

Oracle® Beehive

Administrator's Guide

Release 1 (1.5)

E14836-04

August 2009

Oracle Beehive Administrator's Guide Release 1 (1.5)

E14836-04

Copyright © 2008, 2009, Oracle and/or its affiliates. All rights reserved.

Primary Authors: Andrew Mitchell, Joshua Stanley

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

Contributors: Henrik Blixt, Pradeep Chulliyar, Vikas Dhamija, Richard Hall, Duane Jensen, Ravi Jupudy, Rodrigo Lima, Tait McCarthy, Sudip Roy, Joe Paradise, Mark Paterson, Rajesh Parakkal, Gregory Pekofsky, François Perrault, Alain Petit, Jay Rajiva, 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 software or related documentation 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 RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software 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 which may create a risk of personal injury. If you use this software in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure the safe use of this software. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software in dangerous applications.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

This software 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

Preface	xix
Audience.....	xix
Documentation Accessibility	xix
Related Documents	xx
Conventions	xx
 1 Overview of Oracle Beehive Administration	
Getting Started with Oracle Beehive	1-1
Managing Oracle Beehive using beectl	1-1
Managing Oracle Beehive using Oracle Beekeeper	1-2
Managing Oracle Beehive using Clients.....	1-2
Your First Administration Tasks in Oracle Beehive	1-2
 2 Starting and Stopping Oracle Beehive	
Starting and Stopping Oracle Beehive	2-1
Overview of Starting and Stopping Oracle Beehive	2-1
Getting Started.....	2-1
Using beectl Commands in Shell Mode Without Authentication	2-2
Starting and Stopping Oracle Beehive Using the beectl Command-Line Utility	2-2
Starting	2-2
Stopping	2-2
Restarting	2-2
Starting and Stopping Oracle Beehive Managed Components Individually Using the beectl Command-Line Utility	2-3
Determining the Managed Component Identifier	2-3
Starting a Specific Managed Component	2-4
Stopping a Specific Managed Component	2-5
Restarting a Specific Managed Component.....	2-5
Starting and Stopping Oracle Beekeeper	2-6
 3 Managing and Provisioning Oracle Beehive Users	
Introduction to Managing and Provisioning Users with Oracle Beehive	3-1
About the User Directory Service (UDS)	3-2
About Organizations	3-2

About Personal Workspaces.....	3-2
About Provisioning and Deprovisioning Policies.....	3-3
About Using beectl to Manage User Accounts.....	3-3
About User Accounts	3-3
About User Account Fields.....	3-4
About User Account Principals and Credentials	3-9
About User Account Addresses.....	3-10
About User Account Attributes	3-11
About User Account Properties.....	3-11
About the Timezone Attribute with LDAP Synchronization.....	3-12
About Unique User Identifiers.....	3-13
About User <code>display_name</code> Values	3-13
About User Memberships.....	3-15
About Roles.....	3-17
About User Account Status	3-18
About Special and System-Reserved Accounts	3-18
Provisioning User Accounts	3-18
Provisioning User Accounts in Coexistence Environments	3-19
Provisioning User Accounts Using <code>beectl</code>	3-19
Bulk Provisioning User Accounts.....	3-21
Managing External Contacts	3-22
Managing User Accounts	3-23
Listing Users	3-23
Changing Status	3-25
Created Status.....	3-25
Enabled Status	3-26
Locked Status.....	3-26
Disabled Status.....	3-26
Marked for Delete and Deleted Statuses	3-27
Creating Custom User Properties.....	3-28
Modifying User Accounts.....	3-28
Managing Roles	3-29
Deleting User Accounts	3-29
Managing Groups	3-32
About Groups	3-32
Listing groups.....	3-34
Creating and Modifying Groups	3-35
Dynamic Group Query Construction.....	3-36
Queries Using Attributes	3-36
Queries Using Addresses.....	3-37
Queries Using Properties.....	3-38
Using Wildcards with Query Predicate Values.....	3-39
Query Predicate Operators.....	3-39
Example Queries	3-40
Group Inheritance	3-41
Example XML Files	3-41
Example Bulk User Provisioning XML File.....	3-42

4 Managing Oracle Beehive Resources

Introduction to Oracle Beehive Bookable Resources	4-1
About Oracle Beehive Bookable Resource Accounts	4-1
Oracle Beehive Bookable Resource Attributes	4-2
Creating Oracle Beehive Bookable Resource Accounts	4-3
Creating Oracle Beehive Bookable Resource Accounts Using beectl	4-3
Listing Oracle Beehive Bookable Resource Accounts	4-4
Listing Oracle Beehive Bookable Resource Accounts Using beectl	4-4
Modifying Oracle Beehive Bookable Resource Accounts	4-5
Modifying Oracle Beehive Bookable Resource Accounts Using beectl	4-6
Deleting Oracle Beehive Bookable Resource Accounts	4-6
Deleting Oracle Beehive Bookable Resource Accounts Using beectl	4-6
Managing Oracle Beehive Bookable Resource Approvers	4-7
Adding Oracle Beehive Bookable Resource Approvers Using beectl	4-7
Removing Oracle Beehive Bookable Resource Approvers Using beectl	4-7
Oracle Beehive Bookable Resource Booking Characteristics	4-8
Booking Characteristics Options	4-8
Setting Booking Characteristics	4-8

5 Managing Oracle Beehive Services

Introduction to Managing Oracle Beehive Services	5-1
About Oracle Beehive Services	5-2
Managing Oracle Beehive Core Services	5-2
Managing the Access Control Service	5-3
Related beectl Commands	5-3
Managing the Audit Service	5-4
Related beectl Commands	5-4
Managing the Authentication Services	5-4
Managing the Authentication Service	5-5
Related beectl Commands	5-5
Managing the Identity Provider Service	5-5
Related beectl Commands	5-5
Managing the Client Management Service	5-5
Related beectl Commands	5-5
Managing the Device Management Service	5-5
Related beectl Commands	5-5
Managing the Event Services	5-6
Managing the Event Service	5-7
Related beectl Commands	5-7
Managing the Object Event Publisher Service	5-7
Related beectl Commands	5-7
Managing the Management Service	5-7
Related beectl Commands	5-7
Managing the Policy Service	5-7
Related beectl Commands	5-7
Managing the Presence Service	5-8

Related beectl Commands	5-8
Managing the User Directory Service	5-8
Related beectl Commands	5-9
Managing the Workflow Service	5-10
Related beectl Commands	5-10
Managing Collaborative Services	5-10
Managing the Content Management Services	5-11
Managing the FTP Service	5-11
Related beectl Commands	5-11
Managing the WebDAV Service	5-11
Related beectl Commands	5-11
Managing the Discussions Service	5-11
Related beectl Commands	5-12
Managing the E-mail Service	5-12
Related beectl Commands	5-12
Managing the Fax Message Service	5-12
Related beectl Commands	5-12
Managing the Instant Message Services	5-13
Managing the Instant Message Service	5-13
Related beectl Commands	5-13
Managing the XMPP Service	5-13
Related beectl Commands	5-13
Managing the Meetings Services	5-13
Managing the Conference Artifact Service	5-14
Related beectl commands	5-14
Managing the Transcoding Service	5-14
Related beectl Commands	5-14
Managing the Time Management Services	5-14
Managing the Alarm Service	5-14
Related beectl Commands	5-14
Managing the CalDAV Service	5-15
Related beectl Commands	5-15
Managing the Resource Directory Service	5-15
Related beectl Commands	5-15
Managing the Time Management Service	5-15
Related beectl Commands	5-15
Managing the Time Zone Service	5-15
Related beectl Commands	5-15
Managing the Voice Message Service	5-16
Related beectl Commands	5-16
Managing the Zimbra Services	5-16
Managing the Zimbra Connector Service	5-16
Related beectl Commands	5-16
Managing the Zimbra UI Service	5-16
Related beectl Commands	5-17
Managing Enterprise Services	5-17
Managing the Mobility Services	5-17

Managing the Mobile Device Management Service	5-17
Related beectl Commands	5-18
Managing the Mobile Data Synchronization Service	5-18
Related beectl Commands	5-18
Managing the Mobile Mail Service.....	5-18
Related beectl Commands	5-18
Managing the Mobile Push Service.....	5-18
Related beectl Commands	5-18
Managing the Records Management Service.....	5-18
Related beectl Commands	5-19
Managing the Search Service.....	5-19
Related beectl Commands	5-19
Managing the Subscription and Notification Services	5-19
Managing the Notification Delivery Service.....	5-19
Related beectl Commands	5-19
Managing the Subscription Service.....	5-19
Related beectl Commands	5-19
Managing the SMPP Delivery Service	5-20
Related beectl Commands	5-20
Managing the Workspaces Service.....	5-20
Related beectl Commands	5-20
Managing Platform Services	5-21
Managing the Coexistence Service	5-21
Related beectl Commands	5-21
Managing the Platform Services	5-21
Managing the Platform Service.....	5-21
Related beectl Commands	5-21
Managing the Platform Web Service.....	5-21
Related beectl Commands	5-22

6 Managing Oracle Beehive Workspaces

About Workspaces.....	6-1
About Workspace Properties and Controls	6-3
About Workspace Events	6-5
Managing Personal Workspaces.....	6-8
Using Workspace Templates	6-9
About Workspace Templates	6-9
Using Expressions in Workspace Templates	6-12
Modifying Workspace Templates.....	6-12
Creating a New Workspace Template	6-13
Deleting a Workspace Template	6-13
Creating and Managing Team Workspaces	6-13
Creating Team Workspaces.....	6-13
Viewing Team Workspaces	6-15
Modifying Team Workspaces	6-16
Deleting Team Workspaces	6-16
Managing Categories.....	6-17

Managing Team Workspace Membership	6-19
Managing Team Workspace Access Control	6-20
Managing Team Workspace Visibility	6-20
Managing Team Workspace Roles	6-21
Managing Team Workspace Sensitivities	6-22
Managing Files.....	6-23
Managing File System Directories	6-23
Managing FTP and WebDAV Access to Files	6-25
Managing Records Management.....	6-25
Configuring Oracle Universal Records Manager	6-27
Registering URM in Oracle Beehive.....	6-27
Creating Retention Categories and Record Folders in URM.....	6-29
Setting Up Disposition Rules in URM	6-30
Enabling Record Filing of Sent E-mails	6-30
Filing Records of Artifacts in Oracle Beehive	6-31
Filing Records of Artifacts using beectl	6-31
Filing Records of Artifacts using Policies	6-31
Removing Records Management of Artifacts in Oracle Beehive	6-33
Troubleshooting Records Management Service Operations	6-33
Record Filing Failed	6-34
Removing Records Failed	6-35
Disposition Not Processed.....	6-35
URM Login, Password, or URL Incorrect or Changed.....	6-35
Configuring URM for Dispositions Testing.....	6-36
Example Workspace Template Contents	6-36

7 Managing Oracle Beehive Mobility Services

Introduction.....	7-1
Managing the Device Management Service	7-2
Uploading a Device Profile File	7-2
Customizing Device Profile Defaults	7-2
Adding a New Device Type to a Profile	7-4
Uploading and Provisioning a New Application.....	7-6
Managing the Mobile Data Sync Service	7-7
Listing Mobile Data Sync Service Configurable Properties.....	7-7
Controlling Synchronized Data Types	7-9
Controlling MD5 Authentication	7-10
Controlling MD5 Authentication for all Devices	7-10
Controlling MD5 Authentication for a Specific Device Profile.....	7-11
Controlling Synchronization Ranges	7-11
Managing the Mobile Mail Service	7-12
Listing Mobile Mail Service Configurable Properties.....	7-12
Controlling Maximum Number of E-mails Pushed to a Device	7-13
Controlling the Maximum Message Size.....	7-14
Controlling Past E-mail Push	7-14

8 Managing Oracle Beehive E-mail

Introduction to Oracle Beehive E-mail	8-1
About Configuring Oracle Beehive E-mail.....	8-1
Configuring Oracle Beehive E-mail Parameters.....	8-1
Configuring Oracle Beehive E-mail Rules using Oracle Beekeeper	8-1
About the Oracle Beehive SMTP Server	8-2
Managing Oracle Beehive E-mail Components	8-4
Configuring Sent E-mail Plugins	8-4
Configuring VMS Routing to Relay Messages	8-5
Configuring VMS Evaluator to Prevent Open Relay	8-6
Examples of Modifying the Inbound and Outbound VMSes	8-6
Configuring SMTP to Require Authentication	8-9
Using a Whitelist for E-mail Addresses.....	8-10
Setting Up E-mail Relay Routing.....	8-11
Configuring E-mail Archiving Rules	8-13
Adding a Virus Engine to Oracle Beehive.....	8-14
Managing Attachment Blocking and Virus Scanning	8-15
Managing Dead Letter.....	8-17
Setting and Modifying Multiple Endpoints	8-18
Delivering Remote E-mails with a Local E-mail Domain	8-19
Configuring Dispatch Rules	8-19
Specifying a Local Users Domain	8-20
E-mail Coexistence in a Single Domain	8-21

9 Managing Oracle Beehive Subscriptions and Notifications

Configuring User Subscriptions	9-1
Configuring Notifications to use SMS	9-4
Configuring SMS using SMPP	9-4
Configuring SMS using XMS.....	9-5

10 Managing Oracle Beehive Time Management

Managing Holidays	10-1
Creating an iCalendar File	10-1
Sample Holiday Entries.....	10-2
Importing Holiday iCalendar Files	10-3
Modifying User Time Management Preferences	10-4
Changing a User's Time Zone Preference	10-4
Changing a User's Defined Working Hours	10-5
Oracle Beehive Time Zone Definitions	10-6
Overview	10-6
Obtaining a New Oracle Beehive Time Zone Package	10-6
Refreshing the Oracle Beehive Time Zone Package.....	10-6

11 Managing Oracle Beehive Voicemail and Fax

Introduction to Managing Oracle Beehive Voicemail	11-1
--	------

About Facilities.....	11-1
About Auto Attendants.....	11-2
About Voicemail Infrastructure	11-2
Cisco Dependencies and Requirements	11-3
Voicemail UDS Requirements.....	11-3
Voicemail Preference Properties.....	11-3
Enterprise Preference Properties	11-3
Facility Preference Properties	11-3
User Preference Properties	11-4
Configuring Oracle Beehive Voice Message Service	11-4
Voicemail Properties.....	11-4
Configuring the Enterprise	11-11
Configuring Enterprise Preferences	11-11
Creating Voicemail Users.....	11-12
Managing Facilities	11-12
Creating a Facility	11-12
Sample Facility XML File	11-13
Configuring the Voicemail Touch-tone User Interface (TUI).....	11-16
Enabling HTTPS for Cisco VXML Enabled Device Access to Oracle Beehive.....	11-16
Configuring Cisco IP Phone Voicemail GUI Application	11-18
Configuring the Voicemail GUI and Message Waiting Indicator.....	11-22
Cisco Router Configuration.....	11-23
Configure Translation Rule	11-23
Configure Global VXML Configuration Options.....	11-24
Configure Voicemail VXML Application	11-25
Configure Transcoding (Optional)	11-25
Cisco Unified Call Manager Configuration.....	11-26
Configuring the Auto Attendant	11-26
Installing an Auto Attendant	11-26
Associating an Auto Attendant with a Facility	11-27
Voicemail TUI Optional Steps.....	11-27
Auto Attendant Administration Commands.....	11-28
Configuring Oracle Beehive Fax	11-28

12 Managing Oracle Beehive Events, Policies, and Workflows

Introduction to Beehive Events, Policies, and Workflows	12-1
Introduction to Beehive Events	12-2
Synchronous Events	12-2
Asynchronous Events.....	12-2
Event Subscriptions	12-2
Disabled Events.....	12-3
Introduction to Beehive Policies	12-4
Introduction to Beehive Workflows	12-4
Managing Beehive Events	12-5
Managing Beehive Policies	12-6
Managing the Provisioning Policy	12-6
Managing the Deprovisioning Policy	12-11

Managing the Password Policy	12-13
Managing Audit Policies.....	12-16
Creating and Managing Custom Policies	12-16
Creating New Policies	12-16
Editing Existing Policies.....	12-17
Deleting Custom Policies.....	12-18
Using Dynamic Policy Attributes	12-18
Managing Beehive Workflows	12-19
Types of Oracle Beehive Workflow	12-19
Oracle Beehive Workflow Tasks	12-20
Oracle Beehive Workflow Templates.....	12-20
Using Dynamic Attributes with Custom Workflow Templates.....	12-26
Oracle Beehive Workflow Template Commands.....	12-27

13 Managing Oracle Beehive Access Control

About Access Control	13-1
Explicit Access Control.....	13-1
Implicit Access Control	13-2
Managing Privileges	13-3
Managing Roles	13-4
About Role Definitions	13-5
Creating Role Definitions.....	13-6
Creating Assigned Roles	13-8
Modifying Roles	13-8
Deleting Roles	13-9
Creating and Managing Access Control Entities and Sensitivities.....	13-9
Creating and Managing ACEs	13-9
Viewing ACEs	13-10
Creating New ACEs	13-10
Modifying ACEs.....	13-10
Deleting ACEs	13-10
Creating and Managing Sensitivities	13-10
Viewing Sensitivities	13-11
Creating New Sensitivities	13-12
Modifying Sensitivities	13-12
Deleting Sensitivities	13-13

14 Managing Oracle Beehive Auditing

About Oracle Beehive Auditing.....	14-1
About Audit Events	14-2
About Audit Templates.....	14-2
About Audit Policies.....	14-2
About Audit Trails	14-2
Creating and Managing Oracle Beehive Audit Templates.....	14-3
Listing Audit Templates.....	14-3
Creating Audit Templates	14-4

Deleting Audit Templates	14-5
Example Audit Templates	14-5
Creating and Managing Oracle Beehive Audit Policies	14-5
Creating Audit Policies	14-6
Listing Audit Policies	14-7
Modifying Audit Policies	14-8
Enabling and Disabling Audit Policies	14-8
Deleting Audit Policies	14-8
Example Audit Policy	14-9
Creating and Managing Oracle Beehive Audit Trails	14-9
Listing Audit Trails	14-10
Creating Audit Trails	14-10
Modifying Audit Trails	14-11
Exporting Audit Trails	14-11
Validating Audit Trails	14-12
Deleting Audit Trails	14-12
Example Audit Trail	14-12
Audit Events Structure	14-15

15 Configuring and Managing Oracle Collaboration Coexistence Gateway

Introduction to Oracle Collaboration Coexistence Gateway	15-1
Terminology	15-2
Concepts	15-3
Architectural Overview	15-4
Installation and Post-Installation Configuration	15-4
Installation	15-4
Post-Installation Configuration	15-5
Post-Installation Tasks	15-6
Adjusting Quota Restrictions for the Oracle Connector for Exchange Mailbox	15-6
Installing the Oracle Change Notification Service for Exchange on the Same Host as Microsoft Exchange	15-6
Redirecting E-mail Back to Microsoft Exchange for Delivery to Oracle Beehive	15-7
Administering Oracle Collaboration Coexistence Gateway	15-11
Oracle Beehive Coexistence Service Administrative Tasks	15-12
Configuring a Coexistence System on Oracle Beehive	15-12
Listing Oracle Beehive Remote Coexistence Systems	15-13
Modifying Oracle Beehive Remote Coexistence Systems	15-14
Provisioning Users for Coexistence	15-15
Viewing the Registration State of Coexisting Users	15-16
Deprovisioning Users from an Oracle Beehive Remote Coexistence System	15-18
Oracle Connector for Exchange Administrative Tasks	15-19
Best Practices for Stopping and Starting Oracle Connector for Exchange	15-19
Stopping Oracle Connector for Exchange	15-20
Starting Oracle Connector for Exchange	15-20
Stopping the BEECONNECTOR OC4J Instance	15-20
Starting the BEECONNECTOR OC4J Instance	15-20
Configuring Oracle Connector for Exchange to use HTTPS	15-21

Oracle Change Notification Service for Exchange Administrative Tasks	15-21
Stopping Oracle Change Notification Service for Exchange	15-21
Starting Oracle Change Notification Service for Exchange	15-22
16 Backing Up and Recovering Oracle Beehive	
Introduction to Backing Up and Recovering Oracle Beehive	16-1
When to Perform Backups	16-1
Backing Up Oracle Beehive	16-2
Creating a Baseline Backup of Oracle Beehive	16-2
Creating a Baseline Backup of the Application Tier	16-3
Creating a Baseline Backup of the Database Tier	16-4
Performing a Closed (Cold) Backup of Oracle Beehive.....	16-4
Performing an Online (Hot) Backup of Oracle Beehive Database	16-5
Performing an Online Backup using SQL Commands.....	16-5
Performing an Online Backup using RMAN.....	16-6
Recovering Oracle Beehive	16-6
Recovering Oracle Beehive from a Baseline Backup.....	16-6
Recovering the Application Tier from a Baseline Backup	16-6
Recovering the Database Tier from a Baseline Backup	16-7
Recovering the Oracle Beehive Application Tier from a Closed (Cold) Backup	16-7
Recovering the Database Tier from an Online (Hot) Backup	16-7
17 Oracle Beehive Disaster Recovery with Data Guard	
Prerequisites	17-1
Configuring Oracle Beehive for Disaster Recovery	17-1
Recording Environment Details.....	17-3
Configuring Service Relocation and Database Triggers.....	17-4
Create Managed Services.....	17-4
Create Database Services Startup Trigger	17-5
Setup FAN ONS Publisher and Database Role Change Trigger.....	17-7
Configuring Oracle Beehive Application Tiers.....	17-7
Performing Role Transitions	17-8
Tuning Options	17-9
Modifying Outbound Connect Timeout.....	17-10
Enabling Listener Throttling on Each Database Node	17-10
18 Oracle Beehive Logging and Diagnosability	
Introduction	18-1
Logging	18-1
Logging Architecture.....	18-1
Understanding Log Entries.....	18-3
Error Code Severities.....	18-3
Error Message Examples.....	18-3
Changing Log Levels	18-5
Log Levels	18-5
Listing Oracle Beehive Logging Properties.....	18-5

Changing Log Levels Globally.....	18-6
Changing Log Levels for Specific Modules	18-6
Managing Logs	18-7
Archiving Logs by File Size	18-7
Archiving Logs by Directory Size.....	18-8
Searching Through Logs	18-8

Index

List of Figures

8-1	Configuring VMS Routing Rules.....	8-7
8-2	Configuring the Outbound VMS Sender Matcher	8-8
8-3	Configuring the Inbound VMS Evaluator.....	8-8
8-4	Configuring Local E-mail Patterns.....	8-9
8-5	Configuring OutboundVMS Authentication.....	8-10
8-6	Configuring an E-mail Whitelist.....	8-11
8-7	Configuring E-mail Relay Routing.....	8-12
8-8	Selecting Accept Invalid Local Users Checkbox	8-19
9-1	My Notifications Preferences Page.....	9-2
11-1	Oracle Beehive Voicemail Centralized Deployment	11-2
15-1	Oracle Collaboration Coexistence Gateway Architecture	15-4
17-1	Oracle Beehive Full-Deployment Disaster Recovery Scenario.....	17-2
17-2	Oracle Beehive Database-Only Recovery Scenario.....	17-3
18-1	Oracle Beehive Log Directory Structure.....	18-2

List of Tables

2-1	Managed Component Services	2-4
3-1	Default User Account Fields	3-6
3-2	Oracle Beehive User Account Address Field Schemes.....	3-10
3-3	display_name Generation Tokens	3-14
4-1	Oracle Beehive Bookable Resource Attributes	4-2
6-1	Workspace Related Business Events	6-6
6-2	Summary of Default Team Workspace Roles	6-22
6-3	Summary of Default Application-Level Roles.....	6-22
6-4	Artifact Metadata Sent to URM with All Artifacts.....	6-26
6-5	Artifact Metadata Sent to URM with E-Mail Messages	6-27
7-1	<Configuration> Attributes in a Device Profile File.....	7-3
7-2	Data Type Properties.....	7-9
9-1	User Subscription Entities and Rules.....	9-3
11-1	Voicemail Properties.....	11-5
11-2	Cisco IP Phone Recommended Deployment Properties	11-19
11-3	Voicemail Properties.....	11-23
12-1	User Attributes in Provisioning and Deprovisioning Policy Conditions	12-7
12-2	Oracle Beehive Workflow Template Keys.....	12-21
13-1	Default Oracle Beehive Privileges	13-3
14-1	Pre-Seeded Audit Templates.....	14-3
14-2	Access Control Events	14-15
14-3	Address Book Events.....	14-15
14-4	Artifact Events.....	14-16
14-5	Notification Events	14-17
14-6	Audit Events	14-18
14-7	BPEL Task Events	14-18
14-8	Calendar Events	14-19
14-9	Client Application Events.....	14-19
14-10	Device Management Events	14-20
14-11	Enterprise Events	14-20
14-12	LDAP Sync Profile Events	14-20
14-13	Message Events	14-21
14-14	Organization Events	14-21
14-15	Policy Subscription Events	14-22
14-16	Records Management Events	14-22
14-17	Search Events.....	14-22
14-18	Security Events	14-23
14-19	Service Configuration Update Events.....	14-23
14-20	System Events.....	14-23
14-21	Time Management Events	14-24
14-22	User Management Events.....	14-24
14-23	Workspace Events.....	14-25
14-24	XMPP Events	14-25
18-1	Oracle Beehive Log Level Severities	18-3
18-2	Explanation of Error Message Fields in Example 18-2	18-3
18-3	Explanation of Error Message Fields in Example 18-2	18-4
18-4	Log Level Values Set vs. Values that Appear in the Oracle Beehive Logs	18-5

List of Examples

3-1	Creating a single user using beectl	3-20
3-2	Dynamic Group Simple Attribute Query	3-40
3-3	Dynamic Group Simple Attribute Query with Wildcards	3-40
3-4	Dynamic Group Simple Attribute Query Using CollabIDs.....	3-40
3-5	Dynamic Group Complex Query	3-41
3-6	Bulk User Provisioning XML File.....	3-42
6-1	Adding Members to a Team Workspace During Creation.....	6-14
6-2	Example Team Workspace	6-15
6-3	Example Category XML File	6-17
6-4	Example Category with Attributes XML File	6-17
6-5	Sample Team Workspace Adding Members XML File.....	6-19
6-6	Sample Records Management Policy XML File	6-32
6-7	Example Workspace Template XML File	6-36
7-1	Disabling the Contacts Synchronization Data Type	7-9
7-2	Enforcing a Four Week Data Synchronization Limit.....	7-12
7-3	Enforce a Maximum Number of E-Mails to Push to a Device	7-13
7-4	Modify the Maximum Message Size.....	7-14
7-5	Modify Past E-mail Push	7-14
10-1	Sample Holiday Event Recurring on a Fixed Date	10-2
10-2	Sample Holiday Event Recurring on a Variable Date	10-2
10-3	Importing Holiday Events.....	10-3
10-4	Changing a User's Time Zone.....	10-4
10-5	Changing a User's Working Hours	10-5
10-6	Importing a Time Zone Definition Package.....	10-7
11-1	Sample Facility XML File.....	11-14
11-2	Sample Modifying Facility XML File	11-15
11-3	Sample Fax User Event Subscription	11-29
12-1	Provisioning Policy with Customized Conditions.....	12-8
12-2	Provisioning Policy with Complex Customized Conditions	12-10
12-3	Deprovisioning Policy with Customized Conditions.....	12-12
12-4	Default Password Policy	12-14
12-5	Serial Workflow Template XML File	12-23
12-6	Parallel Vote Workflow Template XML File.....	12-25
14-1	Simple Audit Policy	14-9
14-2	Example Audit Trail	14-12
14-3	Example Exported Audit Trail Data	14-12
15-1	Simple Rule for Rewriting E-mail Domains.....	15-8
15-2	Composite Rule for Rewriting Multiple E-mail Domains	15-9
15-3	Provisioning a User for Coexistence	15-15
15-4	Deprovisioning a User for Coexistence	15-18
17-1	Example CRS-Managed Service Startup Script With Affinity Services StartupService.sh	17-5
17-2	Example CRS-Managed Service Startup Script Without Affinity Services StartupService.sh	17-6
17-3	Example listener .ora File for Listener Throttling	17-11
18-1	Example BEECORE Log File Error Message.....	18-3
18-2	Example BEEAPP Log File Error Message.....	18-4
18-3	Searching All Log Records in the error_code Framework Module.....	18-8
18-4	Limiting the Search to BEEAPP Logs.....	18-8
18-5	Sending the Output of a Search Result to a File	18-8

Preface

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

Audience

The *Oracle Beehive Administrator's Guide* is directed at any administrator whose task is the installation, deployment, configuration, administration, and maintenance of Oracle Beehive.

