategory	Events
XMPP_ASYNC_EVENTS	XMPP_FILE_TRANSFERRED XMPP_USER_LOGGEDIN XMPP_USER_LOGGEDOUT

15

Backing Up and Recovering Oracle Beehive

This module gives recommendations for backup and recovery strategies for your Oracle Beehive deployment. It includes the following sections:

- Introduction to Backing Up and Recovering Oracle Beehive
- Backing Up Oracle Beehive
- Recovering Oracle Beehive
- Backing Up and Recovering Individual E-mail Accounts

Introduction to Backing Up and Recovering Oracle Beehive

An Oracle Beehive deployment can comprise of two or three tiers, depending on the deployment topology selected, and may include a Web tier running the application listener services, an Application tier running the application business logic, and a Database tier containing the Oracle Beehive data repository (including business data, application seed data and configuration data). Your backup and restore strategy should include each tier individually, while also insuring that after a recovery there are no synchronization issues between the various tiers.

This module provides recommendations. However, every Oracle Beehive deployment is unique, and your organization's requirements for availability, backup storage strategy, and recovery scenarios are also unique. You should use the recommendations in this module as a baseline for forming a comprehensive backup strategy that best suits your organization's needs. You should also consider writing a set of recovery procedures specific to your hardware and deployment, to ensure rapid and accurate restoration whenever a problem occurs.

> **See Also:** For more information about backing up and recovering Oracle Databases, see:
>
> - Chapter 15, "Backup and Recovery", in *Oracle Database 10g Concepts*
> - *Oracle Database 10g Backup and Recovery Basics*
> - *Oracle Database 10g Backup and Recovery Advanced User's Guide*

This section contains the following topic:

- When to Perform Backups

When to Perform Backups

Your backup and recovery strategy should be prepared to handle hardware and software failures, as well as human errors. Human errors include accidental deletion of critical code or configuration files, dropping a table or tablespace, accidentally purging unarchived data, and so forth. If these errors occur while the system is in production use by live users, then damage control should be performed quickly by taking appropriate restore measures.

Sometimes incorrect configuration can trigger corruption of data to such an extent that the system can become unusable. Under such circumstances it becomes imperative to restore the system from a backup of the system taken before the symptoms of logical corruption began to manifest.

Because most of the Oracle Beehive data (business, seed and configuration) resides in the database and there is very little information which is persisted in the Applications tier file system, the frequency of database backups (both full and incremental) should be significantly higher than the Applications tier backups.

You should consider performing a backup under any of the following circumstances:

- Create a Baseline backup:
  - Immediately after installation of Oracle Beehive
  - Immediately before installing any software patch or upgrade, to provide rollback capability should the upgrade cause a problem
  - Immediately after installing any software patch or upgrade, to create a snapshot prior to any post-installation configuration

  See: "Creating a Baseline Backup of Oracle Beehive" on page 15-2
- Whenever you have already scheduled system downtime, you should use the opportunity to perform a full closed (cold) backup. See "Performing an Offline (Cold) Backup of Oracle Beehive" on page 15-4
- During minimum system usage times, on a regular schedule, perform online (hot) database backups. Oracle recommends daily incremental backups, and weekly full backups, of a production Oracle Beehive database. See "Performing an Online (Hot) Backup of Oracle Beehive Database" on page 15-5

Backing Up Oracle Beehive

This section describes some options for backing up your Oracle Beehive deployment. It contains the following topics:

Creating a Baseline Backup of Oracle Beehive

During software product installations, you can perform a backup of the entire environment as a snapshot. The purpose of taking such a snapshot is to successfully recover to a "known good" baseline, if irrecoverable errors are made during the post-installation phase. This is sometimes referred to as a baseline install backup.

For example, after you complete the basic installation of Oracle Beehive, but before getting started with more advanced configurations such as setting up end-to-end SSL encryption/decryption or integrating with a third party LDAP user directory, you should consider making a baseline backup.

Oracle recommends taking complete backups after every major milestone of the installation. The Application tier and database backups need to be synchronized. You can also make use of a backup naming convention, so that the name includes both the timestamp and a brief description of the milestone.

A baseline backup of Oracle Beehive includes:

- Creating a Baseline Backup of the Application Tier
- Creating a Baseline Backup of the Database Tier

> **Note:** If you have configured Transport Layer Security (TLS) with Oracle Wallet, as described in the Oracle Beehive install guide for your platform, you should also back up the files in the following location:
>
> `<Oracle home>/Apache/Apache/conf/ssl.wlt/default`
>
> For more information, see "Configuring TLS with Oracle Wallet" in the *Oracle Beehive Installation Guide* for your platform.

Creating a Baseline Backup of the Application Tier

Shutdown all Oracle Beehive Application tier processes and then backup the ORACLE_HOME and oraInventory for the Application tier installation using the archiving tool of your choice; tar, cpio, WinZip, or any other archiving tool. Backing up the oraInventory is important as it contains crucial information about the Application tier installation. To reduce the backup overhead you can exclude some of the following beehive log file directories from this backup:

- $ORACLE_HOME/beehive/logs
- $ORACLE_HOME/opmn/logs
- $ORACLE_HOME/Apache/Apache/logs
- $ORACLE_HOME/j2ee/home/log
- $ORACLE_HOME/j2ee/oc4j_soa/log
- $ORACLE_HOME/j2ee/OCSCORE/log
- $ORACLE_HOME/j2ee/OCSMGMT/log
- $ORACLE_HOME/j2ee/OCSAPP/log

On Windows systems, the above files are typically located in `C:\Program Files\`. The remainder of the directory structure is identical. In addition, you should back up the following registry entries:

```
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet002\Control\Session Manager\Environment
HKLM\SOFTWARE\ORACLE
HKLM\SYSTEM\CurrentControlSet\Services
```

Caution: On UNIX and Linux platforms, if certain listener services in the Oracle Beehive installation are running on privileged port numbers (any port in the 0 - 1024) range, then you must perform the backup using super-user privileges. This is because under such circumstances, certain files have the set-uid set by the root user. If the archive is created by a user not having root privileges then during a restore operation, the set-uid will be lost and the file will lack appropriate privileges. This could prevent Oracle Beehive from making use of those privileged ports, until the problem is fixed.

Creating a Baseline Backup of the Database Tier

Backing up Oracle Beehive database repository is similar to backing up any other Oracle database. Various options are available for backing up Oracle databases. Database backup options include:

- Oracle provided database backup tools like RMAN
- Oracle database export utility
- Third party backup tools
- Custom shell or SQL scripts

You can perform two types of database backups:

- Offline (cold) backup

  An offline backup is taken by shutting down the database first, and then backing up all data, log and control files of the database. Because the database has to be shut down first, this is also referred to as offline backup. See Performing an Offline (Cold) Backup of Oracle Beehive for example closed backup procedures.

- Online (hot) backup - An online backup is preferred when the database needs to be available for read/write and can't be shut down. However, in order to take a online backup, the database must be in ARCHIVELOG mode.

By default Oracle uses the database control files to store information about backups. Normally it is better to set up an RMAN catalog database to store RMAN metadata in. Read the Oracle Backup and Recovery Guide before implementing any RMAN backups.

Performing an Offline (Cold) Backup of Oracle Beehive

Oracle recommends taking a full offline (cold) backup of all Application tiers before going into production. From that point onwards, you should take a full closed backup taken after every Oracle Beehive patch is applied to the system. Additionally, a full offline backup is recommended every time a change impacting connectivity to configuration data, such as a change to the database connect string, schema passwords, and so forth is implemented, as this has the potential to hamper usability of older backups. Apart from that, whenever you have a scheduled outage window, take the opportunity to perform a full offline backup of all Application tiers.

Use an archival tool of your choice such as tar, cpio, or zip to take the backup of the entire Application tier ORACLE_HOME, and the oraInventory.

The following is a simple example demonstrating an offline backup of the database:

1. Ensure all Oracle Beehive processes are shut down. You must shut down Oracle Beehive Application tiers before shutting down the Oracle Beehive database

2. Run the following queries to get a list of all files that need to be backed up:

   ```
   select name from sys.v_$datafile;
   select member from sys.v_$logfile;
   select name from sys.v_$controlfile;
   ```

3. Shut down the Oracle Beehive database from SQL*Plus
4. Back up all files to disk or secondary storage (such as magnetic tape). Ensure that you backup all data files, all control files and all log files
5. When completed, restart the database, and then you may restart Oracle Beehive

> **Note:** Because the Oracle Beehive database is always in ARCHIVELOG mode, you can use archived log files to roll forward from a closed (cold) backup.

Performing an Online (Hot) Backup of Oracle Beehive Database

An online (hot) backup is preferred when the database needs to be available for read/write operations, and can't be shut down. In order to take an online backup, the database must be in ARCHIVELOG mode. Oracle Beehive requires the database to be in ARCHIVELOG mode, so online backup is a viable option.

You can use RMAN or custom scripts to schedule regular online backups of the database. For live production installations, Oracle recommends that at least one full RMAN backup be scheduled every week in addition to daily incremental backups.

> **Note:** Do not run online (hot) database backups during peak processing periods. The Oracle database will write complete database blocks, instead of the normal deltas, to redo log files while in backup mode. This can lead to excessive database archiving and even database freezes if the database experiences heavy use while in backup mode.

This section contains the following topics:

- Performing an Online Backup using SQL Commands
- Performing an Online Backup using RMAN

Performing an Online Backup using SQL Commands

The following procedure is a simple example demonstrating an online backup of the Oracle Beehive database:

1. One at a time, switch each database tablespace that needs to be backed up into backup mode:

   ```
   ALTER TABLESPACE xyz BEGIN BACKUP;
   ```

> **Note:** It is better to backup tablespaces one at a time, rather than all tablespaces at once, because substantial overhead is incurred for each tablespace in backup mode.You can script the command for all tablespaces, dynamically accounting for changes to the physical structure of the database since the last backup.

2. Copy the tablespace files into your backup directory or offline storage:

   ```
   ! cp xyzFile1 /backupDir/
   ```

3. When the copy is complete, disable the backup mode on the tablespace:

   ```
   ALTER TABLESPACE xyz END BACKUP;
   ```

4. Repeat the procedure for every tablespace in the database
5. When you have finished backing up each tablespace, backup the control files:

   ```
   ALTER SYSTEM SWITCH LOGFILE
   ```

   This command forces log switch to update control file headers

   ```
   ALTER DATABASE BACKUP CONTROLFILE TO '/backupDir/control.dbf';
   ```

Performing an Online Backup using RMAN

RMAN can facilitate the task of taking, organizing, and managing database backups to a great extent and is preferred over traditional ways of taking database backups. The biggest advantage of RMAN is that it will only backup used space in the database. RMAN does not put tablespaces in backup mode, saving on redo generation overhead. RMAN will re-read database blocks until it gets a consistent image.

The following is an example of how to perform an online backup of the Oracle Beehive database using RMAN:

```
rman target sys/*** nocatalog
run {
   allocate channel t1 type disk;
   backup
      format '/app/oracle/db_backup/%d_t%t_s%s_p%p'
      ( database );
   release channel t1;
}
```

Recovering Oracle Beehive

This section contains advice about recovering Oracle Beehive from backup. It contains the following topics:

- Recovering Oracle Beehive from a Baseline Backup
- Recovering the Oracle Beehive Application Tier from an Offline (Cold) Backup
- Recovering the Database Tier from an Online (Hot) Backup

Recovering Oracle Beehive from a Baseline Backup

A baseline backup of Oracle Beehive is usually a "known-good" backup, providing a restoration option that is sure to re-establish availability, at the possible expense of losing changes made to the system more recently.

Recovering the Application Tier from a Baseline Backup

A restore operation uses the same tool or utility which was used for taking the full backup. It is performed offline.

For the restore operation, first remove or move the existing ORACLE_HOME and oraInventory of the Oracle Beehive installation, and then restore them from the full backup. You should perform a corresponding database restore concurrently.

Recovering the Database Tier from a Baseline Backup

Database restore can be done using RMAN, import (if the backup was a logical backup taken using Oracle export utility), or flashing back the database (if flashback database is enabled). The advantage of using RMAN or flashback database is that you can restore the database to a specific point in time.

> **See Also:** For detailed information and examples, see Chapter 16, "Performing Flashback and Database Point-in-Time Recovery," in the *Oracle Database Backup and Recovery User's Guide*.

Recovering the Oracle Beehive Application Tier from an Offline (Cold) Backup

You can recover the Application tier, without making any changes to the Oracle Beehive database.

To restore the Application tier, perform the following procedure:

1. Use the same tool which you used to take the Application tier backup to restore the ORACLE_HOME and oraInventory from the last known-good backup just before the failure occurred
2. On the Application tier, use the `beectl modify_local_configuration_files` command to synchronize the Application tier with the latest information from the Oracle Beehive repository:

   ```
   beectl> modify_local_configuration_files
   ```

   This command must be run on every Application tier affected by the service outage. The process may take a while to complete.

   > **Note:** The Oracle Beehive data repository is used for storing all configuration data and is the final authority about a deployment configuration. Do not attempt to manually change configuration data on an Application tier.

Recovering the Database Tier from an Online (Hot) Backup

In case of a hardware failure, you can restore the database from RMAN catalog or use flash recovery to go to a point-in-time before the failure occurred. Though there is not much information which is persisted in the Application tier file system, an effective change control mechanism has to be in place to address any such deltas which might arise, such as from patches.

The following procedure is an example of how to restore a database which you have backed up using the tablespace-by-tablespace online (hot) backup method:

1. Copy all applicable archive log files to the target database destination file system. Also, copy all data, index and redo log files to the target database's file system (all files from the online backup)
2. Alter trace file for new file locations and ensure the `CREATE CONTROLFILE` statement specifies:

   ```
   USING <source_sid> RESETLOGS ARCHIVELOG
   ```

3. Using the original (source) database ORACLE_SID value, startup the target database with the new init.ora (New control file locations, LEAVE DB_NAME=<source_sid>, dump_dest, and so forth)
4. Recover the database using the following command:

   ```
   RECOVER DATABASE USING BACKUP CONTROLFILE UNTIL CANCEL;
   ```

5. Run the following command:

   ```
   ALTER DATABASE OPEN RESETLOGS;
   ```

6. Close the recovered database

Backing Up and Recovering Individual E-mail Accounts

Beginning in Oracle Beehive 2.0, you can use beectl to export e-mail for a specified user to an external archive file. This utility serves two main purposes:

- Simplifying the process of transporting an account from one Oracle Beehive deployment to another
- Periodic backup of selected user account data for emergency recovery purposes

You can selectively and periodically export certain users' e-mail accounts and store the exported files on the file system. Each export serves as a single-user backup. The export file captures the snapshot of the entire e-mail account, including the folder hierarchy, folder names, message received date, message flags and read status, as well as message payload in its original MIME form. All e-mail messages in all "regular" folders will be exported. A "regular" folder is a folder subscribable from an IMAP client. E-mail messages in the workspace Trash are not exported.

The file is stored in a proprietary format. It is not encrypted.

Because of the highly individualized approach, e-mail export is not suitable for entire system backup for any sizable user population. The mechanism is not incremental and not parallelized. Use the system-wide backup strategies described earlier in this chapter for such purposes.

When importing e-mail data, you should specify an empty folder in the specified user's personal workspace. Subfolders will be created corresponding to the subfolders that were archived.

If exported e-mails already exist in the target folder, they will be duplicated. If a subfolder with the same name already exists in the target folder, the import will fail.

To export e-mail, use the `beectl export_email_data` command:

```
beectl> export_email_data --user_name <user name> --file <file name>
```

Specify the `user_name` using the user's Primary Principal.

Specify the `file` name in either relative path (to the current working directory where beectl is invoked), or absolute path, to save the export file.

To import e-mail, use the `beectl import_email_data` command:

```
beectl> import_email_data --user_name <user name> --folder <folder name> --file
<file name>
```

Specify the `user_name` using the user's Primary Principal.

Specify the `file` name in either relative path (to the current working directory where beectl is invoked), or absolute path, from which to read the export file.

Specify the relative `folder` path under which the file content should be imported. The folder path should begin at the workspace level and use "/" as delimiter.

The following examples show appropriate syntax:

```
beectl> export_email_data --user_name john.doe@example.com  --file
/backup/10-JAN-2009/john_doe.bkp
beectl> import_email_data --user_name john.doe@example.com --folder INBOX/Restore_
10_Jan_2009 --file /backup/10-JAN-2009/john_doe.bkp
```

See Also: "Oracle Beehive Command-Line Utility" in the *Oracle Beehive Administrator's Reference Guide*

16

Oracle Beehive Disaster Recovery with Data Guard

Using Oracle Data Guard, you can configure your Oracle Beehive deployment for disaster recovery, in which end-user traffic is diverted to a standby database in the event of a failure or planned downtime in the primary system.