Documentation Accessibility

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

Accessibility of Code Examples in Documentation

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

Accessibility of Links to External Web Sites in Documentation

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

Deaf/Hard of Hearing Access to Oracle Support Services

To reach Oracle Support Services, use a telecommunications relay service (TRS) to call Oracle Support at 1.800.223.1711. An Oracle Support Services engineer will handle technical issues and provide customer support according to the Oracle service request process. Information about TRS is available at <http://www.fcc.gov/cgb/consumerfacts/trs.html>, and a list of phone numbers is available at <http://www.fcc.gov/cgb/dro/trsphonebk.html>.

Related Documents

For more information, see the following documents in the Oracle Beehive Release 1.0 documentation set:

- *Oracle Beehive Administrator's Reference Guide*
- *Oracle Beehive Application Developer's Guide*
- *Oracle Beehive Concepts*
- *Oracle Beehive Deployment Guide*
- *Oracle Beehive Installation Guide for Linux*
- *Oracle Beehive Installation Guide for Microsoft Windows*
- *Oracle Beehive Installation Guide for Solaris Operating System (SPARC 64-Bit)*

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.
<i>italic</i>	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
<code>monospace</code>	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

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 should first ensure that your installation is complete, and then perform any necessary post-install tasks.

Post-install tasks might include any 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 listing of post-installation tasks, see the "Oracle Beehive Post-Installation Procedures" module 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/bee hive/bin`

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

`beectl command --option <argument>`

Where *command* represents the `beectl` command, *--option* represents an available option(s) to use with the command, and *<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. Refer to 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.

Refer to Chapter 27, Oracle Beehive Beekeeper Post-Installation Procedures in the *Oracle Beehive Installation Guide* for your platform, for details.

Managing Oracle Beehive using Clients

Some administration tasks may be accomplished by 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, and how to learn more about doing each of them:

- 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.

- Enable Oracle Collaboration Coexistence Gateway, and provision users for coexistence.

See "[Configuring and Managing Oracle Collaboration Coexistence Gateway](#)" 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.

Starting and Stopping Oracle Beehive

This module describes how to start and stop Oracle Beehive, managed components, and processes using the `beectl` command-line utility. It contains the following sections:

- [Starting and Stopping Oracle Beehive](#)
- [Starting and Stopping Oracle Beekeeper](#)

Starting and Stopping Oracle Beehive

The following topics are covered in this section:

- [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.

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. The `beectl` utility can be invoked every time you run a command, or you can use it in 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/bee hive/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

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

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

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
BEEEMGMT_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
	Workflow Service
	XMPP Service
BEEMGMT	Management Service
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 <componentIDn>`

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"](#).

- 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 start or stop the OC4J component, by 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:

- From the command line on the machine where Oracle Beekeeper is installed, start `opmnctl`:

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

- 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
```

- 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...
```

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 opmnctl:

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

Managing and Provisioning Oracle Beehive Users

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

This module contains the following topics:

- [Introduction to Managing and Provisioning Users with Oracle Beehive](#)
- [About User Accounts](#)
- [Provisioning User Accounts](#)
- [Managing User Accounts](#)
- [Deleting User Accounts](#)
- [Managing Groups](#)
- [Example XML Files](#)

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 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 "Integrating and Synchronizing LDAP with Oracle Beehive" in the *Oracle Beehive Installation Guide* for your platform.

This section contains 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. (In Oracle Beehive Release 1, 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 that 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 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; and 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, Policies, and Workflows"](#).

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 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 people 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 that this people-type serves is for 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.

Notes:

- 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 should 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: ["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)	<p>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.</p> <p>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.</p>

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

Attribute	Required or Optional	Field Type	Details
--login_password	Required	Credential	<p>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 <code>beectl</code> shell mode, you may enter the password in plaintext. If you are using single-command mode, you must obfuscate the password using the <code>beectl obfuscate</code> feature.</p> <p>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.</p>
--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	<p>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 <code>beectl list_timezones</code> command.</p> <p>Note that users can update their own timezone attribute from various clients and preference pages.</p>
--locale	Optional	Attribute	You may enter a locale identifier corresponding to the user's home locale. The locales should be in the format <code>x[_y]</code> where <code>x</code> is per per ISO639 and <code>y</code> is per ISO3166. For example, the United States English locale identifier is <code>en_US</code> .
--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 3-11 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 composed only of numbers allows the user to authenticate with a voice mail system using a touch tone telephone; this is called the voice principal.

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; 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
ORAAALERT	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 that 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, the user is unable to change their `given_name` or `job_title` attributes; only the administrator can do that.

When you are making use of LDAP-based directory synchronisation, 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 Beehive Central user configuration page; and so forth) 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 below).

The `display_name_format` should 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 <code>display_name</code>
<code>\$G</code>	<code>given_name</code>
<code>\$M</code>	<code>middle_name</code>
<code>\$F</code>	<code>family_name</code>
<code>\$g</code>	First character of <code>given_name</code>
<code>\$m</code>	First character of <code>middle_name</code>
<code>\$f</code>	First character of <code>family_name</code>
<code>\$P</code>	<code>prefix</code>
<code>\$S</code>	<code>suffix</code>
<code>\$J</code>	<code>job_title</code>
<code>\$N</code>	<code>nickname</code>

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.

Notes:

- 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; there is no pre-seeded value on install.

About User Memberships

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:F4AE2024BD5F4D1D9A7AC00C92AB964C00000000011
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 is created automatically: the BEEadmin account. 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 ["Integrating and Synchronizing LDAP with Oracle Beehive"](#) in the *Oracle Beehive Installation Guide* for your platform.

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 ["Configuring and Managing Oracle Collaboration Coexistence Gateway"](#).

Provisioning User Accounts Using beectl

Note: This section describes how to create an Oracle Beehive user 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. Note that 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  
--family_name User1FamilyName
```

```

--given_name User1GivenName
--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, Policies, and Workflows"](#).

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.

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 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"](#) on page 3-29 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 contains the following topics:

- [Listing Users](#)
- [Changing Status](#)
- [Creating Custom User Properties](#)
- [Modifying User Accounts](#)
- [Managing Roles](#)

Listing Users

You have many options for listing and searching for 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, Policies, and Workflows"](#).

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 [Deleting User Accounts](#) on page 3-29. 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. Depending on the password policy, an account may become locked temporarily (for a set period) or permanently (until unlocked directly by an administrator). You can change this behavior by modifying the password policy. For information about changing the password policy, see ["Managing Oracle Beehive Events, Policies, and Workflows"](#).

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). You could create a custom workflow that is triggered whenever an account is set to locked status, to automate the unlocking of user accounts. For information about creating custom workflows, see ["Managing Oracle Beehive Events, Policies, and Workflows"](#), and *Oracle Beehive Application Developer's Guide*.

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 or workflow 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 ["Managing Oracle Beehive Events, Policies, and Workflows"](#).

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. However, all of the account's data remains stored in the system. This may be convenient for compliance and records retention purposes.

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 3-29.

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"](#) on page 3-6 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 Step 5 below. 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 workaroud 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 openldaprofile
--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_team_workspace`

3. If there are any workflows currently requiring the attention of the user, you may need to cancel or modify each workflow. See ["Managing Oracle Beehive Events, Policies, and Workflows"](#) for details on how to modify existing workflows.

4. 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> }
```

5. At your convenience, purge user data, 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](#)
- [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 ["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, meaning, 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 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*.

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 cannot create dynamic groups based on custom user account properties. You can only use the default properties when constructing 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
- orattytd
- 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:DCEDF8D6310B4AE69911C7607BBADBE400000000000</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:DCEDF8D6310B4AE69911C7607BBADBE400000000000</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/bee hive/templates/uds/Users.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
 - 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
 - In the XML file, specify all extended enterprise users first, and then enterprise users.
-

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>
    <familyname>Holmes</familyname>
    <givenname>Robert</givenname>
    <name>Robert</name>
    <principals>
      <add>
        <principal>
          <name>+16505551234</name>
          <password>8675309</password>
          <type>VOICE</type>
        </principal>
      </add>
      <add>
        <principal>
          <name>rholes</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:rholes@example.com</address>
        <addresstype>BUSINESS_1</addresstype>
    </item>
</add>
<add>
    <item>
        <address>MAILTO:rholes@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>
    <memberships>
        <add>
            <item>
                <cen>orgn=Accounting,enpr=MyEnterprise</cen>
            </item>
        </add>
    </memberships>
    <user_type>ENTERPRISE_USER</user_type>
    <familyname>Langdown</familyname>
    <givenname>Mary</givenname>
    <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>BUSINESS1</addresstype>
            </item>
        </add>
    </addresses>

```

```
<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>
  <memberships>
    <add>
      <item>
        <cen>orgn=Accounting,enpr=MyEnterprise</cen>
      </item>
    </add>
  </memberships>
  <user_type>ENTERPRISE_USER</user_type>
  <familyname>Tran</familyname>
  <givenname>Thanh</givenname>
  <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>BUSINESS1</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>
```

Managing Oracle Beehive Resources

This module describes how to manage bookable resource accounts 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 bookable resources. This module includes the following sections:

- [Introduction to Oracle Beehive Bookable Resources](#)
- [Creating Oracle Beehive Bookable Resource Accounts](#)
- [Listing Oracle Beehive Bookable Resource Accounts](#)
- [Modifying Oracle Beehive Bookable Resource Accounts](#)
- [Deleting Oracle Beehive Bookable Resource Accounts](#)
- [Oracle Beehive Bookable Resource Booking Characteristics](#)

Introduction to Oracle Beehive Bookable 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 search for and reserve bookable resources through the calendar functions provided by the Time Management Service. Bookable resource accounts can also be used to create calendars for tracking related enterprise-wide information, such as employees' travel schedules.

This section includes the following topics:

- [About Oracle Beehive Bookable Resource Accounts](#)
- [Oracle Beehive Bookable 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 System Administrators or Business Administrators.

Bookable resources can be set up to permit reservations on a first come first served basis to prevent double-bookings or to permit more than one reservation at a time.

Oracle Beehive Bookable Resource Attributes

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

Table 4–1 Oracle Beehive Bookable 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, to limit its visibility and use to only users within that Organization.	No	A valid enterprise or organization ID
Resource ID	A resource ID is generated when the bookable resource is created. A resource ID cannot be specified by administrators at the time of creation, nor modified thereafter.	Yes	N/A
Resource identifier	Specifies a bookable resource identifier. 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 1000.
Display name	Specifies the display name for the bookable resource account. This will be the name displayed when searching for a bookable resource. Although a unique name is not required, it can be helpful to users when searching for bookable resources using Oracle Beehive clients.	Yes	An alphanumeric string Note: The maximum number of characters allowed is 1000.
Description	Describes any additional characteristics of the bookable 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 Global Positioning System (GPS) coordinates, name, description, time zone and markers. More than one location can be set for a bookable resource. For example, a vehicle that is shared by two offices in the same city. Depending on the date and time, the GPS coordinates may be different.	No	An alphanumeric string
Phone	Specifies the phone number of the bookable resource.	No	A valid telephone number
Fax	Specifies the fax number of the bookable resource.	No	A valid fax number
Web page	Specifies the Web page of the bookable resource.	No	An alphanumeric string

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

Attribute Name	Description	Required Attribute	Accepted Values
E-mail address	Specifies the e-mail address of the bookable resource. E-mail addresses are not created for bookable resources by default, they must be assigned when the resource is created. 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. The address will become active when the resource is created
Approver	Specifies the user of the resource approver. This attribute acts as a pointer to an Oracle Beehive user. Once specified, the approver can manage the bookable resource's event invites. Note: Multiple approvers can exist for a single bookable resource.	Yes	A valid Oracle Beehive user
Capacity	Specifies 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.	No	A positive integer
Bookable 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 bookable resource. Note: If a type is not chosen, Other will be used.	Yes	Room Equipment Other

Creating Oracle Beehive Bookable Resource Accounts

Oracle Beehive bookable resource accounts can be created using the Oracle Beehive `beectl` command-line utility.

Creating Oracle Beehive Bookable Resource Accounts Using `beectl`

This section describes how to create an Oracle Beehive bookable resource using the `beectl` command-line tool. The instructions in this section assume that the `beectl` shell is being used.

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

```
beectl>
```

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*.

To create an Oracle Beehive bookable resource using the `beectl` command-line tool, use the `add_resource` command. Mandatory options when creating a bookable resource are: `--name`, `--resource_type`, and `--email_address`. For a complete list of options, use the `add_resource` command with the `--help` option.

Example

In the following example, we are creating a bookable resource with the name "Conference Room 1021" and e-mail address of "room1021@domain.com". Additionally, the resource has a capacity of 10, and has a bookable resource type of ROOM:

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

Resource is successfully created.

Listing Oracle Beehive Bookable Resource Accounts

Oracle Beehive bookable resource accounts can be listed using the Oracle Beehive `beectl` command-line utility.

Listing Oracle Beehive Bookable Resource Accounts Using `beectl`

This section describes how to list Oracle Beehive bookable resources using the `beectl` command-line tool. The instructions in this section assume that the `beectl` shell is being used.

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

```
beectl>
```

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*.

To list Oracle Beehive bookable resources using the `beectl` command-line tool, use the `list_resources` command. There are no mandatory options for the `list_resources` command. For a complete list of available options, use the `list_resources` command with the `--help` option.

Example

In the example that follows we are listing 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 Bookable Resource Accounts

Oracle Beehive bookable resource accounts can be modified using the Oracle Beehive `beectl` command-line utility.

Modifying Oracle Beehive Bookable Resource Accounts Using `beectl`

This section describes how to modify an Oracle Beehive bookable resource using the `beectl` command-line tool. The instructions in this section assume that the `beectl` shell is being used.

To use the `beectl` shell, run the `beectl` command-line tool 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>
```

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*.

To modify an Oracle Beehive bookable resource using the `beectl` command-line tool, use the `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 a complete list of options, use the `add_resource` command with the `--help` option.

Example

In the following example, we are modifying the name of the existing Conference Room 1021 resource and its capacity:

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

```
Resource is successfully modified.
```

Deleting Oracle Beehive Bookable Resource Accounts

Oracle Beehive bookable resource accounts can be deleted using the Oracle Beehive `beectl` command-line utility.

Deleting Oracle Beehive Bookable Resource Accounts Using `beectl`

This section describes how to delete an Oracle Beehive bookable resource using the `beectl` command-line tool. The instructions in this section assume that the `beectl` shell is being used.

To use the `beectl` shell, run the `beectl` command-line tool 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>
```

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*.

To delete an Oracle Beehive bookable resource using the `beectl` command-line tool, use the `delete_resources` command. Mandatory options when deleting a resource are: `--resource` or `--select_by_name`. For a complete list of options, use the `delete_resources` command with the `--help` option.

Example

In the following example, we are deleting a bookable resource with the name `Company Vehicle 2`:

```
beectl> delete_resources --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 can open the resource's calendar to accept or decline invitation.

Oracle Beehive resource approvers can be assigned and removed using 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 Bookable Resource Accounts Using beectl](#)" for more information about listing resources.
2. Determine the user you want to assign as the resource approver. For information about listing users using the `list_users` command, see "list_users" in Module 2 of the *Oracle Beehive Administrator's Reference Guide*.
3. Assign the approver by executing 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, and `<user identifier>` represents the user identifier determined in Step 2.

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 Bookable Resource Accounts Using beectl](#)" for more information about listing resources.
2. Determine the user you want to remove as the resource approver. For information about listing users using the `list_users` command, see "list_users" in Module 2 of the *Oracle Beehive Administrator's Reference Guide*.

3. Remove the 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, and *<user>* represents the user determined in Step 2.

Oracle Beehive Bookable Resource Booking Characteristics

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

Booking Characteristics Options

Depending on how the bookable resource will be used, and by what users, 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.

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_resource` commands.

When adding a resource:

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

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

When modifying a resource:

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

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.

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 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 Workflow 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 12, "Managing Oracle Beehive Events, Policies, and Workflows."](#)

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

See Also: ["Managing Audit Policies"](#) on page 12-16.

Related `beectl` Commands

The following `beectl` commands are available for you to use for managing Auditing.

- `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 12, "Managing Oracle Beehive Events, Policies, and Workflows."](#)

- `list_audit_templates`: Lists audit policy templates and their identifiers
- `add_audit_template`: Adds a new audit template
- `delete_audit_template`: Deletes an audit template
- `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.

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 patchsets for a given client application version
- `delete_client_application_patchset`: Deletes a client application patchset
- `list_client_application_modules`: Lists the modules for a given client application patchset
- `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, workflows, 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 12, "Managing Oracle Beehive Events, Policies, and Workflows."](#)

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 ["Configuring and Managing Oracle Collaboration Coexistence Gateway"](#).
- For more information about setting up and managing an external LDAP-based user directory, see "Integrating and Synchronizing LDAP with Oracle Beehive" in the *Oracle Beehive Installation Guide* for your platform.

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_coexistence_users`: Lists the names and coexistence types of all the users provisioned with a coexistence connector.
- `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_coexistence_user`: Provisions or deprovisions a user to a coexistence connector.
- `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 the Workflow Service

The Workflow Service supports and manages for Oracle Beehive all workflow settings and other aspects that are related to integrations with Business Process Execution Language (BPEL) components.

See Also:

- For instructions on how to manage workflows, see [Chapter 12, "Managing Oracle Beehive Events, Policies, and Workflows."](#)
- For instructions on creating custom workflows, see *Oracle Beehive Application Developer's Guide*.

Related beectl Commands

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

- `list_workflows`: Lists all workflows
- `list_workflow_schemas`: Lists workflow schemas
- `add_workflow_schema`: Deploys workflow and seed workflow information
- `delete_workflow_schema`: Deletes a workflow schema
- `list_workflow_templates`: Lists workflow templates deployed in a container. The list includes workflow template names and identifiers
- `add_workflow_template`: Creates a new workflow template by importing from an XML file
- `export_workflow_template`: Downloads workflow template into an XML file
- `delete_workflow_template`: Deletes a specified workflow template

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: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 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 WebDAV Service supports and manages all content management-related features and settings that the system leverages over the Web-based Distributed Authoring and Versioning (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- 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 CalDAV Service supports and manages all time management-related features and settings that the system leverages over the Calendaring Extensions to WebDAV (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 ["Managing Oracle Beehive Voicemail and Fax"](#)

Related beectl Commands

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

- `list_attendants`: Lists all attendants in the system
- `add_attendant`: Adds an auto attendant using the information from an AAML document
- `delete_attendant`: Deletes an auto attendant by name
- `delete_attendant_prompt`: Deletes a prompt.
- `modify_attendant`: Updates an existing auto attendant with a new AAML file
- `list_attendant_prompts`: List all prompts that have been uploaded for an attendant
- `upload_attendant_prompt`: Upload a sound file to serve as an auto attendant prompt
- `list_attendant_aaml`: Prints the Auto Attendant Markup Language for an auto attendant
- `modify_ip_phone_password_seed`: Resets the seed of the password generator for IP phones. To send commands to an IP phone the requestor must be authenticated. The password is generated based on information in the phone but must be seeded to ensure security

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 Zimbra client.

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 Mobility Services](#)
- [Managing the Records Management Service](#)
- [Managing the Search Service](#)
- [Managing the Subscription and Notification Services](#)
- [Managing the Workspaces Service](#)

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 MobileDM 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 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: "[Managing Records Management](#)" on page 6-25.

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

There are no beectl commands related to managing this service.

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 9-4 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_team_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 "[Configuring and Managing Oracle Collaboration Coexistence Gateway](#)".

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
- `list_coexistence_users`: List the names and coexistence types of all the users provisioned with coexistence systems
- `modify_coexistence_user`: Provisions or deprovisions a user to 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.

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](#)
- [About Workspace Properties and Controls](#)
- [About Workspace Events](#)
- [Managing Personal Workspaces](#)
- [Using Workspace Templates](#)
- [Creating and Managing Team Workspaces](#)
- [Managing Team Workspace Membership](#)
- [Managing Team Workspace Access Control](#)
- [Managing Files](#)
- [Managing Records Management](#)
- [Example Workspace Template Contents](#)

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 a workspace - 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 may contain shared calendars, files and folders, tasks, address books, and other objects.

Team workspaces may be listed in the Workspace Directory, enabling them to be discovered by any user with access to the Workspace Directory. Team workspaces can be in one of two membership modes: Open Membership allows any user to join the team workspace; Invite Only ensures that a user must be added by a workspace administrator (someone with the workspace-coordinator role).

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.
- `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_team_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

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:

- **Display Name**: A plain text name for the team workspace. Display names of workspaces must be globally unique across the enterprise, 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
- **Primary Contact**: The user who should be contacted when workspace actions need to be taken, such as when a user requests to join the workspace. For team workspaces, the primary contact should usually be the person with the workspace-coordinator role
- **Summary**: A URL that points to a document that exists within the workspace and that provides additional details about the workspace. The document to which the Summary property points is private and can only be viewed by workspace members
- **Public Summary**: A URL that points to a document that exists outside of the workspace and that provides additional details about the workspace. The document to which the Public Summary property points is public and can be viewed by all users in an enterprise
- **Categories**: Categories can be assigned to a team workspace just as they can be assigned to other artifacts within Oracle Beehive
- **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
- **Directory listed status**: Whether the team workspace is listed in the public directory (true or false). Personal workspaces are not listed in the directory
- **Participation mode**: Whether users may join a team workspace, or must be invited first (open, or invite only). Personal workspaces do not have the participation mode property
- **Soft Quota**: The soft quota defines a threshold at which a warning is given that quota is being exceeded. This value is set in KB, 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 KB, 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
- **Personal Tags:** metadata objects that can be applied to data entities such as files and folders. Personal tags are for organizational convenience, because they can be used when searching for objects
- **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

Locking

A user who has the workspace-coordinator role can lock a workspace. Once a workspace is locked, the workspace becomes read-only to all other members (even for other administrators). Only the person who locked the workspace (the lock owner) can modify the workspace. The locking user can optionally specify a set of users who can override the lock (and hence be able to modify the workspace or release the lock). This functionality is useful, for instance, when one or more users want to re-organize content in a workspace.

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 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 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 undeleted. 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 workflows, and auditing. When you are creating policies, event subscriptions, or workflows related to Workspaces, 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, Policies, and Workflows](#)".

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	
BOND_DELETED	
BOND_UPDATED	
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	
ANNOUNCEMENT_CREATED	
ANNOUNCEMENT_DELETED	
ANNOUNCEMENT_READ	
ANNOUNCEMENT_UPDATED	
ATTACHMENT_CREATED	
ATTACHMENT_DELETED	
ATTACHMENT_UPDATED	
BOND_CREATED	
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	

Table 6–1 (Cont.) Workspace Related Business Events

Event	Comments
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	
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	
FOLDER_WORKFLOW_CONFIGURATION_ADDED	
FOLDER_WORKFLOW_CONFIGURATION_REMOVED	
FOLDER_WORKFLOW_CONFIGURATION_UPDATED	
LABEL_CLASS_CREATED	
LABEL_CLASS_DELETED	
LABEL_CLASS_UPDATED	
LABEL_INSTANCE_APPLIED	
LABEL_INSTANCE_REMOVED	
LABEL_INSTANCE_UPDATED	
LINK_CREATED	

Table 6–1 (Cont.) Workspace Related Business Events

Event	Comments
LINK_DELETED	
LINK_UPDATED	
ORGANIZATION_CREATED	
ORGANIZATION_DELETED	
ORGANIZATION_SECURITY_CONFIGURATION_ADDED	
ORGANIZATION_SECURITY_CONFIGURATION_REMOVED	
ORGANIZATION_SECURITY_CONFIGURATION_UPDATED	
ORGANIZATION_UPDATED	
WORKSPACE_ARCHIVED	
WORKSPACE_CREATED	
WORKSPACE_HQUOTA_OVERFLOW	
WORKSPACE_MEMBERSHIP_CHANGED	
WORKSPACE_PURGED	
WORKSPACE_SECURITY_CONFIGURATION_ADDED	
WORKSPACE_SECURITY_CONFIGURATION_REMOVED	
WORKSPACE_SECURITY_CONFIGURATION_UPDATED	
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 6-9.

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 12, "Managing Oracle Beehive Events, Policies, and Workflows."](#) 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 specifies the blueprint for a workspace. A template can be used for capturing best practices and for domain-specific customizations. For example, a New Product Launch Workspace Template could specify the blueprint for creating workspaces that are suitable for collaboration among members of teams responsible for launching new products.

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, Workflow 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 Template	wstp=Community of Practice Workspace T 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 is designed for general use in team-based workspaces. This template provides the broadest coverage of collaborative features and options, and it is not specific to any particular type of group or function.

By default, workspaces that are based on the basic team workspace template are listed in the system's public workspace directory, although users must receive invitations to join them.

The basic team workspace template is the default team workspace template: if a team workspace is created without designating a template, this template will be used.

- **Community of Practice Workspace Template**

The Community of Practice workspace template is designed for workspaces where users who share common interests can post topics discussions of interest and share best practices. Workspaces that are based on the Community of Practice workspace template are listed in the system's public workspace directory and any enterprise user can join them, with or without an invitation.

By default, the Community workspace template provides a best practices folder hierarchy for optimized content management.

- **Project Workspace Template**

The Project workspace template is designed for time-constrained or date-defined projects. This template provides the Oracle Beehive features and options that facilitate collaborative, team-based projects such as repeating status meetings and a best practices folder hierarchy for optimized management of project content.

By default, workspaces that are based on the Project workspace template are not listed in the system's public workspace directory and members may join them by invitation only.

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.

- **Member Subgroups:**

One or more subgroups of workspace members can be specified in a team workspace template. These subgroups can be used in the template as targets of privilege and task assignments and as meeting attendees.

- **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, Workflow 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 three ways to create new team workspaces: by using the Platform Web Services; by using a client which supports workspace creation, such as Oracle Beehive

Integration for Outlook (OBIO)); or by using the command-line. When creating with OBIO, the workspace is created in the same enterprise or organization scope as the creator's personal workspace. The default team workspace 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:36C1F8C16EC34206A5021B92DDC97279000000000000</cen>
      </identity>
      <role>
        <cen>0038:6B48:acrd:8B1514BAD5FC427E9AE42FB3A88664D200000000001A</cen>
      </role>
    </participant>
    <participant>
      <identity type='USER'>
        <cen>0038:6B48:user:36C1F8C16EC34206A5021B92DDC97279000000000001B</cen>
      </identity>
      <role>
        <cen>0038:6B48:acrd:8B1514BAD5FC427E9AE42FB3A88664D200000000001A</cen>
      </role>
    </participant>
  </body>
</teamWorkspaceTemplate>
```