This module contains the following topics:

Introduction

Using Oracle Data Guard with your Oracle Beehive deployment ensures high availability, data protection, and disaster recovery for enterprise data. Data Guard provides a comprehensive set of services that create, maintain, manage, and monitor one or more standby databases to enable production Oracle databases to survive disasters and data corruptions and to minimize downtime for planned outages. Data Guard maintains these standby databases as copies of the production database. Then, if the production database becomes unavailable because of a planned or an unplanned outage, Data Guard can switch any standby database to the production role, minimizing the downtime associated with the outage. Data Guard can be used with traditional backup, restoration, and cluster techniques to provide a high level of data protection and data availability.

Prerequisites

This module assumes you are familiar with installing, configuring, and maintaining Oracle databases, Oracle Beehive, and Oracle Data Guard.

To configure a disaster recovery system for Oracle Beehive, your primary deployment can either be a single-instance Oracle Beehive server, or a deployment with multiple Oracle Beehive servers. Likewise, you can use either a single instance Oracle Database, or a multi-instance RAC deployment, for your primary and standby databases.

Configuring Oracle Beehive for Disaster Recovery

These configuration steps show how to set up Oracle Beehive for a switchover or failover to a local or remote standby deployment.

Planned and Unplanned Outages

In a planned outage (switchover) scenario, you perform manual steps to reverse database roles between the standby system and the primary system, and then shut down the (former) primary system. This is useful to maintain uptime when performing maintenance on the primary system, such as replacing hardware, that requires the servers to be shut down.

In an unplanned outage (failover) scenario, a failure of the primary system is detected and the standby system is made primary. Depending on your Oracle Data Guard configuration, manual steps may be required, or the failover may be performed automatically.

Local HA Standby (Database-Only) versus Full Deployment

In an Oracle Beehive local High-Availability (HA) standby (sometimes called 'database-only') recovery deployment, you create a standby deployment of the database used by Oracle Beehive. This provides for reduced downtime for planned and unplanned primary database outages. In this scenario, you use your primary deployment's Application tiers with the standby database, so no switchover or failover of the Oracle Beehive Application tier occurs.

In a full deployment scenario, you create a standby deployment of both the database and the Application tier(s). In a switchover or failover, you start the Oracle Beehive Application tiers on the standby system, and then redirect user traffic to the standby Oracle Beehive deployment. This provides for reduced downtime for planned and unplanned outages of the entire primary deployment.

> **Note:** it is also possible to have a hybrid deployment, in which you maintain both full standby infrastructure and local HA standby database servers.

Example Deployments

Figure Figure 16–1 shows a typical full-deployment scenario. In this scenario, a deployment with multiple Application tiers and database instances is duplicated by the standby deployment.

Figure 16–1 Oracle Beehive Full-Deployment Disaster Recovery Scenario

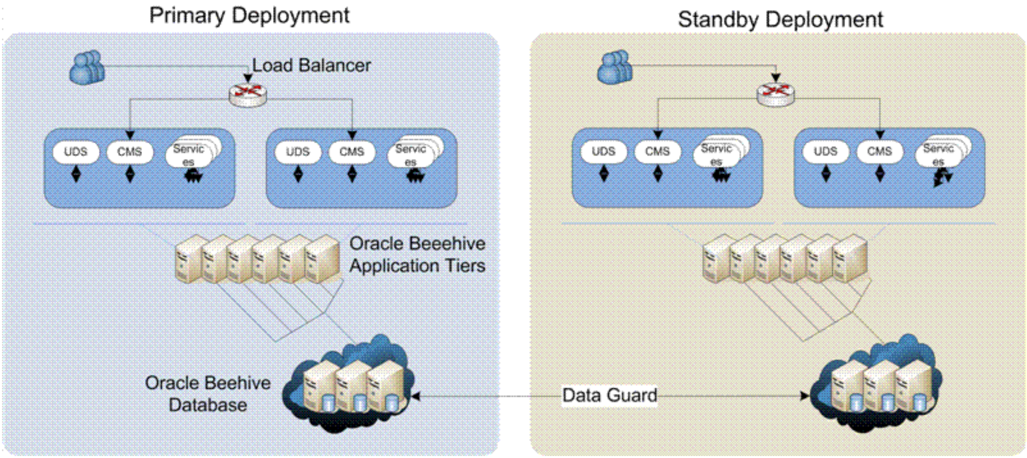
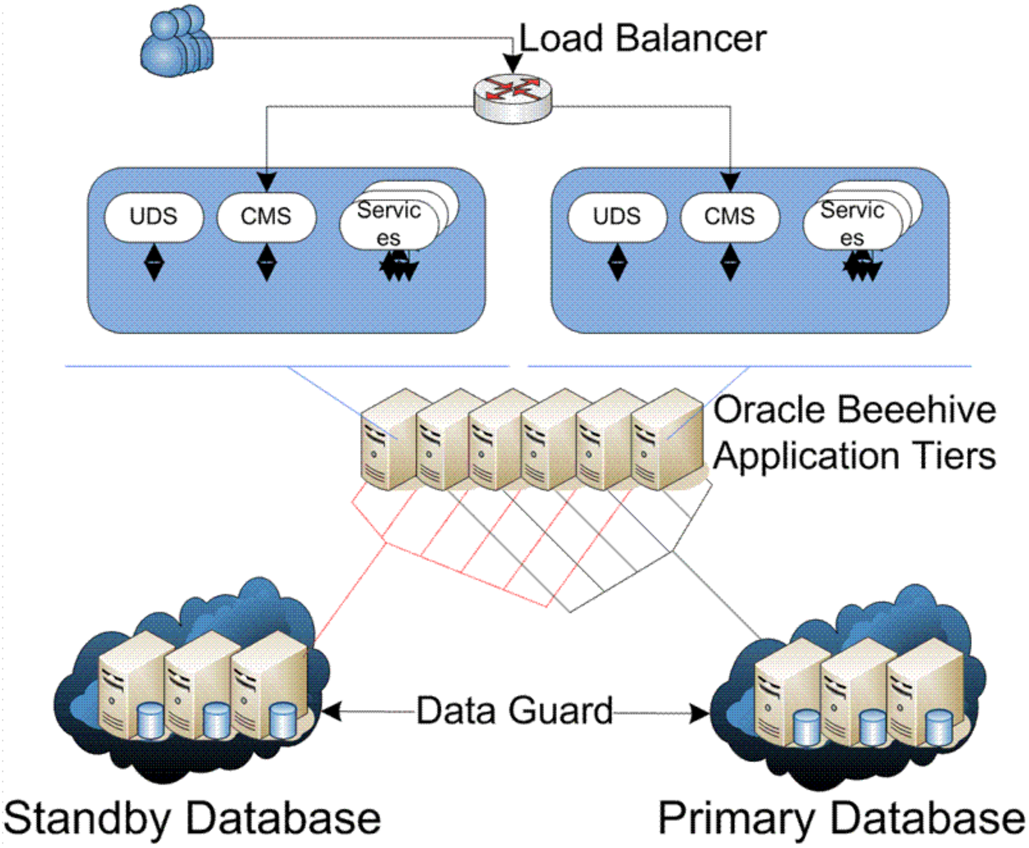


Figure Figure 16–2 shows a typical database-only deployment. In this scenario, there is a standby database that duplicates the primary multi-node database, but there are no standby Application tiers. The primary Application tiers remain active and connect to the standby database. This example provides database redundancy and failover in the event of a failure in the primary database, but there is no provision for Application-tier failover.

Figure 16–2 Oracle Beehive Local HA Standby Recovery Scenario



To configure Oracle Beehive for use in these scenarios, perform the following procedures:

- Creating the Physical Standby Database
- Recording Environment Details
- Configuring Service Relocation and Database Triggers
- Configuring Oracle Beehive Application Tiers

Creating the Physical Standby Database

There are multiple ways to create the physical standby database. Whether the standby is local or remote you can find the steps for creating a standby database in the Data Guard Administration guide, or on the Oracle Technology Network website at the following URL:

http://www.otn.oracle.com/goto/maa.

After you create the standby database, verify that it is performing properly as detailed in the Data Guard Administration guide in section 3.2.7. If it is a RAC deployment, also ensure that the new physical standby database is registered with the cluster database configuration. For example:

```
srvctl add database -d stby_orcl -o $ORACLE_HOME -m us.oracle.com -p
"+DATA/bhdc/spfilebhdc.ora" -n stby_orcl -r physical_standby
srvctl add instance -d stby_orcl -i stby_orcl1 -n stby_db1
srvctl add instance -d stby_orcl -i stby_orcl2 -n stby_db2
```

If this standby database is for a remote disaster recovery site, make sure when cloning the Oracle Beehive Application tiers that the `--do_not_start_at_end` option is used on the `clone_midtier` command.

Additionally, the standby Oracle Beehive servers should be disabled. To disable the standby Oracle Beehive servers, perform the following steps:

1. From any Oracle Beehive Oracle Home in the standby deployment, use the `beectl list_components` command to get the instance identifier for each standby Oracle Beehive server:

   ```
   beectl> list_components --type BeehiveInstance
   ```

2. For each instance, use the `beectl modify_property` command to disable the server:

   ```
   beectl> modify_property --component <component_id> --name Status --value
   DISABLED
   ```

3. After making property changes, you must run the `beectl activate_configuration` command to activate your proposed configuration changes:

   ```
   beectl> activate_configuration
   ```

Recording Environment Details

Having a record of the details of your primary and standby environments will be useful during setup, configuration, and maintenance of your disaster-recovery configuration.

The following is the sample environment used in examples in this chapter. The example Primary deployment consists of a two-node RAC database and a Beehive deployment using two Application tiers. The Standby deployment duplicates this configuration.

Tip: Refer back to this section as you read through the rest of this chapter. In all of the examples, you can use the example values to see what your actual values should be.

Example Deployment

- **Bootstrap Service**: A common service defined on all nodes, specifically used for service relocation. **In this chapter,** `beehivedg` **is the example common service name on all nodes**
- **Primary:**
  - **RAC Nodes (and Virtual IP)**: These are defined in the DNS and/or in `/etc/hosts`. In this chapter, `pr_db1` (`pr_db1_vip`) and `pr_db2` (`pr_db2_vip`) are the example primary database nodes
  - **Oracle Database Name**: In this chapter, the example primary database is `pr_orcl`
  - **ORACLE_SID**: In this chapter, `pr_orcl1` and `pr_orcl2` are the example ORACLE_SIDs
  - **Application tiers**: In this chapter, `pr_app1` and `pr_app2` are the example primary Application tiers
- **Standby:**

  See also: MAA 10*g* Setup Guide: Creating a RAC Physical Standby Database for a RAC Primary Database

  - **RAC Nodes (and Virtual IP)**: In this chapter, `stby_db1` (`stby_db1_vip`) and `stby_db2` (`stby_db2_vip`) are the example standby database nodes
  - **Oracle Database Name**: In this chapter, the example standby database is `stby_orcl`
  - **ORACLE_SID**: In this chapter, `stby_orcl1` and `stby_orcl2` are the example standby ORACLE_SIDs
  - **Application tiers**: In this chapter, `stby_app1` and `stby_app2` are the example standby Application tiers
- **IP Addresses**: You may find it useful to make a note of the IP addresses for each database node. IP addresses can be used as a RAC configuration method, or as a part of the DNS and/or `/etc/hosts` file.

Configuring Service Relocation and Database Triggers

To configure service relocation and database triggers, perform the steps in the following sections:

1. Create Managed Services
2. Setup Service Startup
3. Setup FAN ONS Publisher and Database Role Change Trigger

Create Managed Services

Perform the following steps to create managed services:

Caution: When creating managed services, do not include the managed service names in the database `SERVICE_NAMES` parameter.

1. If you are using a RAC database deployment, add a CRS-managed bootstrap service and the affinity services for the primary and standby databases by running the following commands:

   For the primary, run the following commands on any primary RAC node:

   ```
   srvctl add service -d pr_orcl -s beehivedg -r pr_orcl1,pr_orcl2
   srvctl add service -d pr_orcl -s aff1 -r pr_orcl1
   srvctl add service -d pr_orcl -s aff2 -r pr_orcl2
   ```

   For the standby, run the following commands on any standby RAC node:

   ```
   srvctl add service -d stby_orcl -s beehivedg -r stby_orcl1,stby_orcl2
   srvctl add service -d stby_orcl -s aff1 -r stby_orcl1
   srvctl add service -d stby_orcl -s aff2 -r stby_orcl2
   ```

2. If you are using a single-instance database deployment, create the managed service using the `dbms_service` package. In SQL*Plus, connect as SYS or SYSDBA and enter the following:

   ```
   begin
       dbms_service.create_service(service_name => 'beehivedg',
                                   network_name => 'beehivedg' );
     end;
      /
   ```

Setup Service Startup

When a switchover or failover to a standby database occurs, managed services are not automatically started. Perform the following steps to create a trigger to start up database-related services.

See Also: "Using Fast Application Notification Callouts" in Chapter 4, "Introduction to Automatic Workload Management" of the *Oracle Real Application Clusters Administration and Deployment Guide*

Setup Service Startup

1. In a RAC deployment, install a RAC callout script that will start the services when the database is opened.

Note: Instructions for setting up affinity services are included in the RAC post-install steps in the *Oracle Beehive Installation Guide* for your platform.

2. For a single instance deployment, create a database startup trigger on the primary database (Oracle Data Guard will subsequently replicate your trigger on the standby database). For example:

   ```
   CREATE OR REPLACE TRIGGER manage_services_start after startup on database

   DECLARE
     role VARCHAR(30);
   ```

```
BEGIN
  SELECT DATABASE_ROLE INTO role FROM V$DATABASE;
  IF role = 'PRIMARY' THEN
       DBMS_SERVICE.START_SERVICE('beehivedg');
  ELSE
       DBMS_SERVICE.STOP_SERVICE('beehivedg');
  END IF;
END;
/
```

3. To test the setup, you can restart the database (in a RAC deployment, restart the primary RAC node database). The related services should start automatically:

```
srvctl stop database -d pr_db1
srvctl start database -d pr_db1
srvctl status service -d pr_db1
```

Setup FAN ONS Publisher and Database Role Change Trigger

You must configure the FAN ONS Publisher and Database Role Change trigger in order to enable database failover operations. (In a "full deployment" scenario, the switchover or failover between database deployments includes a transition to a new set of Oracle Beehive Application tier nodes. Since the remote application tier does not receive FAN events, you can skip this step.)

> **See Also:** Pages 11-13 of "Client Failover in Data Guard Configurations for Highly Available Oracle Databases: Oracle Database 10*g* Release 2" Oracle Maximum Availability Architecture White Paper, found at the following link on the Oracle Technology Network website:
>
> `http://www.otn.oracle.com/goto/maa`

1. Configure the FAN ONS Publisher Configuration File in `$ORACLE_HOME/dbs/cfo${ORACLE_SID}.ora` (on both the primary and the standby deployments).

   For example (including affinity):

```
pr_orcl peer=stby_orcl
stby_orcl peer=pr_orcl
pr_orcl service=beehivedg location=pr_db1,pr_orcl1:pr_db2,pr_orcl2
pr_orcl service=aff1 location=pr_db1,pr_orcl1
pr_orcl service=aff2 location=pr_db2,pr_orcl2
stby_orcl service=beehivedg location=stby_db1,stby_orcl1:stby_db2,stby_orcl2
stby_orcl service=aff1 location=stby_db1,stby_orcl1
stby_orcl service=aff2 location=stby_db2,stby_orcl2
```

2. On all primary and standby RAC nodes, copy the `$ORACLE_HOME/dbs/cfo${ORACLE_SID}.ora` file to each node and rename it with the appropriate `ORACLE_SID`

3. Build a wrapper script around the Publisher program:

```
$ORACLE_HOME/bin/cfo.sh

#!/bin/ksh
export TZ=PST8PDT
export ORACLE_SID=pr_orcl1
export ORACLE_HOME=/u01/app/oracle/product/11.1.0/db_1
export LD_LIBRARY_PATH=$ORACLE_HOME/lib
export PATH=$ORACLE_HOME/bin:$PATH
$ORACLE_HOME/bin/cfo r
echo `date` "executed $0" > /tmp/cfo.log
```

Copy `cfo.sh` to each node, and edit each copy to change the `ORACLE_SID` value on each node appropriately.