In Oracle Beehive Release 1 version 1.3 or later, you can alternatively specify participants and roles using the shorter BODN identifiers:

```
<participant>
  <identity type="USER">
    <bodn>user=example_user1</bodn>
```

```

    </identity>
    <role>
      <body>acrd=test_role1,orgn=example_organization1,enpr=example.com</body>
    </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 6-16.

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
Primary contact:          16C3:57F2:syac:37D536448BC43F3DE040578C211A3EA8000000001CA
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:37D536448BC43F3DE040578C211A3EA8000000001D6
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. The workspace is set to INVITE_ONLY participation mode, and is not listed in the public directory.

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, indicate whether or not it is directory listed, and to modify the quota. You can make many other modifications to a workspace using OBIO, 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_team_workspace` command:

```
beectl> delete_team_workspace --workspace <Workspace identifier>
```

When you delete a team workspace, all artifacts stored in that workspace are also deleted.

Managing Categories

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>
      </attributeTemplate>
    </attributeTemplates>
  </defaultTemplate>
</CategoryDefinition>
```

```
<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 6-22.

[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/bee hive/transportabletemplate'
  xsi:schemaLocation='http://xmlns.oracle.com/bee hive/transportabletemplate
http://xmlns.oracle.com/bee hive/transportabletemplate.xsd'>

  <templateAttributes>
  </templateAttributes>
  <body>
    <participant>
      <identity type='USER'>

<cen>0038:6B48:user:36C1F8C16EC34206A5021B92DDC97279000000000000</cen>
      </identity>
      <role>

<cen>0038:6B48:acrd:8B1514BAD5FC427E9AE42FB3A88664D2000000000001A</cen>
      </role>
    </participant>
    <participant>
      <identity type='USER'>
```

```
<cen>0038:6B48:user:36C1F8C16EC34206A5021B92DDC9727900000000001B</cen>
    </identity>
    <role>

<cen>0038:6B48:acrd:8B1514BAD5FC427E9AE42FB3A88664D2000000000001A</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:

- [Managing Team Workspace Visibility](#)
- [Managing Team Workspace Roles](#)
- [Managing Team Workspace Sensitivities](#)

See Also: For complete instructions on managing access control, see [Chapter 13, "Managing Oracle Beehive Access Control."](#)

Managing Team Workspace Visibility

Team workspaces have an attribute called **participation mode**. It may be set to either `INVITE_ONLY`, or `OPEN`.

By default, users can join a team workspace by invitation only: a workspace membership coordinator can make them workspace members. This is defined as the invite-only membership mode.

In addition, a team workspace may be listed in the Workspace Directory for other users to see (discover) and join.

An artifact (folder, file, calendar, event, task, and so on) can be visible to users outside the workspace by setting the Public sensitivity on the artifact.

All actors who have appropriate permissions to browse the workspace directory may find workspaces of interest and can join (if Open Membership) or request membership (with an automated e-mail message that is sent to the workspace's primary contact).

Note: The fact that a team workspace is listed in the Workspace Directory is unrelated to whether or not any content in the workspace is granted the Public sensitivity. See ["Managing Team Workspace Sensitivities"](#) on page 6-22 for details.

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, WORKFLOW_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, WORKFLOW_MGR	discover, read, write, execute, delete
workspace-viewer	none	discover, read
workspace-member	none	discover, read, write, execute, delete

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

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 filesystem access to the specified server path. SQL requires a local filesystem 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.

Oracle Beehive manages records *in place*. In-place means that Oracle Beehive retains artifacts for which you have filed records within the Oracle Beehive database, but treats them specially.

When an artifact becomes records managed in-place, Oracle Beehive ensures that the content is never truly altered or deleted from the system until a URM action is issued. From an end-user perspective, users are able to 'delete' e-mails or documents that are records managed, and empty them from the Workspace Trash.

Those records managed artifacts are then stored in a special Records Management container in the Oracle Beehive database. The URM application can still query for and perform operations on such stored artifacts.

Artifacts can be placed under *retention*, which means that they can be treated as regular artifacts by Oracle Beehive.

Artifacts can also be "records managed" without being placed under retention. These types of records are called *non-records* (as a short form of "non-retained records"). Artifacts with non-records can be treated as regular artifacts (including being modified or deleted) by Oracle Beehive. URM will send an instruction to Oracle Beehive when it is time to handle non-record artifacts if they still exist in the system.

In Oracle Beehive Release 1, you can create records of:

- Any document stored in a workspace in Oracle Beehive
- Any e-mail in an Oracle Beehive e-mail Inbox or subfolder of an Inbox
- Any e-mail sent from an Oracle Beehive e-mail user, if you turn on this feature by setting a property of the E-mail Service

As an administrator, you can manually file records for these artifacts using a `beectl` command, or by using Oracle Beekeeper. Additionally, you can create policies to automatically file records for documents and e-mails, based on triggering criteria.

Note: Oracle Beehive's built-in auditing function automatically audits Records Management related events. Such auditing is not Administrator-configurable. You can create an audit trail to review Record management activity. For more information about auditing in Oracle Beehive, see "[Managing Audit Policies](#)" on page 12-16.

Whenever a document is filed as a record, metadata about the artifact is sent to URM. The metadata describes the artifact and its original context. The artifact itself continues to be stored in Oracle Beehive, which ensures that the content is immutable from a system perspective (unless it is a non-record). [Table 6-4](#) lists the document metadata sent to URM with documents. [Table 6-5](#) lists the metadata also sent to URM when e-mail messages are filed as records.

Table 6-4 Artifact Metadata Sent to URM with All Artifacts

Name	Type
CollabID	String
Creator Name	String
Artifact URL ¹	String
Media Type	String

Table 6–4 (Cont.) Artifact Metadata Sent to URM with All Artifacts

Name	Type
Creation Date	Date in "MM/DD/YY" format

¹ Documents only. E-mail messages are not accessible via a URL.

Table 6–5 Artifact Metadata Sent to URM with E-Mail Messages

Name	Type
Sender	String
Sent Date	Date in "MM/DD/YY" format
Subject	String
Hidden Addresses	String, comma delimited list of addresses
Primary Addresses	String, comma delimited list of addresses
Secondary Addresses	String, comma delimited list of addresses

This section contains the following topics:

- [Configuring Oracle Universal Records Manager](#)
- [Filing Records of Artifacts in Oracle Beehive](#)
- [Removing Records Management of Artifacts in Oracle Beehive](#)

Configuring Oracle Universal Records Manager

This section describes the steps required to enable the Records Management Service in Oracle Beehive. Oracle Beehive should already be installed.

Oracle Beehive Records Management requires an existing installation of Oracle Universal Records Management 10g Release 3 (10.1.3) or later. URM itself requires an Apache Web server.

See Also: For detailed instructions on installing and configuring Oracle Universal Records Management, see the *Universal Records Manager Installation Guide*

After installing URM, to configure URM with Oracle Beehive, perform the procedures in each of the following sections, in order:

1. [Registering URM in Oracle Beehive](#)
2. [Creating Retention Categories and Record Folders in URM](#)
3. [Setting Up Disposition Rules in URM](#)
4. [Enabling Record Filing of Sent E-mails](#)

Registering URM in Oracle Beehive

Perform the following steps to register URM in Oracle Beehive:

1. Ensure the Records Management Admin user in URM (typically the `sysadmin` account) is granted all possible roles:
 - a. Log in to URM
 - b. Click **Admin Applets**

- c. Click **User Admin**
- d. Select the Admin user (sysadmin) and click **Edit**
- e. Add the following roles: rma, rmaadmin, admin, ermadmin, ermrequestor, rmaprivileged, sysmanager
- f. Save and exit

Note: The Oracle Beehive Records Management Service uses this account to connect to URM using Web Services. These values must be updated and the Records Management Service restarted should these values change in URM. Use the `beectl modify_property` command to update these values.

- 2. Use the `beectl add_urm` command to update the values of the URM system object in Oracle Beehive:

Caution: If you intend to configure Oracle Beehive to enable filing records of sent e-mail messages, you must provide a user name for the `--rm_email_admin` option when running the `beectl add_urm` command. That user account will be used to store sent e-mail messages. You can use any existing Oracle Beehive account name.

Once you activate the configuration, you cannot easily change this account value.

If you need to change the account that you used when running `add_urm`, contact Oracle support.

```
./beectl add_urm --rm_admin_name name <URM Admin User, like sysadmin> --rm_admin_password <URM Admin Password> --urm_url <URL to connect with URM> --agent_name <Unique Agent Name for communication between Oracle Beehive and URM> [--rm_email_admin user=<user account for storing sent e-mails>]
```

Note: The `add_urm` command shown must be issued from the operating system command line, and not from `beectl` shell mode.

The URL required by the adapter to communicate with the URM server varies depending on certain URM install options (instance name and server name). The URL can be obtained from any URL to the URM server on the Web interface. Discard everything after the question mark (?). For example, if service requests on the Web interface are of the form:

```
http://yourcompany.stellent.com/xpedio/idcplg?IdcService=GET_DOC_PAGE&Action=GetTemplatePage&Page=HOME_PAGE&Auth=Internet
```

You can find the service URL by taking the first part of any service URL string. In this case, it is `http://yourcompany.stellent.com/xpedio/idcplg`.

The agent name can be any meaningful name that indicates that this agent is for the Oracle Beehive Records Management Service, up to 30 characters. For example, "BeeAdapter" or "BeehiveRMAdapter".

3. Enable the records management service (which is disabled by default), by using the `beectl modify_property` command:

```
beectl>modify_property --component <RecordsManagementServiceInstance_id> --name
Status --value ENABLED
```

Note: You can get the component identifier for the Records Management service instance by using the `beectl list_components` command.

4. Optionally, if you want to be able to create records of sent e-mail messages (as opposed to only e-mail messages received by Oracle Beehive users), you must enable that feature by using the `beectl modify_property` command on the Email Service transport properties component:

```
modify_property --component _EmailService:TransportProperties --name
SentEmailPluginEnabled --value true
```

5. Use the `beectl activate_configuration` command to update Oracle Beehive with this proposed configuration:

```
beectl> activate_configuration
```

6. Restart the BEEAPP OC4J managed component using the `beectl restart` command with the `--component` option:

```
beectl> restart --component <your OC4J BEEAPP component identifier>
```

You can use the `beectl status` command to list the managed components and their identifiers for your deployment.

Creating Retention Categories and Record Folders in URM

As the next step, log in to Oracle URM using a Records Management administrator account. Create the various Retention Categories and Record Folders as required by your organization.

To create Retention Categories and Record Folders, perform the following steps:

1. Create retention categories under Record Series or under File plan in the Oracle URM user interface.

To create a retention category under a file plan, click **Create** and select **Retention Category**.

To create a retention category under a record series, click a record series, then click **Create** and select **Retention Category**.

For each retention category, provide values required on the creation page. The retention category ID must be unique, but names can be duplicated. If the **Allow Non-records** check box is checked, the retention category will allow non-records to be checked in.

Note: Non-Records are artifacts that respect Disposition Rules but have no content modification or deletion restrictions.

2. Create Record Folders under a retention category or under another record folder.

To create a record folder under a retention category, click a retention category, click **Create**, and select **Record Folder**.

Record folders can be nested, so you can create a record folder within another record folder. To create a record folder within another record folder, open a record folder, click **Create**, and select **Record Folder**.

For each new record folder, fill in the values required on the creation page. Provide a unique record folder ID.

Once retention categories and record folders are created in Oracle URM, they may be viewed from Oracle Beehive using the `beectl list_file_plan` command:

```
beectl> list_file_plan
```

Setting Up Disposition Rules in URM

Disposition rules are set on Retention Categories only.

In Oracle Beehive Release 1, the Records Management Service supports only the "DESTROY" disposition action.

Perform the following procedure to set up disposition rules for testing:

1. Using the Oracle URM user interface, on the retention category page, select the **Information** drop-down list and click **Disposition Information**
2. On the Disposition Instruction page, click the **Add** link. A **Disposition Rule** panel is displayed
3. On the Disposition Rule panel:
 - a. Select a triggering event. For Oracle Beehive Records Management disposition tests, choose the **Cancel** event, which can be triggered from the Oracle URM administrator UI
 - b. Specify a retention period. For Oracle Beehive Records Management disposition tests, set the retention period to be zero weeks
 - c. Choose Disposition Action to be **Destroy**

Note: In Oracle Beehive Release 1 (1.2), only Destroy commands are supported. Archive, transfer, and accession commands will be available in future versions of Oracle Beehive.

- d. Leave the **Destination Location** and **Destination Container** blank
 - e. Set values for **Apply to Records Folder** and **Notification Reviewer**. By default, the Notification Reviewer is sysadmin
4. After filing a record, you must manually click **Actions**, select **Trigger Dates**, and select **Cancel** on the record from Oracle URM. This will trigger a cancel action and starts disposition processing in Oracle URM.

Enabling Record Filing of Sent E-mails

By default, you can file records of e-mail messages from any Oracle Beehive user's Inbox or subfolder of Inbox. However, you cannot file records of e-mail messages sent by Oracle Beehive users.

You can enable record filing of sent e-mails by modifying a property of the Transport Properties subcomponent of the E-mail Service.

See ["Configuring Sent E-mail Plugins"](#) on page 8-4 for details.

Filing Records of Artifacts in Oracle Beehive

You can manually file records of artifacts (documents and e-mails) using `beectl` commands.

You can file records automatically using policies.

Each of these procedures is detailed in the following sections:

- [Filing Records of Artifacts using `beectl`](#)
- [Filing Records of Artifacts using Policies](#)

Filing Records of Artifacts using `beectl`

The command-line based record filing is provided primarily as a tool for system administrators to manually file records that were not filed using a policy action. You can file a record of an e-mail or document using the `beectl add_record` command:

```
beectl> add_record --artifact <identifier of the artifact to be filed as a record
or non-record> { --retention_category <identifier of the URM retention category> |
--record_folder <identifier of the URM record folder> } [ --no_retention <boolean
specifying a record or non-record> ]
```

You must supply an identifier for the artifact for which you want to file a record.

Specify a Retention Category or Record Folder. You can view a list of record categories and record folders, along with their identifiers, using the `beectl list_file_plan` command:

```
beectl> list_file_plan
```

Filing Records of Artifacts using Policies

You can make use of the Oracle Beehive policy framework to create records management policies. Records management policies automate the process of filing records for artifacts. You specify a policy condition for record filing, and a destination Retention Category. Oracle Beehive automatically files records for artifacts that meet the policy condition.

To set up a records management policy, perform the following steps:

1. Select a Retention Category for this policy. All records that meet this policy's condition will be filed into that Retention Category. You can list available Retention Categories using the `beectl list_file_plan` command:

```
beectl> list_file_plan
```

2. Create an Oracle Beehive policy XML file, using the special Records Management policy action `File a Record`, and setting the `actionPreferenceInfo`, as in the following example:

```
.
.
.
<ActionInfo>
  <name>File a Record</name>
</ActionInfo>
<ActionPreferenceInfos>
  <actionPreferenceInfo>
```

```

        <key>category_id</key>
        <value>Your Category ID</value>
    </actionPreferenceInfo>
</actionPreferenceInfo>
    <key>is_record</key>
    <value>true</value>
</actionPreferenceInfo>
</ActionPreferenceInfos>

```

In this example, you must replace **Your Category ID** with the actual Retention Category ID you determined in step 1

3. Complete the XML file by specifying conditions to trigger the policy. Any condition which is valid for an Oracle Beehive policy may be used.

See Also: For detailed instructions on how to create Policy XML files, see ["Creating and Managing Custom Policies"](#) on page 12-16

4. Create the policy by using the `beectl add_policy` command:

```
beectl> add_policy --file <full path to the policy xml file>
```

[Example 6–6](#) shows an example of a typical Records Management policy. In this example, a record will be filed on any file uploaded to a specified folder, with the Retention Category LC.

Example 6–6 Sample Records Management Policy XML File

```

<?xml version="1.0" encoding="UTF-8" ?>
<PolicyInfo isExtensible="true">
  <policy></policy>
  <scope></scope>
  <template></template>
  <name>Records Management Document Policy</name>
  <description>This policy files all documents uploaded to container identified by
eid 4A92592959297602769797962 under retention category with id LC</description>
  <attributes>
    <attributeDefId></attributeDefId>
    <type></type>
    <value></value>
  </attributes>
  <RuleInfos>
    <RuleInfo priority="1">
      <name>RULE ONE</name>
      <description>Rule One</description>
      <eventTypeName>DOCUMENT_CREATED</eventTypeName>
      <ruleId></ruleId>
      <toRemove>false</toRemove>
      <templateRuleIds>
        <templateRuleId></templateRuleId>
      </templateRuleIds>
      <ConditionInfo>
        <Simple>
          <leftSide>common_attributes.container.eid</leftSide>
          <operator>=</operator>
          <rightSide>'4A92592959297602769797962'</rightSide>
        </Simple>
      </ConditionInfo>
    </RuleInfo>
  </RuleInfos>

```

```

    <ActionInfo>
      <name>File a Record</name>
    </ActionInfo>
    <ActionPreferenceInfos>
      <actionPreferenceInfo>
        <key>category_id</key>
        <value>LC</value>
      </actionPreferenceInfo>
    </ActionPreferenceInfos>
  </RuleInfo>
</RuleInfos>
</PolicyInfo>

```

Removing Records Management of Artifacts in Oracle Beehive

Filing an artifact as a record makes it immutable as long as it is under records management control. This means that no Oracle Beehive command or action will delete the artifact.

Caution: The `beectl delete_record` command cannot be reversed or undone.

You must have the `RECORDS_MGR` privilege in order to release the records management control of an artifact.

To release an existing artifact, use the `beectl delete_record` command:

```
beectl> delete_record --artifact <identifier of the artifact to be deleted as a
record/non-record>
```

Note: The `beectl delete_record` command does not delete the artifact from Oracle Beehive. Instead, it removes the designation of that artifact as a record, allowing it to be handled normally (including being deleted) by other Oracle Beehive operations.

If the artifact is in the Record Store and its records management is removed (released), that artifact will be deleted from the system. This is because only those records managed artifacts that have been deleted and purged by users are placed in the Record Store. Once the record for the artifact is deleted, the previous action of deleting the artifact becomes complete.

Troubleshooting Records Management Service Operations

This section describes actions you can take to help troubleshoot issues with Records Management in Oracle Beehive. If you cannot diagnose your problem using the following steps, contact your Oracle support representative.

This section contains the following topics:

- [Record Filing Failed](#)
- [Removing Records Failed](#)
- [Disposition Not Processed](#)
- [URM Login, Password, or URL Incorrect or Changed](#)

- [Configuring URM for Dispositions Testing](#)

Record Filing Failed

Use the following steps as guidelines for troubleshooting cases where the filing of a record for an artifact failed to occur:

- If a policy-based record filing failed, check the following:
 1. Check to see if the policy was successfully registered, using the `beectl list_policies` command. If your policy is not listed, it has not been created in Oracle Beehive
 2. Check that the policy was created at the correct scope and for the correct operation (event). Events include "DOCUMENT_CREATED", "ES_MSG_SENT" for filing sent e-mail messages as records, and "ES_MSG_DELIVERED" for filing received e-mail messages as records.. Review the policy XML file for the scope and event attributes, ensuring there are no errors
 3. Try creating a policy with the same triggering conditions and scope, but with a general (non record-filing) action. You can test to see if this policy successfully runs when the triggering event occurs

Note: When writing policies for filing records of e-mail messages, you should be aware that a separate e-mail event is raised once for each recipient of an e-mail. The event payload of each event contains the `recipient_eid` of the user getting the e-mail.

If you create a policy which depends on `recipient_eids` with two or more different values, it will never be true, since the policy will only ever evaluate events containing one recipient at a time.

- Ensure that the Records Management Service is running. You can use the `beectl status --all_services` command, and check the status of the Records Management Service. Alternatively, try running the `beectl list_file_plan` command: this command only completes successfully if the Records Management Service is running
- Ensure that your Oracle Universal Records Management Server is running. You can verify this by logging in to URM as `sysadmin`
- Review the log files for the BEEAPP component, to see if the RM Event Action is triggered. Event processing is asynchronous, so there is a delay between the service raising an event (such as `document_created`) and the event service sending the event to the interested service (in this case the Records Management Service). If the log file does not show the RM Event Action, it may be that the event was not dispatched.

The log file is located at `$ORACLE_HOME/bee hive/logs/oc4j/BEEAPP/log.txt`. You may have rotated logs of the format `log.txt.<number>`

- In the Oracle Beehive Database, check the `ECA_FAILED_ACTION_DETAILS` in the `bee_code` schema for any exception messages stored to event processing failures
- Log in to URM as `sysadmin`. Select Browse Content and Search for Records filed under the name of the Adapter that was used when you registered URM with Oracle Beehive. Try to locate the CollabID for the document or e-mail for which you are trying to file a record, in the list of returned records

- The Records Management Service audits all possible actions and errors. You can query the `AUDIT_RECORDS` table with the event names that should have triggered the records management policy, to see if there are any results
- Check the `ocs_logs` table in the `bee_code` schema. Messages are stored here in the event that auditing fails

Removing Records Failed

Typically, deleting a record of an artifact is a synchronous manual operation, so the exception message returned by `beectl` should tell you what went wrong.

Use the following steps as guidelines for troubleshooting cases where the removal of an artifact's record failed to occur:

- Ensure that the Records Management Service is running. You can use the `beectl status --all_services` command, and check the status of the Records Management Service. Alternatively, try running the `beectl list_file_plan` command: this command only completes successfully if the Records Management Service is running
- Ensure that your Oracle Universal Records Management Server is running. You can verify this by logging in to URM as `sysadmin`

Disposition Not Processed

Disposition processing is an asynchronous automatic processing performed by the Records Management Service. By default, Oracle Beehive loads and processes dispositions from Oracle URM once every hour.

Use the following steps as guidelines for troubleshooting cases where a record disposition failed to occur:

- Ensure that the Records Management Service is running. You can use the `beectl status --all_services` command, and check the status of the Records Management Service. Alternatively, try running the `beectl list_file_plan` command: this command only completes successfully if the Records Management Service is running
- Ensure that your Oracle Universal Records Management Server is running. You can verify this by logging in to URM as `sysadmin`
- Review the log files for the BEEAPP component and the `ocs_logs` table in the `bee_code` schema, to see if there are any messages that indicate dispositions have been fetched from URM and loaded in the Oracle Beehive Database.

If there are any dispositions listed, check the `AUDIT_RECORDS` table and `ocs_logs` table for disposition errors. If a disposition action fails, its status will be changed to `ERR`. The `disposition_exception` column indicates the cause of the failure, along with a full stack trace

URM Login, Password, or URL Incorrect or Changed

During the procedure outlined in ["Registering URM in Oracle Beehive"](#) on page 6-27, if the `beectl add_urm` command was specified with incorrect `--rm_admin_name`, `--rm_admin_password`, or `--urm_url`, or if these values change (such as if you change the URM admin password), you can use the `beectl modify_property` command to change the values:

```
beectl> modify_property --component <component id of the URM connector created
with the add_urm command> --name <property name> --value <new value>
```

You can find the component ID by using the `beectl list_components` command:

```
beectl> list_components --type StellentUrm
```

You can then list the properties of the component using the `beectl list_properties` command:

```
beectl> list_properties --component <component identifier>
```

After making changes to component properties, you must run the `beectl activate_configuration` command to validate and activate your proposed configuration:

```
beectl> activate_configuration
```

Configuring URM for Dispositions Testing

Oracle URM processes dispositions in batches. The impact of this is that disposition tasks are not necessarily available at any given time. For convenient testing of dispositions in Oracle Beehive, you can make the following configuration changes:

1. From the Oracle URM administrator UI, click **Administration**, select **Configure Records Management**, choose the **Audit** tab, and select **Checked-in Audit Entries**
2. On the option screen that opens, click the link for **Default Metadata for Checked-in Audit Entries**
3. A check-in screen will open. Enter a value in the required **Title** field for the title of the checked in Audit logs and click the **Submit** button
4. Select a retention category, and the system will list records checked in under this retention category by a given agent
5. Choose a few documents, click **Actions**, select **trigger dates**, and select **Cancel**
6. Wait for one to two minutes for the Oracle URM table to get updated. Then from **Administration**, select **Configure Records Management**, select **Batch Services**, and select **Run All**
7. Click **My Content Server** and select **My Records Assignments**. All dispositions that are due will show up on the list
8. For a given disposition, use the actions icon on the right side of the screen to approve it. Once you do that, the disposition will be a pending completion. You can see the disposition by clicking the **Pending Completion** tab and choosing **My Completed Option** for the external source. At this point the disposition is approved and will show up as a pending disposition for the Oracle Beehive adapter.

Example Workspace Template Contents

[Example 6–7, "Example Workspace Template XML File"](#) shows an example workspace template XML file (in this case, the Community of Practice Workspace template:

Example 6–7 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>2007-08-12</authorCreationTime>
<contactInfo>Oracle Corporation</contactInfo>
<copyrightInfo>Copyright (c) 2007 Oracle Corporation. All rights reserved.</copyrightInfo>
<description>template for creating a workspace that can serve as a place for individuals who share
common interests to post interesting discussions and share best practices</description>
<name>Community of Practice Workspace Template</name>
<templateId>oracle.com.community-of-practice-workspace-template</templateId>
<domain>general</domain>
</templateAttributes>
  <templateVariable>
    <name>default-announcement-body</name>
    <description>starting instructions for workspace members</description>
  </templateVariable>
</body>
  <publicSensitivityTemplateBodyId>public_sensitivity</publicSensitivityTemplateBodyId>
  <defaultSensitivityTemplateBodyId>default_sensitivity</defaultSensitivityTemplateBodyId>
  <attributes>
    <name prompt="true" promptMessage="enter workspace name">workspace name</name>
    <description>enter project description here</description>
    <publiclyListed>T</publiclyListed>
    <participationMode>OPEN</participationMode>
  </attributes>
  <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 workspace ${sys.workspace.name}</subject>
      <messageBody>
        <mediaType>text/plain</mediaType>
        <body>${default-announcement-body}</body>
      </messageBody>
    </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>
</defaultCalendar>
<defaultEmailInbox id="inbox_folder">
    <name>INBOX</name>
    <description>inbox for email messages</description>
</defaultEmailInbox>
<defaultTaskList id="default_task_list">
    <name>Tasks</name>
    <description>workspace tasks</description>
</defaultTaskList>
<entities>
    <folder id="documents_folder">
        <name>Documents</name>
        <description>folder for workspace documents</description>
        <entities>
            <folder>
                <name>Best Practices</name>
                <description>folder for best practice documents</description>
            </folder>
        </entities>
    </folder>
    <folder id="public_documents_folder">
        <name>Public Documents</name>
        <description>folder for workspace documents</description>
        <ace>
            <grantAccessType>READ</grantAccessType>
            <accessor type="GROUP"><systemDefinedGroupName>ALL_
USERS</systemDefinedGroupName></accessor>
        </ace>
    </folder>
    <forum>
        <name>Best Practices</name>
        <description>forum for discussing best practices</description>
    </forum>
</entities>
<summary>
    <inlineSummary>
        <fileName>summary.html</fileName>
        <mediaType>text/html</mediaType>
        <body> <title>&lt;title> Workspace Summary for Members &lt;/title>
            &lt;b> &lt;sys.workspace.name> &lt;/b>
            &lt;p>The Workspace Coordinator should edit this text to share private
information about this workspace with the workspace members. For example, add a little text here
describing workspace goals and member etiquette. &lt;/p>
        </body>
    </inlineSummary>
</summary>
<publicSummary>
    <inlineSummary>
        <fileName>publicSummary.html</fileName>
        <mediaType>text/html</mediaType>
        <body> <title>&lt;title> Workspace Public Summary &lt;/title>
            &lt;b> &lt;sys.workspace.name> &lt;/b>
            &lt;p>The Workspace Coordinator should edit this text to share public
information about this workspace with others in your company. For example, add a little text here
describing the purpose of this workspace. &lt;/p>

```

```
        </body>
      </inlineSummary>
    </publicSummary>
  </body>
</teamWorkspaceTemplate>
```

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](#)
- [Managing the Device Management Service](#)
- [Managing the Mobile Data Sync Service](#)
- [Managing the Mobile Mail Service](#)

Administrative tasks described in this module involve using the `beectl` command-line tool. The instructions assume that the `beectl` shell is being used.

See Also: For more information using the `beectl` shell, see "Oracle Beehive Command-Line Utility" in Module 2 of the *Oracle Beehive Administrator's Reference Guide*.

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. Topics are organized by service, and then further detailed by task.

Before performing the administrative tasks described in this module, Oracle recommends the following reading:

- For an overview of and conceptual information about Oracle Beehive Mobility Services, see "Mobility Services" in *Oracle Beehive Concepts*.
- For information about all Oracle Beehive Services relating to Mobility Services, see "[Managing the Mobility Services](#)" in "[Managing Oracle Beehive Services](#)".
- For reference information about `beectl` commands used in this module, see "Oracle Beehive Command-Line Utility" in the *Oracle Beehive Administrator's Reference Guide*.

Managing the Device Management Service

Oracle Beehive allows you to create new mobile device profiles, configure certain Device Management Service properties, and manage available applications. This section explains how to manage the Device Management Service using `beectl` commands, and contains the following topics:

- [Uploading a Device Profile File](#)
- [Customizing Device Profile Defaults](#)
- [Adding a New Device Type to a Profile](#)
- [Uploading and Provisioning a New Application](#)

Uploading a Device Profile File

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:

1. Save the device profile XML file in a directory accessible by the Oracle user.
2. Execute the following 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 will be made available periodically by Oracle in subsequent patches.

Customizing Device Profile Defaults

Device profile files are located in the `$ORACLE_HOME/bee hive/seed/oma` directory of your Oracle Beehive deployment. Oracle Beehive administrators can customize the default values in these files to accommodate the needs of the users.

To customize the device profile defaults:

1. Open the `$ORACLE_HOME/bee hive/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.

See Also: For a list of configurable attributes in the `<Configuration>` section of the device profile file, refer to [Table 7-1, "<Configuration> Attributes in a Device Profile File"](#).

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

Table 7–1 (Cont.) <Configuration> **Attributes in a Device Profile File**

Preference Set	Attribute	Description	Accepted Values
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
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	conflict_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_attachments	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 the section ["-Uploading a Device Profile File"](#).

To add a new device to a device profile file:

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 the configuration changes applied in Steps 1 and 2:

```
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 the 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. Exit the SyncML message file.
10. Open the \$ORACLE_HOME/bee hive/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 exit the device profile file.
13. Disallow uncertified devices from access 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 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 File"](#).

Uploading and Provisioning a New Application

Occasionally, Oracle Beehive administrators 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:

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/bee hive/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.

Managing 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:

- [Listing Mobile Data Sync Service Configurable Properties](#)
- [Controlling Synchronized Data Types](#)
- [Controlling MD5 Authentication](#)
- [Controlling Synchronization Ranges](#)

Listing Mobile Data Sync Service Configurable Properties

To list the Mobile Data Sync Service configurable properties, the component identifier is required. To obtain the component identifier, use the `list_components` command with the `--type` option. The component type for the Mobile Data Sync Service is `OmaService`.

To list the Mobile Data Sync Service configurable properties:

1. Determine the component identifier by running the following command:

```
beectl> list_components --type OmaService
```

Component Type	Component Identifier
OmaService	_OmaService

- Using the component identifier determined in Step 1, list the configurable properties for the Mobile Data Sync Service by running the following command:

```
beectl> list_properties --component _OmaService
```

The command will return a list of properties in a tabular format, similar to the following:

```
beectl> list_properties --component _OmaService
```

Property name	Property value
Alias	_OmaService
CalendarSyncEnabled	true
ContactsSyncEnabled	true
DbLogKeepDays	7
EmailNumberLimit	1000
EmailSyncEnabled	true
LogSyncmlForUsers	
MaxConcurrentRequests	100
MaxMessageSize	60000
MaxObjectSize	5000000
MaxSyncRangeBack	365
Md5Required	false
Md5Supported	false
NotesSyncEnabled	true
OmaDsEnabled	true
ResumeEnabled	true
ServiceApplication	svccapp_oma-service
ServiceInstances	instance_oma-service-BEEAPP_site1.host.domain.com
SessionLogRequired	true
SessionTimeOut	600
Site	_CURRENT_SITE
Status	ENABLED

```
-----+-----
SyncmlLogRequired | true
-----+-----
```

Controlling Synchronized 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.
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 property changes 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: The `activate_configuration` command must be executed after modifying a property.

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 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 Data Sync Service 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/bee hive/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 File](#).

Controlling Synchronization Ranges

Oracle Beehive administrators 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. Administrators 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 do not affect default values.

To modify the maximum date range allowed for data synchronization:

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: The `activate_configuration` command must be executed after modifying a property.

Managing 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:

- [Listing Mobile Mail Service Configurable Properties](#)
- [Controlling Maximum Number of E-mails Pushed to a Device](#)
- [Controlling the Maximum Message Size](#)
- [Controlling Past E-mail Push](#)

Listing Mobile Mail Service Configurable Properties

To list the Mobile Mail Service configurable properties, the component identifier is required. To obtain the component identifier, use the `list_components` command with the `--type` option. The component type for the Mobile Mail Service is `PushMailService`.

To list the Mobile Mail Service configurable properties:

1. Determine the component identifier by running the following command:

```
beectl> list_components --type PushMailService
-----
| Component Type      | Component Identifier |
-----
| PushMailService     | _PushMailService    |
-----
```

2. Using the component identifier determined in Step 1, list the configurable properties for the Mobile Mail Service by running the following command:

```
beectl> list_properties --component _PushMailService
```

The command will return a list of properties in a tabular format, similar to the following:


```
beectl> list_properties --component _PushMailService
```

Property name	Property value
Alias	_PushMailService
HeartbeatInterval	300
IMAPConnLiveTime	720
MaxConcurrentUsers	2000
MaxInboxMessages	200
MaxMessageSize	50
NumberDaysPast	7
ServiceApplication	svcapp_pushmail-service
ServiceInstances	instance_pushmail-service_BEEAPP_sitel.host.domain.com
Site	_CURRENT_SITE
Status	ENABLED

Controlling Maximum Number of E-mails Pushed to a Device

Oracle Beehive administrators 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:

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: The `activate_configuration` command must be executed after modifying a property.

Controlling the Maximum Message Size

Oracle Beehive administrators 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:

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: The `activate_configuration` command must be executed after modifying a property.

Controlling Past E-mail Push

Oracle Beehive administrators can control the maximum number of days in the past of e-mail to push 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 of e-mail that can be pushed to a device:

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: The `activate_configuration` command must be executed
after modifying a property.

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 E-mail Service properties, see ["Managing the E-mail Service"](#) on page 5-12.

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 methods by which you configure Oracle Beehive E-mail:

- [Configuring Oracle Beehive E-mail Parameters](#)
- [Configuring Oracle Beehive E-mail Rules using Oracle Beekeeper](#)

Configuring Oracle Beehive E-mail Parameters

Most configuration of Oracle Beehive E-mail not related to setting server-side rules 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.

Configuring Oracle Beehive E-mail Rules 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.

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 dispatch rules are 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** section to view the current rules.

See ["Setting Up E-mail Relay Routing"](#) on page 8-11 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 Sent E-mail Plugins](#)
- [Configuring VMS Routing to Relay Messages](#)
- [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 Dispatch Rules](#)
- [Specifying a Local Users Domain](#)

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: ["Filing Records of Artifacts in Oracle Beehive"](#) on page 6-31

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 proposed configuration without closing the configuration window, or **Save and close** to apply the proposed configuration and close the window.

Activate the configuration by clicking **Configuration Control** in the **System** box, and then clicking **Activate**.

Configuring VMS Evaluator to Prevent Open Relay

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, and
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 emails 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.

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

Save & Close Cancel Apply

General Properties **SMTP Properties** Transport Properties IMAP Properties

Endpoint Configuration

☒ Assume single endpoint *i* Port 25 *i*

☐ Enable authentication *i*

VMS Definition

VMS Routing

Add Mapping Rule *i*

Type Endpoint Filter Rule +

1 Host Filter Rule

From Remote Host *.your_company.com +

From Remote Host *.another_trusted_domain.com X

From Remote Host 123.45.0.1-123.45.0.100 X

☐ Not matching

Route to VMS OutboundVMS v

2 Catch-All Rule

All other Emails

Route to VMS InboundVMS v

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** checkbox 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 Save & Close Cancel Apply

General Properties **SMTP Properties** Transport Properties IMAP Properties

Endpoint Configuration

☒ Assume single endpoint Port 25

☐ Enable authentication

VMS Definition

VMS Routing

InboundVMS Advanced

OutboundVMS Advanced

* VMS Name OutboundVMS

☐ 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

Recipient *@* ✗

Add Recipient Pattern

Sender Matcher

Matching Mode ☒ Accept ☐ Refuse

☒ Match local email patterns

Sender *@another_trusted_domain.com ✗

Add Sender Pattern

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 will be delivered.

Figure 8–3 Configuring the Inbound VMS Evaluator

Proposed Configuration Save & Close Cancel Apply

General Properties **SMTP Properties** Transport Properties IMAP Properties

InboundVMS Advanced

* 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

Add Recipient Pattern

Sender Matcher

Matching Mode ☒ Accept ☐ Refuse

☐ Match local email patterns

Sender *@* ✗

Sender ✗

Add Sender Pattern

OutboundVMS Advanced

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, click the **Add Pattern** button to add local e-mail pattern(s), according to your internal e-mail domain(s).

Figure 8-4 Configuring Local E-mail Patterns

The screenshot shows the 'Proposed Configuration' window with the 'General Properties' tab selected. The 'General Settings' section is expanded, showing options to 'Enable or Disable Email Sub-Components'. Both 'Enable SMTP server' and 'Enable email processing' are checked. The 'Global Email Settings' section includes an 'Alias' field set to '_EmailService', a 'Global Domain Name' field set to 'default', and a 'Max Email Size' field set to '50000000' with a unit dropdown set to 'Byte'. Below these are two sections for adding email address patterns. The 'List of valid local e-mail address patterns' section contains two entries: '*@yourcompany.co.uk' and '*@yourcompany.com', both marked with a red 'X' and an 'Add Pattern' button. The 'List of valid non-local e-mail address pattern' section is empty with an 'Add Pattern' button. At the bottom, the 'Persistence Settings' section is also visible.

When you have finished making your configuration changes, click **Apply** to apply the proposed configuration without closing the configuration window, or **Save and close** to apply 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 8-9.

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 endpoint. It does not force other endpoints to require authentication.

Figure 8–5 Configuring OutboundVMS Authentication

The screenshot shows the 'Proposed Configuration Version' window for SMTP Properties. The 'Endpoint Configuration' section is expanded, showing 'Assume single endpoint' and 'Enable authentication' both checked. The 'Port' is set to 25. Below this, the 'VMS Definition' section is expanded, showing 'VMS Routing' and 'InboundVMS' (Advanced). The 'OutboundVMS' section is also expanded (Advanced), showing 'VMS Name' as 'OutboundVMS'. Under 'OutboundVMS', 'Require authentication' is checked. Other options include 'Log all exceptions', 'Suppress received from', 'Suppress response text', 'Use recipient matcher', and 'Use sender matcher'.

6. When you have finished making your configuration changes, click **Apply** to apply the proposed configuration without closing the configuration window, or **Save and close** to apply 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*

The screenshot shows the 'Proposed Configuration' window for SMTP Properties. The 'SMTP Properties' tab is selected. Under 'InboundVMS', the 'VMS Name' is 'InboundVMS'. The 'Matcher Configuration' section shows 'Use recipient matcher' and 'Use sender matcher' both checked. The 'Recipient Matcher' section has 'Matching Mode' set to 'Accept' and 'Match local email patterns' checked. The 'Sender Matcher' section also has 'Matching Mode' set to 'Accept' and 'Match local email patterns' checked. Below this, there are two 'Sender' entries: '*@trusted_domain1.com' and '*@trusted_domain2.com', both marked with a red 'X' indicating they are not valid patterns. The 'OutboundVMS' section is collapsed. At the bottom, there is an 'Add a Note' text area.

You can add or remove as many lines as you like.

When you have finished making your configuration changes, click **Apply** to apply the proposed configuration without closing the configuration window, or **Save and close** to apply 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 you specify more than one server, the first server will be used and additional servers will serve as fallback servers.

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, and leave the **Use simple relay mode** checkbox checked
4. If you want to specify multiple servers, deselect the **Use simple relay mode** checkbox, 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

The screenshot displays the 'Proposed Configuration' window for E-mail Relay Routing. The 'Transport Properties' tab is selected, showing various settings for SMTP, IMAP, and General Properties. The 'General Settings' section includes 'Max Retry Period' (120 h) and 'Retry Interval' (30 min). The 'Relay Server' field is set to 'YOUR_RELAY_HOST_HERE'. The 'Use simple relay mode' checkbox is checked. The 'Delivery Routing' section is expanded, showing a list of relay delivery rules. The first rule, '1 Relay Delivery', has a recipient pattern of '*@another_local_domain.com' and a relay host of 'internal_relay.host.domain' with a priority of 25. The second rule, '2 Relay Delivery', has a recipient pattern of '*@*' and three relay hosts: 'gateway1.your_company.cc' (priority 25), 'gateway2.your_company.cc' (priority 25), and 'gateway3.your_company.cc' (priority 25). The 'Dead Letter Notification' section is also visible at the bottom.

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 proposed configuration without closing the configuration window, or **Save and close** to apply 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 forwarded to one or more special archiving addresses. This is a convenient method of capturing all e-mail messages, so that you can then use any e-mail client to access and archive them 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.

7. When you have finished making your configuration changes, click **Apply** to apply the proposed configuration without closing the configuration window, or **Save and close** to apply 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 a virus engine for use in scanning e-mail messages.

Note: This procedure describes how to add a virus scanning engine. However, to enable virus scanning, you must also follow the instructions in ["Managing Attachment Blocking and Virus Scanning"](#) on page 8-15.

At any time, you can validate virus scan engine connectivity, using the `beectl validate_virus_scan_engine_connectivity` command:

```
beectl> validate_virus_scan_engine_connectivity [ --hostname <scanengine_hostname> ] [ --port <scanengine_port> ]
```

To add a virus scanning engine to Oracle Beehive, perform the following steps:

1. Install and configure a virus scanning application. In Oracle Beehive Release 1, only Symantec Scan Engine version 5.1.2 or later is supported
2. Update Oracle Beehive with the virus scanner's host name and port, using the `beectl add_virus_scan_engine` command:

```
beectl> add_virus_scan_engine --hostname <scanengine_hostname> --port <scanengine_port> --validate_connection <true/false>
```

Note: You can use this command to add multiple virus scan engines. Issue the command once for each virus scan engine you want to add.

3. Specify a virus scan policy for Oracle Beehive, by using the `beectl modify_virus_scan_policy` command:

```
beectl> modify_virus_scan_policy --scanpolicy <virusscanpolicy>
```

The following are valid values for `<virusscanpolicy>`:

- NO_SCAN_OR_REPAIR
- SCAN_ONLY
- SCAN_AND_REPAIR

When the `<virusscanpolicy>` is set to `NO_SCAN_OR_REPAIR`, virus scanning is disabled.

4. Validate and activate your proposed configuration using the `beectl activate_configuration` command:

```
beectl> activate_configuration
```

Managing Attachment Blocking and Virus Scanning

After you have added a virus scanning engine, you can enable virus scanning using Oracle Beekeeper.

You can also enable attachment blocking using Oracle Beekeeper. Attachments can be blocked based on the filename's extension, such as ".zip" or ".exe".

You can enable attachment blocking only, enable virus scanning only, or enable both virus scanning and attachment blocking.

If you have enabled virus scanning, attachment blocking, or both, you can customize the notification sent to the addressee (intended recipient).

Caution: Before enabling virus scanning, be sure that you have followed the instructions in ["Adding a Virus Engine to Oracle Beehive"](#) on page 8-14. If you enable virus scanning but do not have a virus scan engine configured, Oracle Beekeeper may prevent delivery of e-mail messages, while it stores them in a queue and waits for a virus scanning engine to respond.

This behavior is intended to prevent delivery of unscanned messages in the event an external virus scan engine becomes nonresponsive.

To enable virus scanning:

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 **Virus Scanning** section, and select the **Activate virus scanning** checkbox
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
5. When you have finished making your configuration changes, click **Apply** to apply the proposed configuration without closing the configuration window, or **Save and close** to apply the proposed configuration and close the window.
6. Activate the configuration by clicking **Configuration Control** in the **System** box, and then clicking **Activate**.

To enable attachment blocking:

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 **Attachment Blocking** section, and select the **Activate attachment blocking** checkbox
4. You can enable or disable attachment blocking depending on the origin of the message. Click the plus button next to the Origin field to add additional origin filters.

You can specify one or more attachment types to block. Click the plus button next to the Extension field to add additional file extension types to filter.

You can configure whether to send a notification to local senders and external senders of a blocked attachment, and you can customize the notification message that will be sent.

5. When you have finished making your configuration changes, click **Apply** to apply the proposed configuration without closing the configuration window, or **Save and close** to apply the proposed configuration and close the window.
6. Activate the configuration by clicking **Configuration Control** in the **System** box, and then clicking **Activate**.

To customize the notification to addressees (intended recipients) of messages with blocked attachments or viruses:

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 **Virus & Bad Attachment Notification** section
4. Enter a Subject and Notification body for the notification
5. When you have finished making your configuration changes, click **Apply** to apply the proposed configuration without closing the configuration window, or **Save and close** to apply the proposed configuration and close the window.
6. Activate the configuration by clicking **Configuration Control** in the **System** box, and then clicking **Activate**.

Reviewing and Deleting Virus Scan Results

Infection details (including the virus name, virus ID, and so forth) for entities that have been scanned by the Oracle Beehive virus scanner and found to be infected are stored along with the Entity Identifier (Entity ID, Entity Type, CollabID), the date of scan, status of whether the virus was removed or not, number of attempts made to repair the entity, and component within a container (if the entity was a container or multi-part mime message) that was infected.

You can use the `beectl list_virus_scan_results` command to review the results of virus scans:

```
beectl> list_virus_scan_results [ [ --scandate <scandate> ] | [ --scandate_from  
<scandate_from> ] [ --scandate_to <scandate_to> ] ] [ --virus_name <virus_name> ]  
[ --virus_id <virus_id> ] [ --entity_type <entity_type> ] [ --obsolete_only  
<true/false> ] [ [ --maximum_results <max_results> ] [ --display_columns <display_  
columns> ] | [ --count_only <true/false> ] ]
```

The following options allow you to query for specific sets of results:

- **--display_columns**

Use this option to limit what information should be displayed about each result. Enter a comma-delimited list of values. Valid choices are `virus_id`, `virus_name`, `component_name`, `scan_date`, `repair_attempts`, `entity_id`, `entity_type`, `repaired`, `aux_data`, and `collab_id`.

- **--scandate**, **--scandate_from**, and **--scandate_to**

With these options, you can specify a date/time (`--scandate`) or a range of dates for the results set. Only the results of scans conducted on the specified dates will

be shown. Specify date/times using the format YYYY-MM-DDTHH:MM:SS. For example, 2008-07-10T23:59:59 specifies one second before midnight on July 10, 2008

- --virus_name, --virus_id

This option will return only results for the exact specified virus name or ID

- --entity_type

This option will return only results for the given type of entity scanned. For e-mail messages, the entity type is `emsg`

- --obsolete_only

This option will return only results where the entity scanned (for example, an e-mail message) has been removed from the system (deleted)

- --maximum_results

Use this option to specify a maximum number of results to be returned by the query

- --count_only

Use this option to return only a count of the results that match

You can delete stored results. You should periodically delete results to avoid consuming an inordinate amount of space in the Oracle Beehive data store. To delete virus scan results, use the `beectl delete_virus_scan_results` command:

```
beectl> delete_virus_scan_results [ [ --scandate <scandate> ] | [ --scandate_from
<scandate_from> ] [ --scandate_to <scandate_to> ] ] [ --virus_name <virus_name> ]
[ --virus_id <virus_id> ] [ --entity_type <entity_type> ] [ --obsolete_only
<true/false> ]
```

The following options allow you to specify which results should be deleted:

- --scandate, --scandate_from, and --scandate_to

With these options, you can specify a date/time (`--scandate`) or a range of dates for the results to be deleted. Only the results of scans conducted on the specified dates will be deleted. Specify date/times using the format YYYY-MM-DDTHH:MM:SS. For example, 2008-07-10T23:59:59 specifies one second before midnight on July 10, 2008

- --virus_name, --virus_id

This option will delete only results for the exact specified virus name or ID

- --entity_type

This option will delete only results for the given type of entity scanned. For e-mail messages, the entity type is `emsg`

- --obsolete_only

This option will delete only results where the entity scanned (for example, an e-mail message) has been removed from the system (deleted)

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

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 proposed configuration without closing the configuration window, or **Save and close** to apply 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`.

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 8-5 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** checkbox 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** checkbox 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 proposed configuration without closing the configuration window, or **Save and close** to apply 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** checkbox, as shown in [Figure 8-8](#)

Figure 8-8 Selecting Accept Invalid Local Users Checkbox

The screenshot shows the 'Proposed Configuration' window for Oracle Beehive. The 'Transport Properties' tab is selected. Under the 'General Settings' section, the 'Accept invalid local users' checkbox is checked and highlighted with a red box. Other settings visible include 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 (checked), Local Delivery Batch Size (100), SMTP Delivery Batch Size (100), and SMTP Check Point (5). Buttons for 'Save & Close', 'Cancel', and 'Apply' are at the top right.

5. When you have finished making your configuration changes, click **Apply** to apply the proposed configuration without closing the configuration window, or **Save and close** to apply 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 8-11.

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 the proposed configuration without closing the configuration window, or **Save and close** to apply the proposed configuration and close the window.
11. Activate the configuration by clicking **Configuration Control** in the **System** box, and then clicking **Activate**.

Specifying a Local Users Domain

The Local E-mail Patterns property is a list of domains, subdomains, or other e-mail address patterns. Users which 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 proposed configuration without closing the configuration window, or **Save and close** to apply 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** checkbox in the Recipient Matcher or Sender Matcher, to include the list of local e-mail patterns in the rule.

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** checkbox, as shown in [Figure 8-8, "Selecting Accept Invalid Local Users Checkbox"](#) on page 8-19. 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 proposed configuration without closing the configuration window, or **Save and close** to apply 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

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 Subscriptions](#)
- [Configuring Notifications to use SMS](#)

See Also: For information on configuring the Notification Delivery Service, see "[Managing the Notification Delivery Service](#)" on page 5-19

Configuring User Subscriptions

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

Although users cannot set up new subscriptions, they can disable any of the default subscriptions provided to them, and they can control where the notifications are delivered.

Users make changes to their subscription preferences using the Oracle Beehive user preference pages, which can be accessed at `http(s)://<beehive host>:<port>/uss`.

Clicking on the **Notifications** link shows the My Notifications page, shown in [Figure 9-1](#).

Figure 9–1 My Notifications Preferences Page

[Out of Office](#) | [Server Side Rules](#) | **Notifications** | [Mobile Devices](#)

My Notifications

This page lets you configure how you want to receive your notifications. When enabled, you will receive the notifications via the specified channels. You can specify your preference by clicking on the "Edit" link.

Notify me when...	Where	Enabled	Action
I am invited to a meeting	Default		Edit
my meeting invitation is withdrawn	Default		Edit
I am assigned a new task	Default		Edit
my task assignment is withdrawn	Default		Edit

You can configure where your notifications will be delivered by default if no delivery channel is specified.

- ☒ Inbox
☐ SMS
☐ Instant Message

Delivery Control

You can control the notification delivery for each channel. When a channel is turned off, no notification will be delivered for that channel.

- ☒ Turn on notification delivery
☒ Inbox
☐ SMS
☐ Instant Message

From the My Notifications page, users can click an **edit** link to enable or disable one of the default subscriptions, or to choose a particular delivery channel they would like the notification sent to. They can also specify the default delivery channel. By default, notifications are sent to the default delivery channel which in turn by default is 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 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_DOCUMENT NOTIFY_ON_ANY_MODIFIED_DOCUMENT NOTIFY_ON_ANY_DELETED_DOCUMENT
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
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

Table 9–1 (Cont.) User Subscription Entities and Rules

Entity	Rule
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
Workflow	NOTIFY_ON_ANY_WORKFLOW_ADDED NOTIFY_ON_ANY_WORKFLOW_COMPLETED NOTIFY_ON_ANY_WORKFLOW_DELETED
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>
```

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.

You can configure SMS notifications in one of two 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 SMPP (Short Message Peer-to-Peer). It will not be able to deliver SMS notifications until a subscription from an SMPP-compliant aggregator is secured. The Oracle Beehive SMS delivery channel has been tested with the following aggregators:

- Verisign Intelligent Messaging Network (IMN)
- Clickatell Gateway

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
```

Managing Oracle Beehive Time Management

This module includes information about various administration tasks relating to Oracle Beehive Time Management services, and contains the following topics:

- [Managing Holidays](#)
- [Modifying User Time Management Preferences](#)
- [Oracle Beehive Time Zone Definitions](#)

Note: All `beectl` commands specified in this module are stated as though the shell mode has been invoked. For more information about `beectl` shell mode, and how to invoke it refer to the "Oracle Beehive Command-Line Utility" module in the *Oracle Beehive Administrator's Reference Guide*.

Managing Holidays

Use the `beectl import_icalendar` command-line tool 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 the *Internet Calendaring and Scheduling Core Object Specification (iCalendar)* document on the Official Internet Protocol Standards Web site:

<http://www.rfc-editor.org/rfcxx00.html>

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-line tool 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 following command from the `$ORACLE_HOME/bee hive/bin` directory:

```
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:A6E2F29FEB7A49DB9F27C0C3E3226413000000000002, iCalendar
UID=2e49f2d5-fcd5-4d19-b46b-5ece651a8f46@example.com.

Imported invitation series, unique

identifier=3449:5915:ocrs:A6E2F29FEB7A49DB9F27C0C3E3226413000000000006, 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 workspace 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 full list of available time zones and their associated identifiers.

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 following command from the `$ORACLE_HOME/bee hive/bin` directory:

```
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: The `--user` option can be specified 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 defined as `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, an end at 6:00 PM -- in the user's defined time zone.

To change a user's defined working hours:

1. Run this command from the `$ORACLE_HOME/bee hive/bin` directory to list the user's preference properties:

```
beectl> list_preference_profiles --consumer <userid>
```

Where `<userid>` represents the user 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 string of characters, within and including the `<WeekBusinessHours>` tags. The string will resemble the following text:

```
<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. Copy the text modified in Step 3.
5. Run the following command from the `$ORACLE_HOME/bee hive/bin` directory to change the working hours for the user:

```
beectl> add_preference_property --set prfs=TimeManagement,<userID> --name
working_hours --type BUSINESS_HOURS --value <WeekBusinessHours>
```

Where `<userID>` represents the user identifier of the user, and `<WeekBusinessHours>` represents the text copied 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 --value "<WeekBusinessHours
xmlns="http://xmlns.oracle.com/2006/Beehive/BOM/business-hours"><<WeekShift
StartDay="MONDAY" StartTime="08:00:00" EndDay="MONDAY" EndTime="18:00:00"
Type="REGULAR"/><WeekShift StartDay="TUESDAY" StartTime="08:00:00"
EndDay="TUESDAY" EndTime="18:00:00" Type="REGULAR"/><WeekShift
StartDay="WEDNESDAY" StartTime="10:00:00" EndDay="WEDNESDAY" EndTime="19:00:00"
Type="REGULAR"/><WeekShift StartDay="THURSDAY" StartTime="10:00:00"
```

```
EndDay="THURSDAY" EndTime="19:00:00" Type="REGULAR"/><WeekShift StartDay="FRIDAY"  
StartTime="08:00:00" EndDay="FRIDAY"  
EndTime="14:00:00"Type="REGULAR"/></WeekBusinessHours>"
```

In [Example 10-5](#), the user with user identifier `user=jsmith` had their working hours modified to 8:00 AM to 6:00 PM on Monday and Tuesday; 10:00 AM to 7:00 PM on Wednesday and Thursday; and 8:00 AM to 2:00 PM on Friday.

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](#)
- [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>

Obtaining a New Oracle Beehive Time Zone Package

The most recent Oracle Beehive time zone package is always 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.

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-line tool.

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 following command from the `$ORACLE_HOME/bee hive/bin` directory:

```
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.

Managing Oracle Beehive Voicemail and Fax

Note: Oracle Beehive Voicemail functionality requires advanced configuration to enable. Some necessary Cisco configuration is not fully documented in this Guide. Please contact your Oracle support representative for assistance with deploying Oracle Beehive Voicemail.

Oracle Beehive Voicemail is provided by the Voice Message Service. It enables a variety of functionality for Oracle Beehive users, including the ability to access and manage voice messages from a telephone or as audio files in the e-mail Inbox. Oracle Beehive Fax is provided by the Fax Message Service. This module describes how to set up and configure the necessary software components to enable the voicemail functionality. It contains the following sections:

- [Introduction to Managing Oracle Beehive Voicemail](#)
- [Configuring Oracle Beehive Voice Message Service](#)
- [Configuring Oracle Beehive Fax](#)

See Also:

- For more information on configuring the Voice Message Service, see "[Managing the Voice Message Service](#)" on page 5-16
- For more information on configuring the Fax Service, see "[Managing the Fax Message Service](#)" on page 5-12

Introduction to Managing Oracle Beehive Voicemail

This section includes the following topics:

- [About Facilities](#)
- [About Auto Attendants](#)
- [About Voicemail Infrastructure](#)

About Facilities

A Facility is an Oracle Beehive group, defined for a physical location that is connected to a single PBX. This concept allows all users at that facility to have common attributes and be treated in a common way. For example, a facility will set the default language for the voicemail users of that facility. Users may also have their own unique parameters as required.

Facilities are optional. You can set preferences and settings at the enterprise level, and those settings not specified at the facility level will default to the enterprise values. Facilities allow you the flexibility of having different settings at different physical locations, each with their own PBX.

About Auto Attendants

An Auto Attendant is an automated call-answering and routing server, which presents a collection of menus that are created for individual phone numbers supported at a facility. The menus allow callers to be routed to the correct department or extension, or expose additional recorded information (such as street directions or operation hours), and the corporate directory.

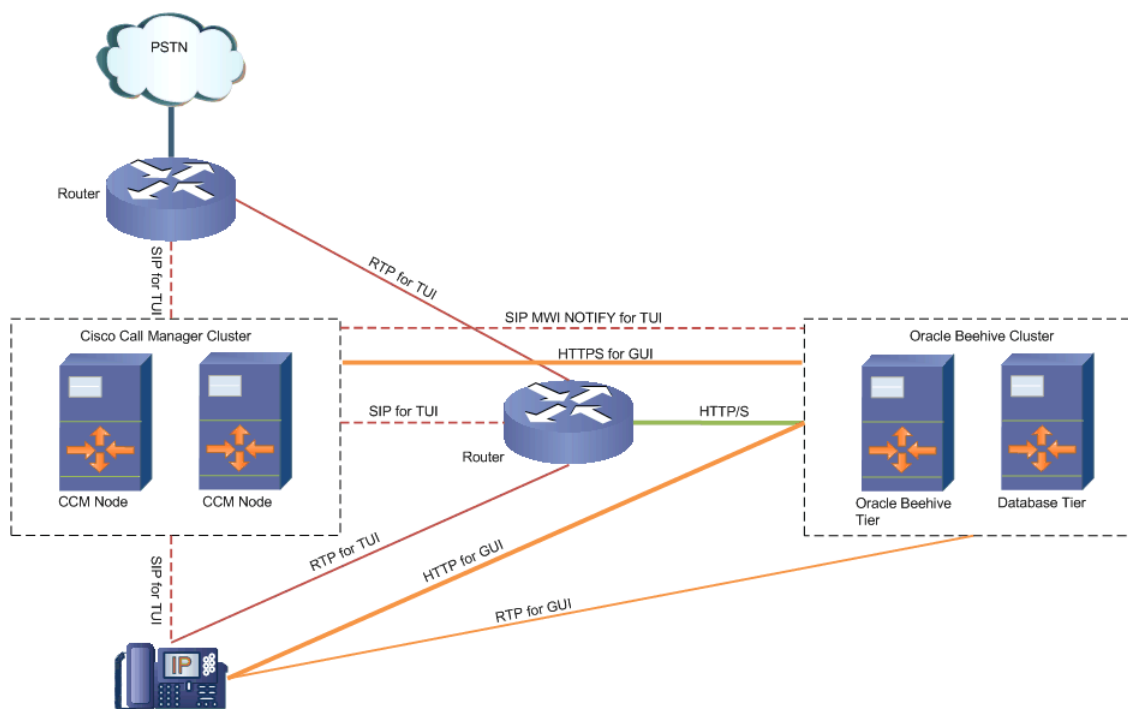
Auto Attendants are optional. You can deploy Oracle Beehive voicemail functionality with or without the use of Auto Attendants. You can have one Auto Attendant for each facility, or one for the whole enterprise.

About Voicemail Infrastructure

Oracle Beehive voicemail and automated attendant (AA) telephone user interface (TUI) use VoiceXML to present the TUI to the caller via a Cisco VoiceXML capable router. The Cisco router has a VoiceXML browser imbedded in the Cisco IOS operating system. VoiceXML is a W3C standards-based approach for voice applications and services, which leverages a Web-based development and deployment model instead of a proprietary telephony hardware and software approach. Since VoiceXML uses a Web based model, all the Oracle Beehive voice applications are executed on the server and only the VoiceXML and audio .wav files are served to the Cisco VoiceXML browser via HTTP or HTTPS.

Figure 11–1 shows an example deployment architecture linking Cisco Call Manager (CCM) hardware to Oracle Beehive.

Figure 11–1 Oracle Beehive Voicemail Centralized Deployment



This section contains the following topics:

- [Cisco Dependencies and Requirements](#)
- [Voicemail UDS Requirements](#)
- [Voicemail Preference Properties](#)

Cisco Dependencies and Requirements

In order to deploy Oracle Beehive voicemail, Cisco VoiceXML-capable hardware and Cisco IOS VXML software is required. The supported Cisco VXML routers are the 2800 and 3800 Series Internet Service Routers (ISR), and the AS5350XM/AS5400XM Universal gateways. These routers offer a VoiceXML featureset in the Cisco IOS operating system, to execute the Oracle Beehive voice applications.

Please refer to your Cisco account representative to determine the VXML browser software licensing required for your environment and hardware. Cisco Unified Call Manager 5.0 or greater is the only IP PBX that has been tested with Oracle Beehive voicemail redirection.

Voicemail UDS Requirements

For an Oracle Beehive user to be active for voicemail, the user's UDS record must contain `voice_principal`, `voice_pin`, and `tel:` (telephone scheme) address attributes. The voice principal and telephone scheme address must be numbers containing no special characters or spaces. See "[Managing and Provisioning Oracle Beehive Users](#)" on page 3-1 for complete documentation on configuring the appropriate user record attributes. You can use the `beectl` command-line tool to add and modify preference profiles, users' voice properties, groups, and group properties.

Voicemail Preference Properties

Voicemail configuration properties follow the Oracle Beehive model of property inheritance: Enterprise, Facility, and User. This allows you to group configuration properties. Enterprise properties apply to all voicemail users. Facility properties apply to the users defined in the UDS group that constitutes a voicemail facility. User properties are defined at the user level inside the user's UDS Voicemail Preference Profile. For example, the user property `PreferredLocale` will override the locale defined at the facility.

There are three types of preference properties:

- [Enterprise Preference Properties](#)
- [Facility Preference Properties](#)
- [User Preference Properties](#)

Enterprise Preference Properties Voicemail stores application configuration options in Oracle Beehive Preference Property Profiles. The enterprise properties for voicemail are stored in a Preference Profile called `VoiceEnterprisePrefs`. Before you can add or modify any voicemail properties, you must create the profile, by using the `beectl add_voice_enterprise` command.

Facility Preference Properties Voicemail has the ability to define multiple facilities in order to decentralize the deployment of voicemail router resources in remote offices without having to run or administer multiple voicemail applications. A voicemail facility is a group of users and properties that share the same physical location and configuration information. You create voicemail facilities by using the `beectl add_voice_facility` command.

User Preference Properties User properties are defined inside the user's voicemail profile. These properties can define the user's PreferredLocale and ActiveGreetingType.

To set a preference property for a user, first use the `beectl list_preference_profiles` command to get the ID of the user's Voice preference set:

```
beectl> list_preference_profiles --consumer user=<userID> --entity_format id
```

Then, use the `beectl add_preference_property` command to set the PreferredLocale or ActiveGreetingType as needed. For example, to set the PreferredLocale to en-us:

```
beectl add_preference_property --set <Voice Preference Set ID> --name PreferredLocale --type string --value en-us
```

Configuring Oracle Beehive Voice Message Service

You must perform some configuration before Oracle Beehive voicemail functionality is enabled. At a minimum, you must configure the enterprise, and you optionally may configure one or more facilities. You must also ensure that all users who will use Oracle Beehive voicemail have required values for the relevant attributes in their user accounts.

If you want to enable the Message Waiting Indicator and graphical user interface features of your telephones, you must configure them. If you want to use an auto attendant to answer and forward calls, you must configure the auto attendant.

This section contains the following topics:

- [Configuring the Enterprise](#)
- [Creating Voicemail Users](#)
- [Managing Facilities](#)
- [Configuring the Voicemail Touch-tone User Interface \(TUI\)](#)
- [Enabling HTTPS for Cisco VXML Enabled Device Access to Oracle Beehive](#)
- [Configuring Cisco IP Phone Voicemail GUI Application](#)
- [Configuring the Voicemail GUI and Message Waiting Indicator](#)
- [Cisco Router Configuration](#)
- [Cisco Unified Call Manager Configuration](#)
- [Configuring the Auto Attendant](#)

Voicemail Properties

Table 11-1, "Voicemail Properties" on page 11-5 lists the properties used by the voicemail Enterprise and voicemail Facilities. When configuring the voicemail Enterprise Preferences, or a facility's properties, refer to this table for a description and examples.

Inheritance Rules

In general, the hierarchy is Enterprise -> Facility -> User, such that property values set at the lowest level are used first, and if no value is set, the property value at the next-higher level is used. This general flow is not followed in some cases.

The following rules are applied for determining inheritance of property values:

- **UNDEFINED VALUES**

When the Enterprise value is not defined, then the default value is used. When the default value is not defined, then the Facility's value, if defined, is used. When neither value is defined, then the application determines a value, which may cause undesirable results.

- **BOOLEAN**

When the Enterprise value is `true`, then the Facility's value, if set, is used. If not set, then `true` is used. If the Enterprise value is `false`, then all Facilities will use the value `false`, *even if specified at the Facility level*.

- **INTEGER**

When the Enterprise value is `-1`, then the Facility's value is used.