4. Create a database role change trigger:

```
CREATE OR REPLACE TRIGGER ons_JDBCpublish AFTER DB_ROLE_CHANGE ON DATABASE
BEGIN
   dbms_scheduler.create_job(
   job_name=>'publish_events',
   job_type=>'executable',
   job_action=>'/u01/app/oracle/product/11.1.0/db/bin/cfo.sh',
   enabled=>TRUE
   );
END;
```

Configuring Oracle Beehive Application Tiers

Configure your Oracle Beehive servers by performing the following steps:

Notes:

- You only need to perform these steps on one Application tier in an Oracle Beehive deployment
- In steps 1, 2, and 3, if you have both a local and a remote standby, include connection strings for all three sets of database nodes
- You only need to configure affinity services (step 2) for RAC configurations

1. Update bootstrap connect_string to use service relocation:

   a. Run `beectl> stop --all`

   b. Run the following command:

```
beectl> modify_bootstrap_configuration --connect_string \
"(DESCRIPTION=(ADDRESS_LIST=(LOAD_BALANCE=yes)(FAILOVER=on)\
(ADDRESS=(PROTOCOL=TCP)(HOST=pr_db1_vip.example.com)(PORT=1521))\
(ADDRESS=(PROTOCOL=TCP)(HOST=pr_db2_vip.example.com)(PORT=1521))\
(ADDRESS=(PROTOCOL=TCP)(HOST=stby_db1_vip.example.com)(PORT=1521))\
(ADDRESS=(PROTOCOL=TCP)(HOST=stby_db2_vip.example.com)(PORT=1521)))\
(CONNECT_DATA=(SERVER=DEDICATED)(SERVICE_NAME=beehivedg.us.oracle.com)))"
```

2. Update the affinity service connect string:

Note: You only need to configure affinity services for RAC configurations. If you are not using RAC, you can skip this step.

```
beectl> modify_property --component _CURRENT_SITE:Database \
--name AffinityServiceNames \
--value '((DESCRIPTION=
(ADDRESS=(PROTOCOL=TCP)(HOST=pr_db1_vip.example.com)(PORT=1521))
```

```
(ADDRESS=(PROTOCOL=TCP)(HOST=stby_db1_vip.example.com)(PORT=1521))
(LOAD_BALANCE=yes)(FAILOVER=on)
(CONNECT_DATA=(SERVER=DEDICATED)(SERVICE_NAME=bhaffsvc1.us.oracle.com)))' \
--value '((DESCRIPTION=
(ADDRESS=(PROTOCOL=TCP)(HOST=pr_db2_vip.example.com)(PORT=1521))
(ADDRESS=(PROTOCOL=TCP)(HOST=stby_db2_vip.example.com)(PORT=1521))
(LOAD_BALANCE=yes)(FAILOVER=on) (CONNECT_DATA=(SERVER=DEDICATED)(SERVICE_
NAME=bhaffsvc2.us.oracle.com)))'
```

3. Update the ONS entries by modifying the ONS setup to include the standby nodes:

> **Note:** 6200 is the current setting for `remoteport` in the `$CRS_HOME/opmn/conf/ons.config` file and should also show in the `$CRS_HOME/bin/onsctl` ping output

```
beectl> modify_property --component _current_site:Database \
--name OnsNodeConfiguration \
--value pr_db1.example.com:6200 \
--value pr_db2.example.com:6200 \
--value stby_db1.example.com:6200 \
--value stby_db2.example.com:6200
```

4. Activate the Configuration on the first Oracle Beehive Application tier by running the following commands:

```
beectl> activate_configuration
beectl> modify_local_configuration_files
beectl> start --all
```

5. One at a time, run the following commands on each remaining Application tier instance:

```
beectl> stop -all
beectl> modify_local_configuration_files -restart_needed false
beectl> start --all
```

Performing Role Transitions

An Oracle Data Guard configuration consists of one database that functions in the primary role and one or more databases that function in the standby role. Typically, the role of each database does not change. However, if Data Guard is used to maintain service in response to a primary database outage, you must initiate a role transition between the current primary database and one standby database in the configuration.

A database operates in one of the following mutually exclusive roles: primary or standby. Data Guard enables you to change these roles dynamically by issuing SQL statements, or by using either of the Data Guard broker's interfaces (command line or Oracle Enterprise Manager).

> **See Also:** Chapter 1, "Oracle Data Guard Broker Concepts," of *Oracle Data Guard Broker*

Oracle Data Guard supports the following role transitions:

- **Switchover**

Allows the primary database to switch roles with one of its standby databases. There is no data loss during a switchover. After a switchover, each database continues to participate in the Data Guard configuration with its new role. Switchover is generally used for planned outages.

- **Failover**

  Changes a standby database to the primary role in response to a primary database failure. If the primary database was not operating in either maximum protection mode or maximum availability mode before the failure, some data loss may occur. If Flashback Database is enabled on the primary database, it can be reinstated as a standby for the new primary database once the reason for the failure is corrected. Failover is used when unplanned outages occur.

Performing a Switchover

To perform a database-only switchover operation follow the steps in MetaLink Note 751600.1.

To perform a switchover operation in a full-deployment scenario, perform the following steps:

1. Stop Oracle Beehive on the primary site application nodes:

   ```
   beectl> stop --all
   ```

2. Follow the steps in MetaLink Note 751600.1.
3. If this is a switchover in a full deployment configuration, disable the old primary site Application tiers and enable the new primary site Application tiers:

   a. Use the `beectl list_components` command to list all beehive instances:

      ```
      beectl> list_components --type BeehiveInstance
      ```

   b. Use the `beectl modify_property` command to disable each of the old Application tiers:

      ```
      beectl> modify_property --component <component_id> --name Status --value
      DISABLED
      ```

   c. Use the `beectl modify_property` command to enable each of the new Application tiers:

      ```
      beectl> modify_property --component <component_id> --name Status --value
      ENABLED
      ```

   d. Activate your changes by running the `beectl activate_configuration` command:

      ```
      beectl> activate_configuration
      ```

4. Start the Oracle Beehive Application tiers in your standby deployment:

   ```
   beectl> start --all
   ```

5. Once the standby Oracle Beehive system is up and running, if this is a full deployment, you can redirect user traffic to the new primary deployment.

Performing a Failover

To perform a database-only failover operation follow the steps in section 8.2.2, "Performing a Failover to a Physical Standby Database," in Chapter 8 of the *Oracle Data Guard Concepts and Administration Guide*.

To perform a failover operation in a full-deployment scenario, perform the following steps:

1. If possible, stop Oracle Beehive on the primary site application nodes:

   ```
   beectl> stop --all
   ```

2. Follow the steps in section 8.2.2, "Performing a Failover to a Physical Standby Database," in Chapter 8 of the *Oracle Data Guard Concepts and Administration Guide*
3. If this is a failover in a full deployment configuration, disable the old primary site Application tiers and enable the new primary site Application tiers:

   a. Use the `beectl list_components` command to list all beehive instances:

   ```
   beectl> list_components --type BeehiveInstance
   ```

   b. Use the `beectl modify_property` command to disable each of the old Application tiers:

   ```
   beectl> modify_property --component <component_id> --name Status --value
   DISABLED
   ```

   c. Use the `beectl modify_property` command to enable each of the new Application tiers:

   ```
   beectl> modify_property --component <component_id> --name Status --value
   ENABLED
   ```

   d. Activate your changes by running the `beectl activate_configuration` command:

   ```
   beectl> activate_configuration
   ```

4. Start the Oracle Beehive Application tiers in your standby deployment:

   ```
   beectl> start --all
   ```

5. Once the standby Oracle Beehive system is up and running, if this is a full deployment, you can redirect user traffic to the new primary deployment.

Tuning Options

This section includes the following topics

- Modifying Outbound Connect Timeout
- Enabling Listener Throttling on Each Database Node

Modifying Outbound Connect Timeout

You can modify the outbound connect timeout.

> **See Also:** Page 5 of "Client Failover in Data Guard Configurations for Highly Available Oracle Databases: Oracle Database 10*g* Release 2" Oracle Maximum Availability Architecture White Paper, found at the following link on the Oracle Technology Network website:
>
> `http://www.otn.oracle.com/goto/maa`

For example, to modify the outbound connect timeout to three seconds:

```
beectl modify_property --component _current_site:Database:DefaultNonXaPool \
```

```
--name ConnectionProperties \
--value oracle.net.ns.SQLnetDef.TCP_CONNTIMEOUT_STR:3000
```

Enabling Listener Throttling on Each Database Node

Listener throttling is a performance adjustment you can make to the database listener that may be necessary for the standby database to operate efficiently during and after a switchover or failover. When the system initially moves from the primary to the standby, there will be a very high number of connections moved all at once to the standby system, which can cause a "connection storm." Listener throttling is a method of controlling the number of simultaneous connections made to the database.