- **STRING and STRING ARRAY**

Both of these types behave as UNDEFINED VALUES

Table 11–1 Voicemail Properties

Property Name	Type	Example Value	Default Value	Description
isRetrievalEnabled	BOOLEAN	true	true	Determines whether users are allowed to retrieve their messages
isMWIEnabled	BOOLEAN	true	true	Enables or disables users' Message Waiting Indicator (MWI)
isOperatorConfigured	BOOLEAN	true	true	Determines whether a caller can transfer to a live operator from the Touchtone User Interface (TUI)
MaxRecordingDuration	INTEGER		-1	Specifies the maximum duration of a recording, in seconds. -1 means that the Facility's value will be used. If not set at the Facility level, then 180 is used.
RTPServerURI	STRING	http://<BH_MT>:<PORT>/voice-servlet/rtp_server/RtpServer.jsp		The URI to access the RTP Server

Table 11–1 (Cont.) Voicemail Properties

Property Name	Type	Example Value	Default Value	Description
RTPServerUserName	STRING	RtpClient		This value must be defined, but the user name is only used for internal communication
RTPServerPassword	STRING	RtpPassword		This value must be defined but the password is only used for internal communication
isGlobalLookupFallbackEnabled	BOOLEAN	true	true	
isPhoneNumberPresentable	BOOLEAN	true	true	
StoreMsgMenuLocale	STRING	en-US		The locale to use when messages are being left for a user
RetrieveMsgMenuLocale	STRING	en-US		The locale to use when messages are being retrieved by a caller. This locale is always used at the login, but the user's locale, if specified, will be used after he or she is authenticated
IpPhoneUserName	STRING	bhvmgui		Username of the account defined in Cisco Call Manager which has device control over the users' telephones for audio playback and return call functionality in the Voicemail GUI
IpPhonePassword	STRING	<bhvmgui_password>		Password of the account defined in Cisco Call Manager which has device control over the users telephones for audio playback and return call functionality in Voicemail GUI

Table 11–1 (Cont.) Voicemail Properties

Property Name	Type	Example Value	Default Value	Description
IpPhonePasswordSeed	STRING			This only needs to be set if local IP phone authentication is used. Oracle recommends you use Cisco Call Manager Administrative XML Layer (AXL) configuration for IP authentication
IpPhonePasswordAlgorithm	STRING			This only needs to be set if local IP phone authentication is used. Oracle recommends you use Cisco Call Manager Administrative XML Layer (AXL) configuration for IP authentication
IpPhonePasswordNumBits	INTEGER			This only needs to be set if local IP phone authentication is used. Oracle recommends you use Cisco Call Manager Administrative XML Layer (AXL) configuration for IP authentication
IpPhoneRtpMinPort	INTEGER			
IpPhoneRtpMaxPort	INTEGER			
IpPhoneHttpProxyHost	STRING			This only needs to be set if the Oracle Beehive tier needs to communicate through a proxy to reach the IP phones' HTTP server

Table 11–1 (Cont.) Voicemail Properties

Property Name	Type	Example Value	Default Value	Description
IpPhoneHttpProxyPort	INTEGER			This only needs to be set if the Oracle Beehive tier needs to communicate through a proxy to reach the IP phones' HTTP server
MwiAlgorithm	STRING	SHA1PRNG		Defines the algorithm used to create the SIP NOTIFY messages for MWI which must be set
MwiCcmHost	STRING			The IP address of the Cisco Call Manager SIP trunk configured to accept SIP NOTIFY messages
MwiCcmPort	INTEGER		5060	The port of the Cisco Call Manager SIP trunk configured to accept SIP NOTIFY messages
MwiSipProxyHost	STRING			SIP Proxy Host that is authorized to send SIP messages to Cisco Call Manager SIP MWI trunk
MwiSipProxyPort	INTEGER			SIP Proxy Port that is authorized to send SIP messages to Cisco Call Manager SIP MWI trunk
MwiMinLocalPort	INTEGER	5061		Lowest port in the range that Oracle Beehive will use to send and receive SIP messages
MwiMaxLocalPort	INTEGER	5080		Highest port in the range that Oracle Beehive will use to send and receive SIP messages

Table 11–1 (Cont.) Voicemail Properties

Property Name	Type	Example Value	Default Value	Description
MwiSourcePhone	STRING	16505551234		Voicemail DNIS phone number that is presented in the SIP NOTIFY message
RtpServerHost	STRING	beehive.example.com		Host where the RTP Server is running
RtpServerMinPort	INTEGER		20480	Lowest port in the range used by the RTP server to send RTP packets
RtpServerMaxPort	INTEGER		32768	Highest port in the range used by the RTP server to send RTP packets
RtpServerPacketSize	INTEGER		160	RTP Packet Size. Leave at default for Cisco IP phones
RtpServerHttpProxyHost	STRING			This value must be set if the RtpServer has been deployed on a standalone host which can only be accessed by the Oracle Beehive tier through a proxy
RtpServerHttpProxyPort	INTEGER			This value must be set if the RtpServer has been deployed on a standalone host which can only be accessed by the Oracle Beehive tier through a proxy

Table 11–1 (Cont.) Voicemail Properties

Property Name	Type	Example Value	Default Value	Description
AxlConfigA	STRING	https://<CCM_HOST>:8443/axl/,https://<CCM_HOST>:8443/realtimeservice/services/RisPort,,bhvmgui,password,f,f	none	<p>The comma separated fields, in order, are:</p> <ol style="list-style-type: none"> 1. The AXL URL 2. The realtime info URL 3. The proxy host 4. The proxy port 5. The user provisioned for AXL access (read only) 6. The password for the AXL user 7. Whether certificate validation is enabled for SSL connections to CCM 8. Whether hostname verification is enabled for SSL connections to CCM
AxlConfigB	STRING	https://192.188.175.105:8443/axl/,https://192.188.175.105:8443/realtimeservice/services/RisPort,,bhvmgui,password,f,f STRING_LIST		This field's contents are the same as for AxlConfigA, and are used for backup AXL services running on another Cisco Call Manager Node
RecordStreamURIs	STRING ARRAY		http://<host>:<port>/voice-servlet/vmail/crs	
RecordPlaybackURIs	STRING ARRAY		http://<host>:<port>/voice-servlet/vmail/data/shared/playback	
ExclusiveAudioContentURIs	STRING ARRAY		audio	The default value is a relative audio path

Table 11–1 (Cont.) Voicemail Properties

Property Name	Type	Example Value	Default Value	Description
SharedAudioContentURIs	STRING ARRAY		../shared-audio	The default value is a relative audio path
TelephoneAnsweringAddress	STRING ARRAY	vm-no-reply@example.com	TelephoneAnsweringService@example.com	This is the e-mail from: address that is used for voicemail messages when an Oracle Beehive user is not the originator

Configuring the Enterprise

This section describes how to configure enterprise-level preferences for Oracle Beehive voicemail functionality.

Note: The Enterprise is created during Oracle Beehive installation. The only optional steps are to configure the settings for Voicemail and Auto Attendant. The values are stored in enterprise preferences.

This section contains the following topic:

- [Configuring Enterprise Preferences](#)

Configuring Enterprise Preferences

These are set as part of the Enterprise Preferences:

Use the following commands to set any of the Enterprise Preference properties listed in [Table 11–1, "Voicemail Properties"](#).

1. First, get the identifier for your enterprise by using the `beectl list_enterprises` command:

```
beectl> list_enterprises
```
2. Then, use the enterprise identifier with the `beectl list_preference_profiles` command to get the Active Preference Profile identifier:

```
beectl> list_preference_profiles --consumer <Enterprise identifier>
```

Look for output similar to the following:

```
Name : ActivePreferenceProfile
Identifier : prfp=ActivePreferenceProfile,enpr=Oracle
```

3. Create a preference set using the `beectl add_preference_set` command:

```
beectl> add_preference_set --parent <Preference Profile identifier> --name VoiceEnterprisePrefs
```

The console will output an identifier for the new Preference Set.

4. Use the `beectl add_preference_property` command to set preference properties:

```
beectl> add_preference_property --set <Preference Set identifier> --name <name>
--value <value> --type <type>
```

See [Table 11-1, "Voicemail Properties"](#) for the values of the `--name`, `--value`, and `--type` attributes. Repeat this step for each enterprise property.

Creating Voicemail Users

Generally, you can follow the instructions in ["Managing and Provisioning Oracle Beehive Users"](#) to create users with access to Oracle Beehive voicemail functions. You must provide values for any voicemail user for the following specific user attributes, in addition to the required user attributes:

- `--voice_principal <phone number>`
- `--voice_pin <PIN>`
- `--address <type>:tel:<phone number>`

The `voice_principal` and `tel: address` must be integers containing no special characters or spaces. The `voice_principal` is used for user authentication because it is associated to the `voice_pin`. The `tel: address` attribute is used to associate the redirected DNIS passed by the PBX to the user account or user's e-mail Inbox.

Note: It is possible to define multiple phone numbers that map to the same user account, but only the voice principal defined for that account can be used to authenticate.

Managing Facilities

This section describes how to create and configure a Facility. Facilities allow you to deploy more than one voicemail system (such as, at different physical locations), each with its own properties. Those properties not set at the facility level, will default to their enterprise-level values.

This section contains the following topics:

- [Creating a Facility](#)
- [Sample Facility XML File](#)

Creating a Facility

To create a Facility, you must create both a group and a voice facility object. The group's values (enterprise identifier, name, properties, and so on) are defined in an XML file. The values of these properties are shown in [Table 11-1, "Voicemail Properties"](#). See [Example 11-1, "Sample Facility XML File"](#) on page 11-14 for an example Facility XML file.

Perform the following steps to create a Facility:

1. Create a group by using the `beectl add_group` command:

```
beectl> add_group --file <path to Group XML file>
```

See [Example 11-1, "Sample Facility XML File"](#) on page 11-14 for an example Facility XML file.

2. Add a voice facility object using the `beectl add_voice_facility` command:

```
beectl> beectl> add_voice_facility --group_collabid <1234:grup:1234:5678>
```

```
[--include <phone rules>] [--exclude <phone rules>]]
```

Note: This command initializes the voicemail component configuration properties which maps the voicemail phone numbers assigned to the UDS group (facility). This mapping is contained within a table which is used for phone number look-up, defined with the inclusion and exclusion numbers assigned during the `add_voice_facility` command execution.

Using the optional `--include` and `--exclude` options, you can specify a range of phone numbers to be associated with this voice facility. Use a question mark (?) as a wildcard. Multiple include and exclude ranges can be specified on the command line by delimiting them with a pipe (|) symbol in quotes due to the command shell limitations. For example:

```
beectl> add_voice_facility --name --group_collabid <1234:grup:1234:5678>
--include "1866612????|1866264????|4730"
```

This example associates all phone numbers in the range "18666120000-18666129999 and 18662640000-99999 and 4730" number ranges with this facility. The phone number lookup is based upon ANI, then RDNIS (redirect number), and then DNIS (dialed number), in that order, to make the association.

Note: It is also possible to add full SIP URIs to the facility if needed depending upon the router configuration and deployment. If the router has SIP header parsing enabled, then it may be necessary to use a full SIP URI. For example: `--include "sip:1866612????@192.168.1.1"`

3. Use the `beectl list_groups` command with the global option `--entity_format id` to get the group identifier:

```
beectl> list_groups --group <group name> --show ALL --entity_format id
```

You can look up with which facility or facilities a given phone number is assigned and selected based upon weight (strength of number match) by using the `beectl list_voice_facilities` command:

```
beectl> list_voice_facilities --phone <user or voicemail DNIS number>
```

Sample Facility XML File

[Example 11-1, "Sample Facility XML File"](#) is a sample XML-formatted file for creating the Group when setting up a Facility. In this example, a static group is used, but you could use a more complex group if you want. Be sure to replace the name, description, and scope values with the correct ones for the Facility you are creating.

Note: To find the CollabID of the enterprise or a user, use the global option `--entity_format id` with the appropriate `beectl list` command. For example:

```
beectl list_users --user user=exampleuser --show ALL --entity_
format id
```

The user's CollabID will be shown on the Identifier line of the output.

Example 11–1 Sample Facility XML File

```
<?xml version="1.0" encoding="utf-8"?>
<groups>
  <group type="grup">
    <name>18665552020</name>
    <description>18665552020 Voicemail Facility</description>
    <scope>
      <!-- Define Enterprise CollabID here -->
      <cen>178B:5E25:enpr:360B9A7289F63579E040578C0515638900000018845</cen>
    </scope>
    <members>
      <add>
        <actor>
          <item>
            <!-- User 1 -->
            <!-- Define a User's CollabID here -->
            <cen>178B:5E25:user:63386283615A46D59306642C37BF3D07000000000000</cen>
          </item>
        </actor>
      </add>
    </members>
    <properties>
      <add>
        <item>
          <name>isMwiEnabled</name>
          <value>true</value>
          <description>MWI Enabled</description>
        </item>
        <item>
          <name>VMPhoneNumber</name>
          <value>18665552020</value>
          <description>VM DNIS</description>
        </item>
        <item>
          <name>OperatorTransferNumber</name>
          <value>13125554399</value>
          <description>Operator or AA Phone Number</description>
        </item>
        <item>
          <name>isOperatorConfigured</name>
          <value>true</value>
          <description>Enable Operator Transfer Number</description>
        </item>
        <item>
          <name>StoreMsgMenuLocale</name>
          <value>en-US</value>
          <description>Facility Default Message Leaving Language</description>
        </item>
      </add>
    </properties>
  </group>
</groups>
```

```

        <name>RetrieveMsgMenuLocale</name>
        <value>en-US</value>
        <description>Facility Default Message Retrieval Language</description>
    </item>
    <item>
        <name>isUserLookupEnabled</name>
        <value>true</value>
        <description>Allow User Directory Lookup</description>
    </item>
    <item>
        <name>isGlobalLookupFallbackEnabled</name>
        <value>true</value>
        <description>Allow User Directory Lookup Local to Global</description>
    </item>
    <item>
        <name>PhoneList</name>
        <value>1866</value>
        <description>Common prefix of all interal phone numbers for Lookup by
Phone number for directory and Voicemail GUI</description>
    </item>
    <item>
        <name>ExtensionTranslationRules</name>
        <value>1866555????|1866556????</value>
        <description>Enable users to enter extentions instead of full e.164
phone numbers</description>
    <!-- The first rule allow users to enter 51234, which maps to 18665551234 -->
    <!-- The second rule allows users to enter 61234, which maps to 18665561234 -->

    </item>
</add>
</properties>
</group>
</groups>

```

[Example 11–2, "Sample Modifying Facility XML File"](#) shows the XML for modifying a facility. Note that when modifying a group, you do not provide the enterprise identifier, so you should remove the <scope> element tags from a file you previously used to create the group. The facility group's identifier is used. In this example, a second user (User 2) is added to the group.

Example 11–2 Sample Modifying Facility XML File

```

<?xml version="1.0" encoding="utf-8"?>
<groups>
  <group type="grup"
cen="178B:5E25:grup:63386283615A46D59306642C37BF3D0700000000003F">
    <name>18665552020</name>
    <description>18665552020 Voicemail Facility</description>
    <members>
      <remove>
        <actor>
          <!-- User 1 -->
          <cen>178B:5E25:user:63386283615A46D59306642C37BF3D07000000000000</cen>
        </item>
        <item>
          <!-- User 2 -->
          <cen>178B:5E25:user:63386283615A46D59306642C37BF3D07000000000009</cen>
        </item>
        </actor>
      </remove>
    </members>
  </group>
</groups>

```

```
</members>
<properties>
  <remove>
    <item>
      <name>isMwiEnabled</name>
      <value>true</value>
      <description>MWI Enabled</description>
    </item>
    <item>
      <name>VMPhoneNumber</name>
      <value>18665552020</value>
      <description>VM DNIS</description>
    </item>
  </remove>
  <add>
    <item>
      <name>isGlobalLookupFallbackEnabled</name>
      <value>true</value>
      <description>Allow User Directory Lookup Local to Global</description>
    </item>
    <item>
      <name>PhoneList</name>
      <value>1866</value>
      <description>IP Phone GUI User Finder</description>
    </item>
    <item>
      <name>ExtensionTranslationRules</name>
      <value>186655????</value>
      <description>Extension Dialing</description>
    </item>
  </add>
</properties>
</group>
</groups>
```

Configuring the Voicemail Touch-tone User Interface (TUI)

For the Voicemail TUI to function the only required action is to create users with the required attribute values, as described in "[Creating Voicemail Users](#)" on page 11-12. Once a user is created you can call into the voice service, and leave and listen to voice messages.

Enabling HTTPS for Cisco VXML Enabled Device Access to Oracle Beehive

To enable the Cisco VXML router to access Oracle Beehive via HTTPS, the Oracle Beehive system must be configured for HTTPS, and the Voicemail Component Properties must be modified to map the RecordPlaybackURIs and RecordStreamURIs properties to the HTTPS URIs.

The Cisco VXML device also needs to import the Oracle Beehive application server's CA certificate, to enable access to Oracle Beehive using HTTPS.

Perform the following steps:

1. Configure HTTPS for this Oracle Beehive instance

For instructions, see "Changing HTTP Port" in Chapter 12, "Oracle Beehive Post-Installation Procedures" in the *Oracle Beehive Installation Guide* for your platform (Linux or Solaris only).

2. Modify the voicemail facility object's properties, using the following commands, replacing the items in angle-brackets (<>) with the appropriate values:

```
beectl> modify_property --component <voice enterprise alias or object ID>
--name RecordPlaybackURIs --value
https://<host>:<port>/voice-servlet/vmail/data/shared/playback

beectl> modify_property --component <voice enterprise alias or object ID>
--name RecordStreamURIs --value https://<host>:<port>/voice-servlet/vmail/crs

beectl> activate_configuration
```

To secure HTTPS between Cisco VXML-enabled Routers and Oracle Beehive, you need to import the Oracle Beehive certificate into the IOS device during device configuration. Configure your Cisco VXML router for HTTPS application access using the following steps:

1. From Internet Explorer, access the Oracle Beehive Application Server with [https://<ServerIP>:<port>/]
Use the server and HTTP port for the computer hosting the Oracle Beehive tier.]
The **Security Alert** dialog box displays
2. Click **View Certificate**
The **Certificate** dialog box displays
3. Select the **Details** tab
<All> will be highlighted in the **Show** drop-down list
4. Click **Copy to File**
The **Certificate Export Wizard** dialog appears
5. Click **Base-64 encoded X.509 (.CER)** and then click **Next**
6. Specify a file name in the **File to Export** dialog box and then click **Next**
7. Click **Finish**
An **Export was Successful** message displays.
8. Click **OK** and close the **Security Alert** dialog box.
9. Open the exported file in a text editor and copy the text that appears between the ---BEGINCERTIFICATE-- and --END CERTIFICATE-- tags.
You are now ready to copy the Oracle Beehive Application Server certificate information to the IOS device
10. Access the IOS device in privileged EXEC mode

Note: For more information about managing the IOS device, refer to the Cisco IOS Command-Line Interface documentation

11. Access global configuration mode by entering the configuration terminal
12. Create and enroll a trustpoint by entering the following commands:

```
crypto pki trustpoint xxxx
en terminal
revocation-check none
exit
```

Where `xxxx` is a trustpoint name

The IOS device exits configuration terminal mode and returns to privileged EXEC mode

13. To copy the certificate exported to the text file to the IOS device, perform the following steps:

- a. Enter:

```
crypto pki auth xxxx
```

Where `xxxx` is the trustpoint name specified in step 12

- b. Paste in the certificate you copied from the text file in Step 9

- c. Enter:

```
quit
```

A message displays describing the certificate attributes, and a confirmation prompt appears

14. Enter:

```
Yes
```

A message reports that the certificate was successfully imported

15. Associate the imported certificate with the http client by entering the following command:

```
Enter http client secure-trustpoint xxxx
```

Where `xxxx` is the trustpoint name specified in the previous steps

You have finished importing the certificate.

Configuring Cisco IP Phone Voicemail GUI Application

The IP Phone GUI is a Cisco phoneXML application that is served from Oracle Beehive to the Cisco IP Phones, for use on the Cisco IP Phone's graphical display. This application is supported by Cisco Hard Phone 7970 Series as well as Cisco IP communicator, which is a software phone. In order for the voicemail GUI advanced features (play audio and return call) to function properly, the internal Cisco IP phone's Web server must be enabled. These Web servers are enabled by default, but some deployments disable them. Also, the Cisco IP Phones do not support HTTPS, so in order for the IP phones to access the XML application, the Oracle Beehive application server must allow HTTP access from the phone to the URI `http://<beehive server>:<port>/voice-servlet/cisco-ip-phones`.

Note: Cisco IP Communicator version 7.0.1 is supported. Other versions may not be fully compatible.

The configuration for the voicemail GUI is determined by the network topology and how Cisco Unified Call Manager is deployed.

The following are the points of communication:

- HTTP and RTP from the Oracle Beehive application tier to the IP Phones

- HTTPS from the Oracle Beehive application tier to the Cisco Call Manager AXL interface
- Standard client traffic flow of HTTP from the Cisco IP phones to the Oracle Beehive application tier

The communication needed from the Oracle Beehive application tier to the IP Phones is necessary because Oracle Beehive will push requests to the IP Phones via HTTP utilizing the internal Web server running on the Cisco IP phones. Also, to play back voicemail audio files, RTP communication needs to be enabled from the Oracle Beehive application tier to the IP phones. The Oracle Beehive application tier needs to communicate to the Cisco Call Manager AXL interface to look up the IP Phone's registered IP address, to push HTTP commands and RTP streams for audio playback.

To configure the voicemail GUI the properties shown in [Table 11–2, "Cisco IP Phone Recommended Deployment Properties"](#) need to be set, depending on your network topology and Cisco Call Manager Configuration. These properties can be set in the Voice Enterprise preference set or in the facility group file. The values in **bold** are the recommended values to define.

Note: To make the voicemail GUI available from the Cisco IP Phones, you must also set certain properties, as described in ["Configuring the Voicemail GUI and Message Waiting Indicator"](#) on page 11-22.

Table 11–2 Cisco IP Phone Recommended Deployment Properties

Preference Name	Type	Example Value	Default Value	Description
IpPhoneUserName	STRING	bhvmgui	none	Defines the account name provisioned in Cisco Call Manager that has device control over the user's device
IpPhonePassword	STRING	password	none	Defines the account password provisioned in Cisco Call Manager that has device control over the user's device

Table 11–2 (Cont.) Cisco IP Phone Recommended Deployment Properties

Preference Name	Type	Example Value	Default Value	Description
AxlConfigA	STRING	https://<CCM HOST>:8443/axl/,https://<CCM HOST>:8443/realtimeservice/services/RisPort,,,bhvmgui,password,f,f	none	<p>The comma separated fields, in order, are:</p> <ol style="list-style-type: none"> 1. The AXL URL 2. The realtime info URL 3. The proxy host 4. The proxy port 5. The user provisioned for AXL access (read only) 6. The password for the AXL user 7. Whether certificate validation is enabled for SSL connections to CCM 8. Whether hostname verification is enabled for SSL connections to CCM
RTPServerURI	STRING	http://bigip-beehive.example.com/voice-servlet/rtp_server/RtpServer.jsp	http://<beehive_middle_tier>:<PORT>/voice-servlet/rtp_server/RtpServer.jsp	The URI to access the RTP Server
RTPServerUserName	STRING	RtpClient	none	This value must be defined but the username and password is only used for internal communication
RTPServerPassword	STRING	RtpPassword	none	This value must be defined but the username and password is only used for internal communication

Table 11–2 (Cont.) Cisco IP Phone Recommended Deployment Properties

Preference Name	Type	Example Value	Default Value	Description
MwiCcmHost	String	callmanger.example.com	none	The Cisco Call Manager IP address where the MWI SIP trunk is defined
MwiCcmPort	INTEGER	5060	none	The Cisco Call Manager port of the MWI SIP trunk
IpPhoneRtpMinPort	INTEGER	20480	none	Minimum value for allowed RTP port range to IP phones
IpPhoneRtpMaxPort	INTEGER	32768	none	Maximum value for allowed RTP port range to IP phones
IpPhoneHttpProxyHost	STRING	internal-proxy.example.com	none	Defines the proxy needed for the Oracle Beehive tier to access the internal IP Phones
IpPhoneHttpProxyPort	INTEGER	80	none	Defines the proxy port needed for the Oracle Beehive tier to access the internal IP Phones
MwiAlgorithm	STRING	SHA1PRNG	none	Defines the algorithm used to created the SIP NOTIFY messages for MWI
MwiSipProxyHost	STRING	sip-proxy.example.com	none	SIP Proxy Host that is authorized to send SIP messages to Cisco Call Manager SIP MWI trunk
MwiSipProxyPort	INTEGER	5060	none	SIP Proxy Port that is authorized to send SIP messages to Cisco Call Manager SIP MWI trunk

Table 11–2 (Cont.) Cisco IP Phone Recommended Deployment Properties

Preference Name	Type	Example Value	Default Value	Description
MwiMinLocalPort	INTEGER	5060	none	Lowest port in the range that Oracle Beehive will use to send and receive SIP messages
MwiMaxLocalPort	INTEGER	5080	none	Highest port in the range that Oracle Beehive will use to send and receive SIP messages
MwiSourcePhone	STRING	18665551234	none	Voicemail DNIS phone number that is presented in the SIP NOTIFY message
RtpServerHost	STRING	bigip-beehive.example.com	none	Host where the RTP Server is running
RtpServerMinPort	INTEGER	20480	20480	Minimum port range used by the RTP server to send RTP packets
RtpServerMaxPort	INTEGER	32768	32768	Maximum port range used by the RTP server to send RTP packets
RtpServerPacket Size	INTEGER	160	160	RTP Packet Size. Leave at default for Cisco IP phones

Configuring the Voicemail GUI and Message Waiting Indicator

Telephones within your deployment may have a Message Waiting Indicator (MWI), which lights up when the phone number has received one or more voicemail messages. Additionally, sophisticated phones may have a Graphical User Interface (GUI), which presents a menu of choices to the user.

In order to enable the use of voicemail features through an IP phone's GUI, or to enable MWI operation, you must set the properties identified in [Table 11–3, "Voicemail Properties"](#). These are a subset of the total set of properties, which are listed in [Table 11–1, "Voicemail Properties"](#) on page 11-5. You must set these properties at the Enterprise level of scope, but you can also set them at a Facility level; at a given Facility, the Facility-level properties override the global Enterprise-level properties.

The properties listed are for configuring using local IP Phone authentication.

To use local IP Phone configuration every device in Cisco Call Manager needs to be configured with the authentication server URL: `http://<Beehive_HOST>:<port>/voice-servlet/cisco-ip-phones/authenticate.jsp`

["Configuring Enterprise Preferences"](#) on page 11-11 describes how to set the voicemail properties.

Table 11-3 Voicemail Properties

Property	Value	Type
IpPhonePasswordSeed		STRING
IpPhonePasswordAlgorithm	AES	STRING
IpPhonePasswordNumBits	128	INTEGER
IpPhoneRtpMinPort	20480	INTEGER
IpPhoneRtpMaxPort	32768	INTEGER
RtpServerHost	<hostname> (of the computer on which Beehive is installed)	STRING
RTPServerUserName	RTPClient	STRING
RTPServerPassword	RTPpwd	STRING
RTPServerURI		STRING
MwiAlgorithm	SHA1PRNG	STRING
MwiCcmHost		STRING
MwiCcmPort	5060	INTEGER
MwiSipProxyHost		STRING
MwiSipProxyPort	15060 (optional property)	INTEGER
MwiMinLocalPort	5060	INTEGER
MwiMaxLocalPort	5080	INTEGER
MwiSourcePhone		STRING

Cisco Router Configuration

This section assumes you have Cisco IOS and Cisco Call Manager Administration configuration experience. Cisco 2800/3800 Series or AS5400XM with IOS version 12.4(11T) or greater with VXML feature set, is required for Oracle Beehive voicemail.