The appropriate value to use depends on the size of your deployment (number of concurrent users), and the capability of your database hardware.

> **See Also:** For complete details of this procedure, see Appendix E, "Listener Connection Rate Throttling," of "Optimizing Availability During Unplanned Outages Using Oracle Clusterware and RAC " Oracle Maximum Availability Architecture White Paper, found at the following link on the Oracle Technology Network website:
>
> `http://www.otn.oracle.com/goto/maa`

The `CONNECTION_RATE_<listener_name>` parameter indicates the connection rate per second for all addresses with `RATE_LIMIT` set to `YES`, `yes`, or a specific rate.

In this example, 10 connections will be spawned per second only for the first address in the address list. The `RATE_LIMIT` parameter can also specify the rate itself instead of only turning throttling on at the default rate. For example, `RATE_LIMIT=5` on a particular address indicates a specific setting for just that address. When specifying a `RATE_LIMIT` at the address level, you should not specify the `CONNECTION_RATE_<listener_name>` property.

> **Note:** If both the global rate and a specific rate are specified, the global rate is the one enforced.

Perform the following procedure to enable listener throttling on each database node:

1. Edit the `listener.ora` file. In Example 16–1, only the first address has throttling enabled. Note that the setting of 10 is an example; the actual setting depends on the details of your environment

Example 16–1 Example `listener.ora` ***File for Listener Throttling***

```
LISTENER_PR_DB1 =
(DESCRIPTION_LIST =
(DESCRIPTION =
(ADDRESS_LIST =
(ADDRESS = (PROTOCOL = TCP)(HOST = pr_db1_vip)(PORT = 1521)(RATE_
LIMIT=YES)(QUEUESIZE=1024)(IP = FIRST))
)
(ADDRESS_LIST =
(ADDRESS = (PROTOCOL = TCP)(HOST = <X.X.X.X>)(PORT = 1521)(IP = FIRST))
)
(ADDRESS_LIST =
(ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC))
)
)
)
CONNECTION_RATE_LISTENER_PR_DB1=30
```

Substitute the correct IP address for the corresponding non-VIP node (**pr_db1** in this example) for `<X.X.X.X>`.

2. Implement the `listener.ora` changes by running the following command:

   ```
   lsnrctl reload LISTENER_PR_DB1
   ```

17

Oracle Beehive Logging and Diagnosability

This module contains the following topics:

- Introduction
- Logging

Introduction

This module contains information about configuring logging options; monitoring and maintaining logs; and methodologies relating to troubleshooting and diagnosing issues for Oracle Beehive.

Complimentary information about error codes is available in the "Oracle Beehive Error Codes" module of the *Oracle Beehive Administrator's Reference Guide*

Logging

This section describes Oracle Beehive logging, and includes the following topics:

- Logging Architecture
- Understanding Log Entries
- Changing Log Levels
- Searching Through Logs
- Managing Logs

Logging Architecture

Oracle Beehive logs are located in the following directory:

```
$ORACLE_HOME/beehive/logs
```

A variety of directories and files will appear within the `$ORACLE_HOME/beehive/logs` directory. The folders represent different functional areas of Oracle Beehive. The sub-directories and files that are in your `logs` directory depend on the types of operations performed with your Oracle Beehive deployment.

Once a `beectl` command has been used, for example, a `beectl` sub-directory will appear under `$ORACLE_HOME/beehive/logs` directory. Within the `beectl` directory, directories represented using a date *MM.DD.YYYY* format will be created. Each *MM.DD.YYYY* directory represents the day on which the information was logged. A log file denoting the `beectl` command that was used will appear under the dated directory.

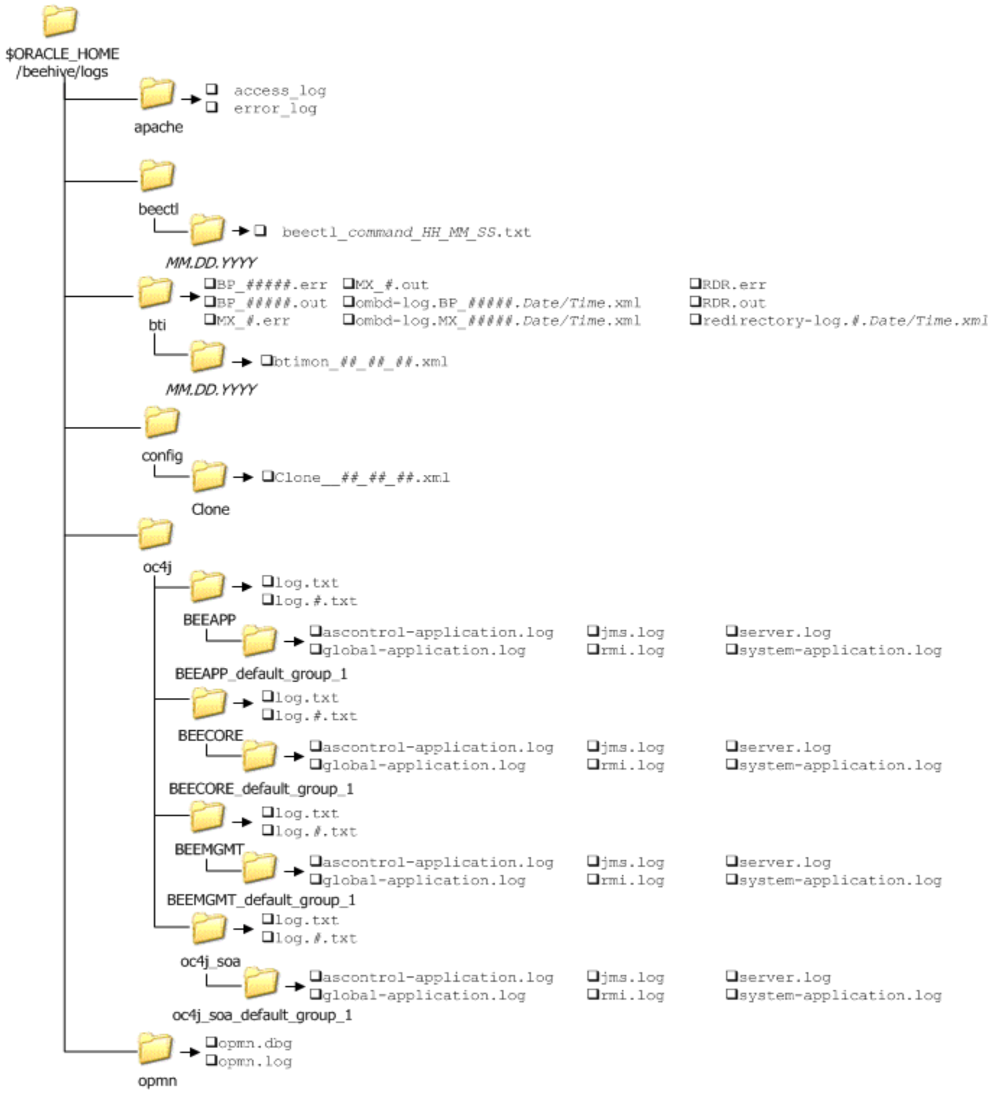
Within the `$ORACLE_HOME/beehive/logs/oc4j` directory, sub-directories are named after managed components and will typically include a `log.txt` file and a series of files named `log.txt.#`. The # in `log.txt.#` represents an archived version of the log file. The most recent log file is always `log.txt`.

For more information about log archiving options refer to the "Managing Logs" section.

Figure 17–1 outlines what you might typically expect to see in the `$ORACLE_HOME/beehive/logs` directory.

Note: The directory structure on your Oracle Beehive deployment may have additional or fewer files and directories.

Figure 17–1 Oracle Beehive Log Directory Structure



Note: The `$ORACLE_HOME/beehive/logs/config/clone` directory will only exist in the log directory structure in cloned environments.

Understanding Log Entries

This section contains information about log entries, including identifying the source of a message and its importance relative to the overall health of Oracle Beehive.

This section contains the following topics:

- Error Code Severities
- Error Message Examples

For a complete listing of Oracle Beehive error codes refer to "Oracle Beehive Error Codes" in *Oracle Beehive Administrator's Reference Guide*

Error Code Severities

Log entries contain information relating to a system action. The entries are not limited to highlighting critical errors in Oracle Beehive; they also serve to inform administrators of events that have occurred within the system.

Although log levels are set in a Java log level format, the severity level of messages that appear in Oracle Beehive logs are based on ODL message type log levels. For information about mappings of Java log level to ODL message type log levels refer to "Logging in OC4J" in the *Oracle Containers for J2EE Configuration and Administration Guide*.

Table 17–1 Oracle Beehive Log Level Severities

Type	Description
INTERNAL_ERROR	Oracle Beehive has experienced an error for internal or unexpected reasons. Oracle recommends reporting these errors to Oracle Support.
ERROR	Some problem that requires attention from the system administrator.
WARNING	Indicates that an action occurred or a condition was discovered that should be reviewed and may require action. This type of message may lead to a message of type ERROR.
NOTIFICATION	Reports a normal action or event. Could be a user operation, such as "login completed" or automatic operation such as a log file switch.
TRACE	A trace or debug message.

Error Message Examples

This section contains two examples of error messages, and explains the significance of the fields in each message.

Example 17–1 Example BEECORE Log File Error Message

```
[2008-03-21T01:50:12.417-07:00] [OJDL] [NOTIFICATION:16] []
[oracle.core.ojdl.FileLogWriter] [org: Acme] [host: myhost.domain.com] [nwaddr:
111.11.111.111] [tid: WorkExecutorWorkerThread-2] [userId: oracle] Deleted log
file: log.txt.30, size = 10485474 bytes
```

The log entry in Example 17–2 has 11 fields. It is important to note that not all error messages have the same number of fields, nor is the information in the same order. As a guideline, Table 17–3 explains the sequence of errors that appear in Example 17–2.

Table 17–2 Explanation of Error Message Fields in Example 17–2

Number	Name	Description
1	Date and time	Specifies the date and time, in ISO standard format, at which the error message was logged.
2	Source	Indicates the source of the message.

Table 17–2 (Cont.) Explanation of Error Message Fields in Example 17–2

Number	Name	Description
3	Log level	Indicates the log level of error message. For a complete list of log levels and their significance, refer to Table 17–1, " Oracle Beehive Log Level Severities".
4	Empty field	This field has no significance, and is always empty.
5	Module or class	Specifies the module or class that raised the error.
6	Organization	Indicates the organization.
7	Host	Indicates the host on which the error occurred.
8	Network address	Indicates the network address of the host on which the error occurred.
9	Thread ID	Specifies the thread ID.
10	User	Specifies the user ID performing the action.
11	Description	A description of the error message. This message will often include Oracle Beehive error code IDs, suspected causes and recommended actions. For a list of Oracle Beehive error code IDs, refer to "Oracle Beehive Error Codes" in *Oracle Beehive Administrator's Reference Guide*

Example 17–2 Example BEEAPP Log File Error Message

```
[2008-03-25T15:27:15.758-07:00] [beehive] [WARNING] []
[tm.service.timemanagement.task.CompositeDetectChangedTask] [tid: 34]
[ecid: 140.87.85.31:24335:1206484035703:96,0]
[bee_compid: 6d50fc8f-42c4-4140-802e-889cac3024cb]
[bee_compname: TimeManagementService] The Time Management Detect Composite
Changed Task received an error while processing composite information. The
failed operation will be retried. This may be an expected transient error unless
it recurs while the Beehive Database and Services are up and working.
```

The log entry in Example 17–2 has nine fields. It is important to note that not all error messages have the same number of fields, nor is the information in the same order. As a guideline, Table 17–3 explains the sequence of errors that appear in Example 17–2.

Table 17–3 Explanation of Error Message Fields in Example 17–2

Number	Name	Description
1	Date and time	Specifies the date and time, in ISO standard format, at which the error message was logged.
2	Source	Indicates the source of the message.
3	Log level	Indicates the log level of error message. For a complete list of log levels and their significance, refer to Table 17–1, " Oracle Beehive Log Level Severities".
4	Empty field	This field has no significance, and is always empty.
5	Thread ID	Specifies the thread ID.
6	Error code fingerprint	Specifies the fingerprint ID of the error message.
7	Oracle Beehive component ID	Specifies the identifier of the service in which the error message was generated.
8	Oracle Beehive component name	Specifies the name of the service in which the error message was generated.

Table 17–3 (Cont.) Explanation of Error Message Fields in Example 17–2

Number	Name	Description
9	Description	A description of the error message. This message will often include Oracle Beehive error code IDs, suspected causes and recommended actions. For a list of Oracle Beehive error code IDs, refer to "Oracle Beehive Error Codes" in *Oracle Beehive Administrator's Reference Guide*

Changing Log Levels

This section includes information about how to change log levels, and the circumstances under which they should be changed.

Log Levels

Table 17–4 outlines valid arguments when setting a log level in Oracle Beehive. The values that appear in the left column should be used when setting the log level using `beectl modify_property` command, whereas the value that appears in the right column represents the value that will appear in Oracle Beehive logs.

Table 17–4 Log Level Values Set vs. Values that Appear in the Oracle Beehive Logs

Value Used to Set Using `beectl`	Value that Appears in the Log
NULL	
SEVERE	ERROR:1
WARNING	WARNING:1
INFO	NOTIFICATION:1
CONFIG	NOTIFICATION:16
FINE	TRACE:1
FINER	TRACE:16
FINEST	TRACE:32

Listing Oracle Beehive Logging Properties

Oracle Beehive logging allows you to configure many logging properties to meet the needs of your deployment.

Use this command to list configurable Oracle Beehive logging properties:

```
beectl> list_properties --component _Current_site:LoggingProperties
```

A table similar to the following will be returned:

```
------------------+---------------------------------------------------------
Property name     | Property value
------------------+---------------------------------------------------------
Alias             |
------------------+---------------------------------------------------------
BaseLogLevel      | WARNING
------------------+---------------------------------------------------------
MaxFileSize       | 10485760
------------------+---------------------------------------------------------
MaxLogSize        | 104857600
------------------+---------------------------------------------------------
ModuleLogLevel    |
-----------------+----------------------------------------------------------
RotationFrequency | DAILY
------------------+---------------------------------------------------------
6 Record(s) displayed.
```

Changing Log Levels Globally

Follow these steps to change the log level for all Oracle Beehive services:

1. Determine the current value of the `BaseLogLevel` property. For instructions about listing current properties, refer to "Listing Oracle Beehive Logging Properties".

2. Modify the `BaseLogLevel` property to the desired value using the `modify_property` command:

   ```
   beectl> modify_property --component _CURRENT_site:LoggingProperties --name
   BaseLogLevel --value <log_level>
   ```

   Where *<log_level>* represents the log level that you want to assign to Oracle Beehive. For a list of valid arguments refer to Table 17–4.

   > **Note:** Oracle does not recommend setting global logging levels to `FINE`, `FINER`, or `FINEST`. To set a specific module to `FINE`, `FINER`, or `FINEST`, refer to "Changing Log Levels for Specific Modules".

3. Activate the configuration changes:

   ```
   beectl> activate_configuration
   ```

Changing Log Levels for Specific Modules

When troubleshooting, you may want to set a higher log level, such as `FINE`, `FINER`, or `FINEST`. These log levels should not be applied to Oracle Beehive globally due to the volume of log messages generated. Administrators can set a higher level of logging on a per module basis to assist in narrowing down a particular issue.

Follow these steps to change a log level for a particular module:

1. Determine the module or class for which you want to increase the log level. This information can be found in the log files.

   Typically, the module or class will be a string of characters separated by periods. In the following excerpt from the `$ORACLE_HOME/beehive/logs/BEEAPP/log.txt`, the `FRAMEwork.service.OnsReceiver` class is triggering the log message:

   ```
   [2008-03-25T15:27:05.581-07:00] [beehive] [WARNING] []
   [FRAMEwork.service.OnsReceiver] [tid: 11] [ecid:
   140.87.85.31:24335:1206483955043:3,0] Status of app 'cms-listener' cahnged from
   'PRESUMED_UNAVAILABLE' to 'INITIALIZING'
   ```

2. Modify the module's log level, using the following command:

   ```
   beectl> modify_property --component _Current_Site:LoggingProperties --name
   ModuleLogLevel --value oracle.ocs.<module>:<log_level>
   ```

Where <*module*> represents the module determined in Step 1, and <*log_level*> represents the log level that you want to assign to the Oracle Beehive module. For a list of valid arguments refer to Table 17–4.

> **Note:** When specifying the argument for the **--value** option, the module must be prefixed with `oracle.ocs`.