To configure your Cisco router hardware for use with Oracle Beehive, perform the following steps:

1. [Configure Translation Rule](#)
2. [Configure Global VXML Configuration Options](#)
3. [Configure Voicemail VXML Application](#)
4. [Configure Transcoding \(Optional\)](#)

Each step is described in its own section.

Configure Translation Rule

The router must be configured for full E.164 phone numbers, which map to the phone numbers defined in users' UDS record `voice_principal` and `tel:` address attributes.

These rules will change, depending on the incoming DNIS delivery method provided by your PRI.

The following example shows how 5-digit DNIS delivery is expanded:

```
voice translation-rule 10
  rule 2 /\(^627..$\)/ /170332\1/
!
!
voice translation-profile FULL_E164_IN
  translate called 10
!
voice-port 0/3/0:23
  translation-profile incoming FULL_E164_IN
```

Configure Global VXML Configuration Options

Make the following configurations:

- http client cache memory pool 8192

Note: If you record prompts with file sizes larger than the http client cache, the prompt will not be cached. Latency between the gateway and the Oracle Beehive instance could cause a timeout while Oracle Beehive waits for the gateway to send audio files. The end-user may experience long pauses while using the TUI, and a timeout could abruptly end the call without any error messages.

If you experience this issue, increase the http client cache memory pool size to a size larger than your largest prompt file.

- http client cache memory file 200
- http client cache refresh 300
- http client response timeout 30
- ivr prompt memory 4096
- ivr prompt streamed http

Note: NOTE: If using https then ivr prompt streamed must be set to none.

- ivr record memory system 48000
- ivr record memory session 1500
- vxml tree memory 100000
- vxml version 2.0

The following settings are necessary only if fax is to be supported on the PSTN side of the ingress gateway:

- fax receive called-subscriber \$d\$
- fax interface-type fax-mail
- mta send server bh-midtier port 5025

Note: This is the SMTP service running on your Oracle Beehive server.

- mta send with-subject both
- mta send mail-from hostname example.com
- mta send mail-from username \$\$

Configure Voicemail VXML Application

Use the following configuration settings:

```
application
  service vm_bh http://beehive.example.com:7777/voice-servlet/vmail/start.vxml
!
dial-peer voice 500 voip
  description Voicemail Pilot Number
  huntstop
  service vm_bh
  session protocol sipv2
  incoming called-number 18665551234
  dtmf-relay rtp-nte sip-notify
  codec g711ulaw
  no vad
```

Use the following settings to configure Fax functionality (if you will be setting up fax service with Oracle Beehive).

```
!
dial-peer voice 310 mmoip
  service fax_on_vfc_onramp_app out-bound
  destination-pattern 1866.....
  information-type fax
  session target mailto:faxservice@example.com
  image encoding MH
```

Note: To complete the configuration of fax functionality with Oracle Beehive, follow the steps in ["Configuring Oracle Beehive Fax"](#) on page 11-28.

Configure Transcoding (Optional)

Cisco only supports uncompressed audio on IVR application call legs. Depending on the Call Manager deployment configuration using compressed audio, it is possible to configure transcoding on the local router in order to support multiple codecs.

Use the following configuration settings:

```
sccp local Loopback0
sccp ccm 192.188.175.105 identifier 1 priority 1 version 5.0.1
sccp
!
sccp ccm group 1
  description Reston Lab transcoding for IPC
  bind interface Loopback0
  associate ccm 1 priority 1
  associate profile 1 register restontvg2
```

```
!  
dspfarm profile 1 transcode  
codec g711ulaw  
codec g711alaw  
codec g729ar8  
codec g729abr8  
codec gsmfr  
codec g729r8  
codec pass-through  
maximum sessions 12  
associate application SCCP
```

Cisco Unified Call Manager Configuration

To configure the Touch-tone User Interface (TUI), you must do the following:

1. Create SIP Trunk
2. Create Voicemail Pilot Number
3. Create Voicemail Profile
4. Assign Voicemail Profile to users' Directory Phone Number

To configure Voicemail GUI, you must do the following:

1. Create CCM User that is associated to all users' phone devices
2. Create a Read Only user with AXL Access
3. Define IP Phone Services in CCM

Note: You can also configure the language availability for Oracle Beehive voicemail users from Cisco Call Manager user interface. The default location for the CCM User interface is `https://<CCM_HOST>:8443/ccmuser/`.

For locale support on Cisco Call Manager, the appropriate locale packs must be installed. Once the locales are installed in Cisco Call Manager, Oracle Beehive voicemail users can select the locale using the Cisco Call Manager User interface.

Configuring the Auto Attendant

This section contains the following topics:

- [Installing an Auto Attendant](#)
- [Associating an Auto Attendant with a Facility](#)
- [Voicemail TUI Optional Steps](#)
- [Auto Attendant Administration Commands](#)

Installing an Auto Attendant

To install an Auto Attendant, create an Auto Attendant Markup Language (AAML) document for your attendant, and record your voice prompts, making sure that they are 8000Hz mono u-law WAVE files. Make the files accessible from the computer on which Oracle Beehive is installed.

Use the `beectl add_attendant` command to add the auto attendant:

```
beectl> add_attendant --file <AAML file>
```

Next, use the `beectl upload_attendant_prompt` command to upload your recorded audio prompts:

```
beectl> upload_attendant_prompt --name <attendant name> --language <language>
--type <type> --file <prompt file>
```

You can review the VoiceXML at

```
http://<host>:<port>/voice-servlet/aa/view/<attendant name>.do.
```

Associating an Auto Attendant with a Facility

To associate an Auto Attendant to a facility, first create one or more facilities by following the directions in ["Managing Facilities"](#) on page 11-12. When you create a Facility, you create both a group and a config object. The group's values, such as identifier, Name, Properties, and so on, are defined in an XML file. See [Example 11-1](#), ["Sample Facility XML File"](#) on page 11-14 for an example Facility XML file.

See ["Managing Facilities"](#) on page 11-12 for instructions on creating a Facility.

Create a facility object by using the `beectl add_voice_facility` command:

```
beectl> add_voice_facility --name <voicemail service phone number OR auto
attendant service phone number>
```

Next, use the `beectl list_groups` command to get the name and identifier of the group:

```
beectl> list_groups
```

Use the `beectl modify_property` command:

```
beectl> modify_property --component <config object id or Alias defined by adding
the configuration object> --name FacilityCollabId --value <Group identifier>
```

Use the `beectl activate_configuration` command to validate and activate the configuration changes:

```
beectl> activate_configuration
```

For more information on administration commands for the auto attendant, see ["Auto Attendant Administration Commands"](#) on page 11-28.

Voicemail TUI Optional Steps

An Enterprise is not required for Facility values to be used. You may specify all settings at the Facility level, by following the instructions in the section ["Creating a Facility"](#) on page 11-12, and skipping these Enterprise steps.

You can create Enterprise-level settings, which are used by default if you do not specify a given setting at the Facility level.

To create Enterprise-level settings:

1. Load `defaultfacility.xml` as a group (see ["Sample Facility XML File"](#) on page 11-13). The default Facility must be created for Enterprise values to be used
2. Set Enterprise values, by following the instructions in ["Configuring the Enterprise"](#) on page 11-11

Auto Attendant Administration Commands

This section lists the various `beectl` commands used for managing the Auto Attendant, along with descriptions.

- **add_attendant**
Adds an auto attendant using the information from an AAML document. The name of the attendant will be taken from the name attribute of the root element, "attendant". It should not contain any white space.
- **delete_attendant**
Deletes an auto attendant by name.
- **modify_attendant**
Updates an existing auto attendant with a new AAML file. You can also use this command to rename an existing attendant.
- **upload_attendant_prompt**
Uploads a prompt for an attendant.
- **delete_attendant_prompt**
Deletes a prompt.
- **list_attendant_prompts**
List all prompts that have been uploaded for an attendant.
- **list_attendant_aaml**
Prints the Auto Attendant Markup Language document for an auto attendant.
- **modify_ip_phone_password_seed**
Resets the seed of the password generator for IP phones. To send commands to an IP phone the requestor must be authenticated. The password is generated based on information in the phone but must be seeded to ensure security.

Configuring Oracle Beehive Fax

Oracle Beehive fax functionality is enabled using the same Cisco infrastructure as voicemail. Once you have configured Cisco Call Manager for voicemail, you must perform additional configuration to enable fax.

In Oracle Beehive, the Fax Message Service provides configuration options. You must also create a special Fax User, and set up a business event notification for that user. The Cisco Call Manager will send all faxes to that user, and then the notification will trigger a process that forwards the fax to the intended Oracle Beehive user.

To configure Cisco Call Manager for fax, see "[Configure Voicemail VXML Application](#)" on page 11-25

Perform the following procedure to create the special Oracle Beehive fax account and set up the notification:

1. Add a user using the `beectl add_user` command. You can use any name for the account; in this example, `FaxUser` is used. Give the user an e-mail address, such as `faxuser@<yourcompany.com>`:

```
beectl> add_user --family_name FaxUser --scope <your enterprise identifier>
--login_id faxuser --login_password <password> --address BUSINESS_
1:mailto:faxuser@example.com
```

2. After the fax service user is provisioned, get the Entity ID (EID) of that user by using the `beectl list_users` command with the `--entity_format` option:

```
beectl> list_users --user user=faxuser --show ALL --entity_format id
```

The EID is the portion of the user's CollabID following the `:user:` segment. For example, if the `list_users` command produced the following output:

```
User Identifier: 05C1:7403:user:9AE5E38909BE41C181BAD42CE1B88F5300000000000E
```

Then the EID is `9AE5E38909BE41C181BAD42CE1B88F53000000000000E`

3. Use the XML file shown in [Example 11-3](#) to create a subscription, by using the `beectl add_event_subscription` command:

```
beectl> add_event_subscription --file <name of XML file>
```

When you have completed this step, Oracle Beehive is ready to receive fax messages from the Cisco Call Manager.

Example 11-3 Sample Fax User Event Subscription

In this example, replace the bolded EID with the EID of your Oracle Beehive fax user.

```
<?xml version="1.0" encoding="UTF-8" ?>
<eventSubscription xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="eventSubscription.xsd">
  <subscriberId></subscriberId>
  <name>ES_MSG_DELIVERED_EVENT_FAX_LISTENER</name>
  <description>Subscription to sync Fax repository for incoming fax </description>
  <eventName>ES_MSG_DELIVERED</eventName>
  <Condition>
    <simple>
      <leftSide>RAWTOHEX(custom_attributes.recipient_eid)</leftSide>
      <operator>=</operator>
      <rightSide>'E603E73114BB4944AF5A6E5014D520E10000000003C1'</rightSide>
    </simple>
    <!--
  <conjunction>
</conjunction>-->
    <!--
  <disjunction>
</disjunction>-->
    </Condition>
  <Action>
    <isPLSQLAction>F</isPLSQLAction>
    <actionString>oracle.ocs.management.model.FaxMessageService:ES_MSG_
DELIVERED</actionString>
    <ActionPreferenceInfos>
      <actionPreferenceInfo>
        <key></key>
        <value></value>
      </actionPreferenceInfo>
    </ActionPreferenceInfos>
  </Action>
</eventSubscription>
```

Managing Oracle Beehive Events, Policies, and Workflows

This module describes how to view business events, create and manage policies including the default Oracle Beehive policies, and create and manage workflows. This module contains the following sections:

- [Introduction to Beehive Events, Policies, and Workflows](#)
- [Managing Beehive Events](#)
- [Managing Beehive Policies](#)
- [Managing Beehive Workflows](#)

Introduction to Beehive Events, Policies, and Workflows

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 and/or workflow. 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.

You can also use special workflow policies to trigger workflows, such as an approval process when a file is uploaded to a particular workspace, or when a particular resource is reserved for a meeting. Oracle Beehive deployments include the Oracle BPEL Process Manager. Workflow policies are capable of triggering BPEL workflows. By this mechanism, a workflow can be triggered from various business events.

Oracle Beehive workflow can be created and managed using the `beectl` command line interface. Corresponding BPEL processes can be designed using Oracle JDeveloper. This BPEL Process archive (a jar file) is used during the creation of a workflow in Oracle Beehive.

Workflows allow you to create a multi-stage process flow, in which users make decisions relating to whichever events were used to trigger a workflow policy. Workflow tasks are exposed to users in Oracle Beehive clients as task assignments.

This section includes the following topics:

- [Introduction to Beehive Events](#)
- [Introduction to Beehive Policies](#)
- [Introduction to Beehive Workflows](#)

Introduction to Beehive Events

Oracle Beehive provides about 400 business events on which you can base policies, generate notifications, and drive workflows. 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 (or workflow 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 Oracle Beehive Release 1, 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 or workflows 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 gets executed.

The actions are defined by internal Oracle Beehive services and included in all deployments automatically. These actions are exposed through the policies. You may only create new, custom actions for the use of custom workflows.

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, such as for creating custom workflows, 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 6-25 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.

Introduction to Beehive Workflows

Oracle Beehive provides integration with Oracle BPEL and the BPEL Process Manager, to allow you to automate approvals and other workflow tasks. You can invoke workflows in two ways:

- By configuring workflow templates, which determine which default workflows are invoked (and when, that is, which container) based on content related operations
- By configuring custom workflow policies, which invoke custom workflows you can define

Note: For detailed instructions on creating and using custom workflow policies, see *Oracle Beehive Application Developer's Guide*.

For custom workflows, you can create custom workflow policies, which determine which events will trigger workflows, and how the workflow process should complete.

Workflows are exposed to users through tasks, in Oracle Beehive clients including Microsoft Outlook, CalDAV clients, mobile clients, and so on. When a user needs to approve of an action, a task is delivered to the user. The user can then approve or

disapprove of the task. When all approval tasks are complete, the workflow notifies the original caller of its completion.

You can create and manage custom templates by editing XML-formatted files, and importing them to Oracle Beehive from the command line. See ["Managing Beehive Workflows"](#) on page 12-19 for details.

Managing Beehive Events

Oracle Beehive provides about 400 events for use by policies, workflows, 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. In Oracle Beehive Release 1, 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 ATTRIBS_T	Type: BEE_CODE.WS_DOCUMENT_EVENT_
Name: ARTIFACT_ATTRIBUTES ATTRIBS_T	Type: BEE_CODE.AM_COMMON_EVENT_
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 policies 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. 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](#)

See Also: "Policy Definitions" in Chapter 1, "Oracle Beehive XML File Reference" of the *Oracle Beehive Administrator's Reference Guide*,

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 12–1, "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 12–1 User Attributes in Provisioning and Deprovisioning Policy Conditions

User Account Attribute	Field Name in Policy XML File	Valid Values
Office Location	<code>custom_attributes.office_location</code>	Value of office location
Job Title	<code>custom_attributes.job_title</code>	Value of job title
Is external user	<code>custom_attributes.external_user</code>	Y or N
Organization	<code>custom_attributes.organization_cid</code>	CollabID of organization
Manager	<code>custom_attributes.manager_cid</code>	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 12–1, "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 12–1 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</containerId>

  <templateId></templateId>
  <name>UserProvisioningPolicy</name>
  <RuleInfos>
    <RuleInfo priority="1">
      <name>Provisioning_rule1</name>
      <eventName>USER_PROVISIONING</eventName>
      <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 12-2, "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 12-2 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:355E4C9DBE6147FDE040578C030929770000000001D0</containerId>
  <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 12–1, "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 12–3, "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 12–3 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</containerId>

  <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 12–4, "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 12-4 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>
    </RuleInfo>
  </RuleInfos>
</PolicyInfo>
</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. This section contains the following topics:

- [Creating New Policies](#)
- [Editing Existing Policies](#)
- [Deleting Custom Policies](#)

See Also: "Policy Definitions" in Chapter 1, "Oracle Beehive XML File Reference" of the *Oracle Beehive Administrator's Reference Guide*,

Creating New Policies

There are two types of actions: Java and PLSQL. The default provisioning, deprovisioning, password and auditing 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, or auditing 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, upload an XML formatted policy document using the `beectl add_policy` command:

```
beectl> add_policy --file <full path to the policy xml file>
```

See Also: For detailed instructions on how to create a custom workflow policy using workflow actions, see *Oracle Beehive Application Developer's Guide*.

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, which evaluates whether the document is owned by another user, and if so, triggers an approval workflow. 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.

To create a new policy template, 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>
```

To create a new policy based on a policy template, 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 workflow 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 (and dynamic workflow template 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 `--event_name` option. For example:

```
beectl> list_events --event_name WORKSPACE_UPDATED
```

Managing Beehive Workflows

Oracle BPEL Process Manager is installed and configured, and seeded with default workflows, during Oracle Beehive installation. The `beectl start` and `beectl stop` commands start and stop the BPEL Process Manager along with the rest of the Oracle Beehive processes.

The default workflows provided by Oracle Beehive make use of workflow templates to define when (and which) default workflow is invoked. However, custom workflows make use of policies using the same policy XML files as other policies, with the exception that the actions are different. These custom actions must be created.

You can configure custom workflows using custom workflow policies. You can configure the default serial and parallel vote workflows using workflow templates.

See Also:

- For detailed instructions on creating and using custom workflow policies, see *Oracle Beehive Application Developer's Guide*.
- For documentation on using the BPEL Process Manager, see *Oracle BPEL Process Manager Quick Start Guide*."

You can list all workflows (both default and custom workflows) that have been initiated from Oracle Beehive, and check their status, using the `beectl list_workflows` command:

```
./beectl list_workflows [ --status <status of the workflow to be listed> ] [
--creator <creator id of the workflow to be listed> ]
```

This section contains the following topics:

- [Types of Oracle Beehive Workflow](#)
- [Oracle Beehive Workflow Tasks](#)
- [Oracle Beehive Workflow Templates](#)
- [Using Dynamic Attributes with Custom Workflow Templates](#)
- [Oracle Beehive Workflow Template Commands](#)

Types of Oracle Beehive Workflow

There are two workflow processes provided with Oracle Beehive:

- Serial workflows
- Parallel vote workflows

A serial workflow may include one or more user tasks, but each task applies to only one user. If a serial workflow includes multiple tasks, Oracle Beehive creates and assigns each new task when the previous task is completed. If a serial workflow has

multiple approvers, once any approver rejects the task, no further approvers will be assigned approval tasks.

A parallel vote workflow has a voting mechanism, so that several users collectively make a decision about the workflow tasks. Oracle Beehive determines whether to accept or reject the workflow based on the percentage of approvals. A workflow is accepted if greater than some specified percentage of approvers approve of the workflow.

Note: A parallel vote workflow with a fractional vote percentage of 0.001 or less will fail. A parallel vote workflow must not include more than 999 users.

Oracle Beehive Workflow Tasks

Workflow approvals are exposed within Oracle Beehive and standards-based clients through tasks. When a workflow needs a user input (usually an approval), Oracle Beehive generates a task and assigns it to that user. The approver can approve or reject the workflow process by completing or declining the Oracle Beehive task. The operation that triggered the workflow to be initiated, such as creating a document in a team workspace, will then either be allowed to complete or be rejected. Oracle Beehive will automatically update the BPEL Process Manager, which will continue to process the workflow.

When the workflow is completed, the operation that triggered the workflow is either allowed to complete, or cancelled.

A task generated by a default workflow template includes the following components:

- A title
- An initiator: the user whose action triggered the workflow is shown in Oracle Beehive as the task assigner, while the approvers are the task assignees
- A Justification: this may be an explanation for why the user action is justified. The workflow template provides a default justification. Future versions of Oracle Beehive may allow the triggering user to edit the justification
- A URL, which is the WebDAV link to whatever location (document or folder) was involved in triggering the workflow. For example, if the workflow is triggered by DOCUMENT_DELETE, the link goes to the document being deleted
- A payload: in the current release of Oracle Beehive, this content is not included. Future versions of Oracle Beehive will allow additional text in the payload to be included in the task
- The client's usual task buttons or links to accept or decline (disapprove) the task

Oracle Beehive Workflow Templates

See Also: "Workflow Templates," in Chapter 1, "Oracle Beehive XML File Reference" of the *Oracle Beehive Administrator's Reference Guide*

Oracle Beehive provides two default workflow processes, the serial workflow and the parallel vote workflow. You can determine when and where one such workflow process is triggered using workflow templates. Workflow templates allow you to create a variety of serial and parallel vote workflows at various levels of context, and

triggered by various different events. This allows the same BPEL workflow process to be re-used in different contexts. Oracle Beehive workflow templates make use of "template keys", which are a set of specialized synchronous events on which workflows may be triggered. Oracle Beehive workflows may only be triggered by these specific events.

Because the workflow template keys are synchronous events, the workflow process blocks the event from completing. For example, a workflow triggered by the DOCUMENT_DELETE template key must complete successfully before the document will be deleted by Oracle Beehive. If the approval request is declined, the event does not occur or is cancelled; so if a workflow triggered by DOCUMENT_DELETE is declined, the document will not be deleted.

You can manage workflows by creating and using workflow templates. A Workflow template customizes the attribute settings of a default workflow.

[Table 12–2, "Oracle Beehive Workflow Template Keys"](#) lists the template keys available for Oracle Beehive workflow templates.

Table 12–2 Oracle Beehive Workflow Template Keys

Template key	Description
DOCUMENT_CREATE	Invoked when a new document is created (uploaded)
DOCUMENT_UPDATE	Invoked when an existing document is altered
DOCUMENT_CHECKIN	Invoked when a document is checked in
DOCUMENT_DELETE	Invoked when an existing document is deleted
DOCUMENT_COPY	Invoked when a copy of an existing document is made
DOCUMENT_MOVE_TO	Invoked when a document is moved in to the designated location (container)
DOCUMENT_MOVE_FROM	Invoked when a document is moved from the designated location (container)
FOLDER_DELETE	Invoked when an existing folder is deleted
FOLDER_COPY	Invoked when a copy is made of an existing folder
FOLDER_MOVE_TO	Invoked when a folder is moved to the designated location (container)
FOLDER_MOVE_FROM	Invoked when a folder is moved from the designated location (container)

A sample workflow template XML file, and its corresponding DTD, is located in the following folder:

\$ORACLE_HOME/bee hive/templates/workflow

Workflow templates include the following elements which you must specify:

Note: A size limit exists (measured in bytes) for the <defaultValue> element of the **tasktitle**, **initiator**, and **approvers** attributes, in the serial and parallel vote workflows. Tasktitle is limited to 75 bytes, while initiator and approver are limited to 100 bytes. Note that when using multi-byte character sets, each character may consume more than one byte, reducing the effective limitation in characters to a smaller number.

- **name:**
A name for this workflow template
- **description:**
A description of this workflow template
- **schemaname:**
This value determines whether this is a serial or parallel vote workflow. For serial workflows, use `SerialWorkflow`. For a parallel vote workflow, use `ParallelVoteWorkflow`. No other values are supported for workflow templates
- **scope:**
A container for this workflow template. In the example, an enterprise is specified, but you can specify any level of scope: enterprise, organization, workspace, or folder. Use the `collabID` or alias for any such scope
- **templatekey:**
One of the `templatekey` values shown in [Table 12–2](#) on page 12-21. The `templatekey` determines the user operation that will trigger the workflow
- **default justification:**
The justification appears in the body of the task generated by a workflow
- **default workflow timeout:**
Specify a maximum time for the workflow to be completed before it will automatically complete with a failure (not approved) result. The timeout format is in the form `P<days>D<hours>H<minutes>M`, where each value must be an integer. For example, `P1D12H` specifies one day and twelve hours (1.5 days), `P5D` specifies five days, and `P0D2H30M` specifies zero days, two hours, and thirty minutes
- **default approver timeout:**
Specify a maximum time for any approver to approve of a workflow task before the task automatically completes with a failure (not approved). If there are multiple approvers, each approver will get this amount of time to approve a task; however, the default workflow timeout takes priority, even if any approvers have not yet reached the default approver timeout
- **default approvers:**
One or more approvers, separated by commas. This field is limited to 100 bytes. For a serial workflow, each approver is given a task in turn, while in a parallel vote workflow, all approvers are given the task at the same time. Approvers are specified using their login ID (Primary Principal) only; see ["About User Account Principals and Credentials"](#) on page 3-9 for details about the Primary Principal user ID

Caution: The user identifiers that are specified in the template are case-sensitive and must exactly match the identifier defined for the user in UDS. If a user identifier is not an exact match, other tasks in Oracle Beehive will not be created properly.

- **default payload:**

After listing the justification, the a task generated by the workflow contains the payload text

- **default tasktitle:**

The title of the tasks generated by this workflow template. This field is limited to 75 bytes

- **default initiator:**

When a workflow is generated from this workflow template, the user that took the action (the templatekey event) is the initiator. However, if a failure of some kind occurs, the value of the initiator element becomes the default workflow initiator, to prevent the workflow from becoming invalid. The initiator should be a user that is guaranteed to be valid, such as the BEEadmin user. This field is limited to 100 bytes.

- **default votepercent:** (Parallel Vote workflow templates only)

This integer value is the percentage of votes needed to approve the workflow. Once the votepercent is reached in approvers, the workflow will complete

Note: A parallel vote workflow with a fractional vote percentage of 0.001 or less will fail. A parallel vote workflow must not include more than 999 users.

Note: In the previous lists, items listed as "default" may be overridden in some cases when the workflow is generated, depending on how the user interface exposes the workflow. The option to substitute a new value for a default value does not exist in any client in Release 1 of Oracle Beehive, but future versions of Oracle Beehive clients may include such capability.

[Example 12–5, "Serial Workflow Template XML File"](#) shows an example Serial Workflow template XML file. The items in bold indicate the element values you should specify when creating a new serial workflow template.

Note: The workflow template attributes all have elements called <prompted> and <required>. Eventually these elements may be used by clients to determine whether a value is required, and whether a user triggering the workflow is prompted to enter a new value for the attribute. However, in Oracle Beehive Release 1, these element values are not used (although they must be present in the workflow template). Future releases of Oracle Beehive will expose this functionality.

Similarly, the payload element is not exposed in Oracle Beehive Release 1.

Example 12–5 Serial Workflow Template XML File