3. Activate the configuration changes:

   ```
   beectl> activate_configuration
   ```

You can use the beectl delete_property command to remove the map of ModuleLogLevel properties for a component:

```
beectl> delete_property --component _CURRENT_SITE:LoggingProperties --name
ModuleLogLevel
```

This command clears the property, setting it to null.

You can also append a value to the map, keeping the existing entries and adding new ones:

```
beectl> append_value --component _CURRENT_SITE:LoggingProperties --name
ModuleLogLevel --value "oracle.ocs.mail.service:CONFIG"
```

After making changes to any property, activate the configuration:

```
beectl> activate_configuration
```

Managing Logs

This section includes information about managing logs, including; controlling log archiving, and managing the size of logs and log directories.

Oracle Beehive log files are archived regularly based on the size of a log file, or the size of the directory. This section includes the following topics:

- Archiving Logs by File Size
- Archiving Logs by Directory Size

Archiving Logs by File Size

By default all Oracle Beehive log files are archived when the file size has reached 10485760 bytes.

To increase or decrease this value, for all Oracle Beehive log files:

1. Modify the argument of the `MaxFileSize` property using the following command:

   ```
   beectl> modify_property _CURRENT_site:LoggingProperties --name MaxFileSize
   --value <log_file_size>
   ```

Where <*log_file_size*> represents the size of file, in bytes, at which a log file should be archived.

2. Activate the configuration changes:

   ```
   beectl> activate_configuration
   ```

Archiving Logs by Directory Size

By default the `$ORACLE_HOME/beehive/logs` directory is archived when the directory size has reached 104857600 bytes.

To increase or decrease this value:

1. Modify the argument of the `MaxLogSize` property using the following command:

```
beectl> modify_property _CURRENT_site:LoggingProperties --name MaxLogSize
--value <log_directory_size>
```

Where <*log_directory_size*> represents the size, in bytes, at which `$ORACLE_HOME/beehive/logs` directory should be archived.

2. Activate the configuration changes:

```
beectl> activate_configuration
```

Searching Through Logs

This section includes information about searching through logs, including examples of the most common options you may want to use when searching through logs.

When an unexpected situation arises and the source of an error message has been determined, you may want to query logs for specific errors or strings. To accomplish this task, use the `beectl export_filesystem_logs` command.

The default number of search results returned when using the `export_filesystem_logs` command is 50. You can increase the number of results using the **--maximum_results** option.

Example 17–3 Searching All Log Records in the error_code Framework Module

```
beectl> export_filesystem_logs --search_string "(MODULE_
ID='cspi.OcsExceptionMetadata')" --display_source true
```

In Example 17–3 the `export_filesystem_logs` command is used to search for the string "(MODULE_ID='cspi.OcsExceptionMetadata')" in all log records in the error_code Framework Module. The **--display_source** option instructs the command to return the file in which the record appears.

Example 17–4 Limiting the Search to BEEAPP Logs

```
beectl> export_filesystem_logs --search_string "(MODULE_
ID='cspi.OcsExceptionMetadata')" --file_name_filter oc4j/beeapp --display_source
true
```

In Example 17–4 the `export_filesystem_logs` command is used to search for the string "(MODULE_ID='cspi.OcsExceptionMetadata')". The **--file_name_filter** indicates that the search should be performed exclusively in the $ORACLE_HOME/beehive/logs/oc4j/BEEAPP directory. The **--display_source** option instructs the command to return the file in which the record appears.

Example 17–5 Sending the Output of a Search Result to a File

```
beectl> export_filesystem_logs --search_string "(MODULE_
ID='cspi.OcsExceptionMetadata')" --display_source true -target_output
/tmp/temp.txt
```

In Example 17–5 the `export_filesystem_logs` command is used to search for the string "(MODULE_ID='cspi.OcsExceptionMetadata')" in all log records in the error_code Framework Module. The **--display_source** option instructs the command to return the file in which the record appears. The **--target_output** option indicates that the search results should be output a `temp.txt` file in the `/tmp` directory.

For a list of all options available when searching through logs using the `export_filesystem_logs` command, refer to "`export_filesystem_logs`" in the *Oracle Beehive Administrator's Reference Guide*.

A

Oracle Beehive Ports Reference

This appendix catalogues the ports that are commonly used by Oracle Beehive components.

To list ports currently in use by your Oracle Beehive deployment, run the `beectl list_ports` command:

```
beectl> list_ports
```

See Also:

- For information on configuring ports, see Chapter 4, "Oracle Beehive Property Reference" in the *Oracle Beehive Administrator's Reference Guide*
- For information on beectl commands useful for managing ports, see Chapter 2, "Oracle Beehive Command Line Utility" in the *Oracle Beehive Administrator's Reference Guide*

Oracle Beehive Ports

Table A–1, " Oracle Beehive Ports" lists the commonly-used Oracle Beehive ports, with the following information (where applicable):

- **Protocol**: the protocol used over the port or port range
- **Default port**: the port or port range used by Oracle Beehive in a default installation in which no port-conflicts were encountered and no custom port configuration was made
- **Virtual Port Support**: Virtual Port Support is used in High Availability environments when a load Balancer is listening on a different port number than the Application tiers. The "Virtual Port" is set to match the port number the load balancer is listening to, and then directs the request to the Application tier at the "Listening Port" value.
- **Accessible Through Firewall**: Indicates whether you typically would allow or block this port in your Internet firewall settings (to allow or prevent access from remote clients, for example).
- **Reference**: Additional documentation on the port or protocol available in the Oracle Beehive library
- **Notes**: Additional notes and recommendations

Table A–1 Oracle Beehive Ports

Protocol	Default Port	Virtual Port Support	Accessible Through Firewall	Reference	Notes
HTTP	7777	Yes	Yes	"Changing HTTP Port" in "Oracle Beehive Post-Installation Procedures" of the *Oracle Beehive Installation Guide* for your platform	
HTTPS	4443	Yes	Optional	"Changing HTTP Port" in "Oracle Beehive Post-Installation Procedures" of the *Oracle Beehive Installation Guide* for your platform	
BTP	21401	Yes	Yes		
BTPS	21451	Yes	Optional		
SMTP	25	Yes	Not recommended		Oracle does not recommend using Oracle Beehive SMTP service directly to the Internet. A third-party SMTP gateway such as a sendmail server is recommend as an intermediary.
SMTPS	Not set. Typical value is 465	Yes	Not recommended	"Configuing E-Mail with SSL" in the *Oracle Beehive Installation Guide* for your platform.	Oracle does not recommend using Oracle Beehive SMTP service directly to the Internet. A third-party SMTP gateway such as a sendmail server is recommend as an intermediary.
IMAP	143	Yes	Yes		
IMAPS	Not set. Typical value is 993	Yes	Optional	"Modifying or Adding an Endpoint in IMAP" in "Configuring E-Mail with SSL" in the *Oracle Beehive Installation Guide* for your platform.	
FTP command	2121	Yes	Yes	"Configuring FTP" in the *Oracle Beehive Installation Guide* for your platform	
FTP Data (Active Mode)	2120	n/a	If using active mode		
FTP Data (Passive Mode)	21000 to 21200	n/a	If using passive mode		The range depends on the value of the `MaxDCPortCount` property of the FTP Service; by default, 200.
XMPP	5222	Yes	Yes	"Configuring XMPP" in the the *Oracle Beehive Installation Guide* for your platform	
XMPPS	5223	Yes	Optional		
OWC	1554				
OWC	1954				
OWC	1935				
SIPP	5060				

Table A–1 (Cont.) Oracle Beehive Ports

Protocol	Default Port	Virtual Port Support	Accessible Through Firewall	Reference	Notes
Oracle Beekeeper HTTP	7779	Yes	Optional	"Changing Oracle Beekeeper Port," and "Configuring Virtual Host," in "Oracle Beekeeper Post-Installation Procedures" of the *Oracle Beehive Installation Guide* for your platform	
Oracle Beekeeper HTTPS	Not set	Yes		"Configuring Oracle Beekeeper for SSL Access" in the *Oracle Beehive Installation Guide* for your platform	
DMZ Ports				"Step B: Configuring Oracle Beehive DMZ Instances" of "Configuring Oracle Beehive Demilitarized Zone Instances" in the *Oracle Beehive Installation Guide* for your platform	
OPMN request port	6004		Yes		
OPMN remote port	6201		Yes		
AJP	12501 to 12504		Yes		
AJP for Zimbra	12505 to 12506		Yes		

Index