```
<?xml version="1.0" encoding="UTF-8" ?>
<WorkflowTemplateInfo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://xml.oracle.com/beehive/workflow/workflow_template.xsd"
                    xmlns="http://xml.oracle.com/beehive/workflow"
```

```
isfinal="false">
  <name>Serial WF approval</name>
<description>A template for invoking Serial Workflow</description>
<schemaname>SerialWorkflow</schemaname>
<scope>enpr=my_enterprise_name</scope>
<templatekey>DOCUMENT_CREATE</templatekey>
<Attributes>
  <attribute>
    <name>justification</name>
    <defaultValue>justification Approval required to add a document to the team workspace</defaultValue>
    <prompted>true</prompted>
    <required>false</required>
    <final>false</final>
  </attribute>
  <attribute>
    <name>workflowtimeout</name>
    <defaultValue>P1D</defaultValue>
    <prompted>true</prompted>
    <required>false</required>
    <final>false</final>
  </attribute>
  <attribute>
    <name>approvertimeout</name>
    <defaultValue>P1D</defaultValue>
    <prompted>true</prompted>
    <required>false</required>
    <final>false</final>
  </attribute>
  <attribute>
    <name>approvers</name>
    <defaultValue>example.user,example2.user</defaultValue>
    <prompted>true</prompted>
    <required>false</required>
    <final>false</final>
  </attribute>
  <attribute>
    <name>payload</name>
    <defaultValue>defaultpayload</defaultValue>
    <prompted>true</prompted>
    <required>false</required>
    <final>false</final>
  </attribute>
  <attribute>
    <name>tasktitle</name>
    <defaultValue>Approval required to add a document to the team workspace</defaultValue>
    <prompted>true</prompted>
    <required>false</required>
    <final>false</final>
  </attribute>
  <attribute>
    <name>initiator</name>
    <defaultValue>beeadmin</defaultValue>
    <prompted>true</prompted>
    <required>false</required>
    <final>false</final>
  </attribute>
</Attributes>
</WorkflowTemplateInfo>
```

Example 12–6, "Parallel Vote Workflow Template XML File" shows an example Parallel Vote Workflow template XML file. The items in bold indicate the element values you should specify when creating a new parallel vote workflow template.

Example 12–6 Parallel Vote Workflow Template XML File

```
<?xml version="1.0" encoding="UTF-8" ?>
  <WorkflowTemplateInfo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://xml.oracle.com/beehive/workflow/workflow_template.xsd"
                      xmlns="http://xml.oracle.com/beehive/workflow"

isfinal="false">
  <name>Parallel workflow approval when documents are deleted from the WORKSPACE_
NAME workspace</name>
  <collabid></collabid>
  <description>A Template for Parallel Workflow</description>
  <schemaname>ParallelVoteWorkflow</schemaname>

<containerid>36CB:5780:wstm:DD649BD7DAA641CF811F5A7E1D9BA02D000000000037</containerid>
  <templatekey>DOCUMENT_DELETE</templatekey>
  <Attributes>
    <attribute>
      <name>justification</name>
      <defaultValue>Request to delete a document in the WORKSPACE_NAME
workspace</defaultValue>
      <prompted>true</prompted>
      <required>false</required>
    </attribute>
    <attribute>
      <name>workflowtimeout</name>
      <defaultValue>P1D</defaultValue>
      <prompted>true</prompted>
      <required>false</required>
    </attribute>
    <attribute>
      <name>approvertimeout</name>
      <defaultValue>P1D</defaultValue>
      <prompted>true</prompted>
      <required>false</required>
    </attribute>
    <attribute>
      <name>approvers</name>
      <defaultValue>jamie.rancourt,tait.mccarthy</defaultValue>
      <prompted>true</prompted>
      <required>false</required>
    </attribute>
    <attribute>
      <name>payload</name>
      <defaultValue>defaultpayload</defaultValue>
      <prompted>true</prompted>
      <required>false</required>
    </attribute><attribute>
      <name>initiator</name>
      <defaultValue>beeadmin</defaultValue>
      <prompted>true</prompted>
      <required>false</required>
    </attribute>
    <attribute>
```

```

        <name>tasktitle</name>
        <defaultValue>Request to delete a document in the WORKSPACE_NAME
workspace</defaultValue>
        <prompted>true</prompted>
        <required>false</required>
    </attribute>
    <attribute>
        <name>votepercent</name>
        <defaultValue>51</defaultValue>
        <prompted>false</prompted>
        <required>false</required>
    </attribute>
</Attributes>
</WorkflowTemplateInfo>

```

Using Dynamic Attributes with Custom Workflow Templates

A workflow template is used to configure a content operation to be driven by a workflow. The workflow that drives the operation can be a default workflow process (Parallel Vote or Serial), or it can be a custom workflow.

As with policies, you can use dynamic attributes in your custom workflow templates.

See Also: [Using Dynamic Policy Attributes](#)

You can use any of the actions listed in [Table 12-2, "Oracle Beehive Workflow Template Keys"](#) in your workflow template. In the following example, the DOCUMENT_DELETE template key is used:

```

<?xml version="1.0" encoding="UTF-8" ?>
  <WorkflowTemplateInfo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

  xsi:schemaLocation="http://xml.oracle.com/beehive/workflow/workflow_template.xsd"
    xmlns="http://xml.oracle.com/beehive/workflow"

  isfinal="false">
    <name>My Dynamic Workflow</name>
    <collabid></collabid>
    <description>This workflow uses dynamic attributes</description>
    <schemaname>Name of the Workflow Schema (ParallelVoteWorkflow, SerialWorkflow,
or Custom)</schemaname>

    <containerid>36CB:5780:wstm:DD649BD7DAA641CF811F5A7E1D9BA02D000000000037</containerid>
    <templatekey>DOCUMENT_DELETE</templatekey>
    <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>

```



```
</WorkflowTemplateInfo>
```

You must replace the `attribute1` to `attribute n` values of the `<name>` elements with the parameters of the workflow. For example, for `ParallelVoteWorkflow`, `attribute1` to `attribute n` would be `justification`, `vote percent`, and so forth. These values can come from the following sources:

- Values provided by the user for attributes that are marked as prompted
- Default values provided in the default templates
- Dynamic workflow template attributes

To use a dynamic workflow template attribute, set the default value to be one of the following string constants:

- **#workflow.entity:** The name of the entity (document or folder) that is triggering this workflow
- **#workflow.container:** The name of the container (workspace or folder) where the triggering entity resides
- **#workflow.srccontainer:** The name of the source container (workspace or folder) in a copy or move operation
- **#workflow.dstcontainer:** The name of the destination container (workspace or folder) in a copy or move operation
- **#workflow.initiator:** The name of the actor who caused the event which triggered the workflow
- **#workflow.startdate** The date, in string format, when this workflow was triggered. The format is the same as calling `Date.toString()`.

For example, to trigger a workflow that has a parameter called `docname` with the actual document name, the template could have the following attribute:

```
<attribute>
<name>docname</name>
<defaultValue>#workflow.entity</defaultValue>
```

At runtime, the `#workflow.entity` will be replaced with the actual name of the document.

Oracle Beehive Workflow Template Commands

You can list the workflow templates currently implemented in a given context (workspace or folder) with the `beectl list_workflow_templates` command:

```
beectl list_workflow_templates --scope <container where the workflow is
configured> --template_key <workflow template key>
```

Specify the template key to list all workflow templates in that container that are triggered by that template key.

When you list a workflow template, its unique identifier is also listed.

You can export a workflow template. This creates an XML-formatted file (in a location you specify), which you can then review or edit. Export a workflow template by using the `beectl export_workflow_template` command:

```
beectl export_workflow_template --workflow_template <identifier of the workflow
template to be exported> --destination <destination directory of the generated xml
file>
```

Specify the unique identifier (found by using the `list_workflow_templates` command) of the workflow template you want to export to a file.

You create a workflow template by uploading an XML-formatted Workflow Template file to Oracle Beehive. To create a workflow template, use the `beectl add_workflow_template` command:

```
beectl add_workflow_template --file <full path to the workflow template xml file>
```

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](#)

About Access Control

In Release 1 of 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
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	

Table 13–1 (Cont.) Default Oracle Beehive Privileges

Privilege	Access Granted
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
WORKFLOWTASK_MGR	Administrator access to assigning and managing workflow tasks
WORKFLOW_MGR	Administrator access to workflow functionality
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 1, 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, WORKFLOW_MGR, WORKSPACE_MGR] | true | |
| acrd=workspace-document-coordinator,enpr=mycompany | workspace-document-coordinator | |
| +RWDEO | [CONTENT_MGR, FORUM_MGR, MARKER_MGR, MODIFY_ACL, VERSION_MGR, WORKFLOW_
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 'acs=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 13-9.

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.

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 Templates](#)
- [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 "[Managing Records Management](#)" on page 6-25.

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_audit_events` command:

```
beectl> list_audit_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 8-4.

For more information about business events in Oracle Beehive, see: [Chapter 12, "Managing Oracle Beehive Events, Policies, and Workflows"](#).

About Audit Templates

An audit template is a collection of audit events, on a particular theme. Oracle Beehive includes a collection of pre-seeded audit templates, which you can use unmodified, or as examples when developing your own audit templates. You must specify an audit template whenever you create an audit policy.

About Audit Policies

An audit policy combines a collection of events to be audited (in the form of an audit template) 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 Templates

An audit template defines a collection of events that will be audited. Oracle Beehive comes pre-seeded with a collection of audit templates around a variety of themes.

This section contains the following topics:

- [Listing Audit Templates](#)
- [Creating Audit Templates](#)
- [Deleting Audit Templates](#)
- [Example Audit Templates](#)

Listing Audit Templates

You can list the audit templates using the `beectl list_audit_templates` command:

```
beectl> list_audit_templates
```

You can also review the audit templates using Oracle Beekeeper:

1. Log in to Oracle Beekeeper, and under **Enterprises**, click **Audit**
2. Select the **Templates** tab. All audit templates are listed
3. Select any template, and then click on the **Auditable Events** tab to view the categories of events that the template audits. Categories included in the template are listed in the **Selected Auditable Events** column

[Table 14–1](#) lists the pre-seeded audit templates, and their identifiers.

Table 14–1 Pre-Seeded Audit Templates

Event Category Name	Identifier
Access control	autp=Access control events
Address book/contact events	autp=Address book/contact events
Artifact events	autp=Artifact events
Audit management events	autp=Audit management events
BPEL task events	autp=BPEL task events
Calendar related events	autp=Calendar related events
Client application related events	autp=Client application related events
Device management/profile events	autp=Device management/profile events
Enterprise events	autp=Enterprise events
LDAP synchronization profile events	autp=LDAP synchronization profile events
Message events	autp=Message events
Organization events	autp=Organization events
Policy/subscription events	autp=Policy/subscription events
Record management events	autp=Record management events
Search events	autp=Search events
Security events	autp=Security events
Service configuration updated event	autp=Service configuration updated event

Table 14–1 (Cont.) Pre-Seeded Audit Templates

Event Category Name	Identifier
System management events	autp=System management events
Time management events	autp=Time management events
User/group management and provisioning events	autp=User/group management and provisioning events
Wiki-page events	autp=Wiki-page events
Workspace related events	autp=Workspace related events
XMPP events	autp=XMPP events

You can list details about a specific audit template by using the `--name` option to enter the name of any audit template (without the `autp=` prefix):

```
beectl> list_audit_templates --name <Name of the audit template>
```

For example, if you list the `XMPP events` template, the following information is displayed:

```
beectl> list_audit_templates --name "XMPP events"
```

```
Name:                XMPP events
Description:         Audits all XMPP events
Identifier:          autp=XMPP events
Created on:          Feb 11, 2009 7:05:58 AM
Last modified on:    Feb 11, 2009 7:05:58 AM
Event Name:          XMPP events [XMPP_ASYNC_EVENTS]
```

On the `Event Name:` line, after a description (`XMPP events`), one or more event categories are listed; these correspond to a set of events that are captured by the template.

See Also: For a complete listing of all events encompassed by each event category, see ["Audit Events Structure"](#) on page 14-15.

Creating Audit Templates

To create a new audit template using `beectl`, create an audit template XML file, and then use the `beectl add_audit_template` command to upload it to the system:

```
beectl> add_audit_template --file <Full path of the input file>
```

To create a new audit template using Oracle Beekeeper:

1. Log in to Oracle Beekeeper, and under **Enterprises**, click **Audit**
2. Select the **Templates** tab. All audit templates are listed
3. Click **New**, and choose **Audit Template**
4. In the **New Audit Template** window, enter a name and description for the new template, and then select the **Auditable Events** tab
5. Select one or more event categories from the **Available Auditable Events** column, and use the **Move** arrows to move them into the **Selected Auditable Events** column. The template will include all of the events for each category you move to the **Selected Auditable Events** column

6. Click **Apply** to save your changes without closing the New Audit Template window, or click **Save and Close** to save your changes and close the window.

Your new template appears in the list in the **Templates** tab

Deleting Audit Templates

To delete an audit template using `beectl`, use the `beectl delete_audit_template`:

```
beectl> delete_audit_template --template <Audit template identifier>
```

To delete an audit template using Oracle Beekeeper:

1. Log in to Oracle Beekeeper, and under **Enterprises**, click **Audit**
2. Select the **Templates** tab. All audit templates are listed
3. Select a template, and click **Delete**. In the confirmation dialog box, click **OK**. The selected template is deleted

Example Audit Templates

An example audit template file is located in your Oracle Beehive install folder, in the `templates` subfolder: `ORACLE_HOME/bee hive/templates/audit/templ_ex.xml`:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<AuditTemplateInfo>
  <name>Template Name</name>
  <description>Template Description</description>
  <event>Auditable Event A</event>
  <event>Auditable Event B</event>
</AuditTemplateInfo>
```

In the `<name>` element, give your new template a unique name. Use the `<description>` element to describe the types of events included in the template. Use `<event>` elements to contain a list of each event or event category you want to include in the audit template.

An example audit template:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<AuditTemplateInfo>
  <name>Test credential</name>
  <description>Sample test template</description>
  <event>CREDENTIAL_ASYNC_EVENTS</event>
  <event>LOCK_ASYNC_EVENTS</event>
</AuditTemplateInfo>
```

You can include any auditable event or event category. To review a list of auditable events, use the `beectl list_audit_events` command:

```
beectl> list_audit_events
```

To review the list of audit event categories, see ["Audit Events Structure"](#) on page 14-15

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.

When you create an audit policy, you must use an audit template. You can use one of the pre-seeded audit templates, or, you can create a new audit template.

See Also: For information about creating audit templates, see ["Creating and Managing Oracle Beehive Audit Templates"](#) on page 14-3.

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` by 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/beeфик</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:aure: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</UserSession>
    <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/beeфик</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:aorc: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</UserSession>
  <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 14–2](#) lists auditable events related to access control.

Table 14–2 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 14–3](#) lists auditable events related to address books.

Table 14–3 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 14–4 lists auditable events related to artifacts.

Table 14–4 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 14–4 (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 14–5 , "Notification Events")
TOPIC_ASYNC_EVENTS	TOPIC_ARCHIVED TOPIC_MOVED TOPIC_DELETED TOPIC_CREATED TOPIC_UNDELETED TOPIC_UPDATED

[Table 14–5](#) lists auditable events in the sub-category of Notification events.

Table 14–5 Notification Events

Event Subcategory	Events
NOTIFICATION_ASYNC_EVENTS	NOTIFICATION_CREATED NOTIFICATION_UPDATED NOTIFICATION_DELETED

Table 14–5 (Cont.) Notification Events

Event Subcategory	Events
NOTIFICATION_SCHEMA_ASYNC_EVENTS	NOTIFICATION_SCHEMA_DELETED NOTIFICATION_SCHEMA_CREATED NOTIFICATION_SCHEMA_UPDATED

Audit Events

[Table 14–6](#) lists auditable events related to each audit event category.

Table 14–6 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_EVENTS	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

BPEL Task Events

[Table 14–7](#) lists auditable events related to BPEL tasks.

Table 14–7 BPEL Task Events

Event Subcategory	Events
BPEL_TASK_ASYNC_EVENTS	BPEL_TASK_ASSIGNED BPEL_TASK_COMPLETED BPEL_TASK_UPDATED

Calendar Events

Table 14–8 lists auditable events related to calendars.

Table 14–8 Calendar Events

Event Subcategory	Events
CALENDAR_ASYNC_EVENTS	CALENDAR_ADDED CALENDAR_REMOVED CALENDAR_UPDATED
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 14–9 lists auditable events related to client applications.

Table 14–9 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

Table 14–9 (Cont.) Client Application Events

Event Subcategory	Events
CLIENT_APPLICATION_VERSION_ASYNC_EVENTS	CLIENT_APPLICATION_VERSION_DELETED CLIENT_APPLICATION_VERSION_CREATED

Device Management Events

[Table 14–10](#) lists auditable events related to device management.

Table 14–10 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 14–11](#) lists auditable events related to Enterprises.

Table 14–11 Enterprise Events

Event Subcategory	Events
ENTERPRISE_ASYNC_EVENTS	ENTERPRISE_ARCHIVEPURGED ENTERPRISE_DELETED ENTERPRISE_UPDATED ENTERPRISE_CREATED

LDAP Sync Profile Events

[Table 14–12](#) lists auditable events related to LDAP sync profiles.

Table 14–12 LDAP Sync Profile Events

Event Subcategory	Events
LDAP_SYNC_PROFILE_ASYNC_EVENTS	LDAP_SYNC_PROFILE_DELETED LDAP_SYNC_PROFILE_CREATED

Message Events

[Table 14–13](#) lists auditable events related to messages.

Table 14–13 Message Events

Event Subcategory	Events
DISCUSSION_MESSAGE_ASYNC_EVENTS	DISCUSSION_MESSAGE_ARCHIVED DISCUSSION_MESSAGE_DELETED DISCUSSION_MESSAGE_UPDATED DISCUSSION_MESSAGE_CREATED DISCUSSION_MESSAGE_MOVED
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 14–5, "Notification Events" on page 14-17)
VOICE_MESSAGE_ASYNC_EVENTS	VOICE_MESSAGE_MOVED VOICE_MESSAGE_CREATED VOICE_MESSAGE_UPDATED VOICE_MESSAGE_DELETED VOICE_MESSAGE_COPIED

Organization Events

[Table 14–14](#) lists auditable events related to Organizations.

Table 14–14 Organization Events

Event Subcategory	Events
ORGANIZATION_ASYNC_EVENTS	ORGANIZATION_ARCHIVED ORGANIZATION_UPDATED ORGANIZATION_CREATED ORGANIZATION_DELETED

Policy Subscription Events

[Table 14–15](#) lists auditable events related to policies and subscriptions.

Table 14–15 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 14–16](#) lists auditable events related to Records Management.

Table 14–16 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 14–17](#) lists auditable events related to search.

Table 14–17 Search Events

Event Subcategory	Events
SEARCH_ASYNC_EVENTS	SEARCH_FINISHED SEARCH_STARTED

Security Events

[Table 14–19](#) lists auditable events related to security.

Table 14–18 Security Events

Event Subcategory	Events
ACCOUNT_ASYNC_EVENTS	ACCOUNT_LOGIN_SUCCEEDED ACCOUNT_LOGOUT_SUCCEEDED ACCOUNT_LOCKED
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 14–19](#) lists auditable events related to Service configuration updates.

Table 14–19 Service Configuration Update Events

Event Subcategory	Events
SERVICE_CONFIG_UPDATED	SERVICE_CONFIG_UPDATED

System Events

[Table 14–20](#) lists auditable events related to the core Oracle Beehive system.

Table 14–20 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 14–21](#) lists auditable events related to time management.

Table 14–21 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
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
TM_WORKFLOW_ASYNC_EVENTS	TM_WORKFLOW_ASSIGNMENT_APPROVED TM_WORKFLOW_ASSIGNMENT_REJECTED

User Management Events

[Table 14–22](#) lists auditable events related to user management.

Table 14–22 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

Table 14–22 (Cont.) User Management Events

Event Subcategory	Events
USER_ASYNC_EVENTS	USER_UPDATED USER_DELETED USER_PURGED USER_CREATED

Workspace Events

[Table 14–2](#) lists auditable events related to Workspaces.

Table 14–23 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 14–24](#) lists auditable events related to XMPP messaging.

Table 14–24 XMPP Events

Event Subcategory	Events
XMPP_ASYNC_EVENTS	XMPP_FILE_TRANSFERRED XMPP_USER_LOGGEDIN XMPP_USER_LOGGEDOUT

Configuring and Managing Oracle Collaboration Coexistence Gateway

This module explains the concepts and terminology of Oracle Collaboration Coexistence Gateway. Installation instructions and administration tasks are also explained. The following topics are covered in this module:

- [Introduction to Oracle Collaboration Coexistence Gateway](#)
- [Installation and Post-Installation Configuration](#)
- [Administering Oracle Collaboration Coexistence Gateway](#)

Introduction to Oracle Collaboration Coexistence Gateway

The Oracle solution for coexistence between Oracle Beehive and Microsoft Exchange requires an understanding of fundamental terminology and concepts before installation and deployment. This section serves as a general introduction to Oracle Collaboration Coexistence Gateway, including terminology and concepts, while providing a broad overview of the technology and architecture that facilitates coexistence.

Oracle Collaboration Coexistence Gateway is an Oracle proprietary solution that allows Oracle Beehive users to collaborate with Microsoft Exchange users. Additionally, the solution allows Microsoft Exchange users to make use of the full set of Oracle Beehive features without being migrated from Microsoft Exchange. Oracle Collaboration Coexistence Gateway can be broken down to these services:

- Oracle Connector for Exchange
- Oracle Change Notification Service for Exchange
- Oracle Communication Service
- Oracle Beehive Coexistence Service

The first three services run on the computer or computers where Oracle Connector for Exchange and Oracle Change Notification Service for Exchange were installed. The last service is on Oracle Beehive.

These services update and propagate data between the two coexisting systems. Depending on the deployment, the information being updated and propagated may include events, contacts, tasks, and e-mails.

You can centrally manage Oracle Beehive Coexistence from the `beectl` command line, and from Oracle Beekeeper, including managing components of Coexistence that are installed on remote servers.

This section includes the following topics:

- [Terminology](#)
- [Concepts](#)
- [Architectural Overview](#)

Terminology

Read the following list of terminology to better understand the terms that will be used throughout this module:

Active Directory Mail Enabled User

Users who exists in the Windows domain and can send and receive e-mail.

Active Directory Contact

A Microsoft Active Directory contact is a user who does not have a Microsoft Exchange mailbox, and who resides on a different system. For example, an Active Directory contact could be an Oracle Beehive user.

Active Directory Exchange User

A Microsoft Active Directory Exchange user is a mail enabled user who exists in the Windows domain and has an Exchange mailbox. Among other objects, the Exchange mailbox can contain e-mail, calendar, and task items.

Coexisting Third-Party User

Users residing on a third-party system, such as Microsoft Exchange; however, they have the ability to collaborate and view free/busy information of Oracle Beehive users. Oracle Beehive users are able to collaborate with coexisting third-party users and view their free/busy information. These users can also do the following:

- Collaborate with other users on the same system
- View Oracle Beehive users' free/busy information
- Collaborate with Oracle Beehive users as though they were users native to Microsoft Exchange

Non-Coexisting Third-Party User

Users residing solely on an existing third-party system. These users are not aware of the existence of Oracle Beehive. No coexistence solution has been provided to this group of users. Although these users may exist in the Oracle Beehive Global Address List, an Oracle Beehive user cannot view the free/busy information or cross-schedule these users. All communication between Oracle Beehive users and these types of users on third-party systems is accomplished using e-mail. These users can do the following:

- Collaborate with other users on the same system
- View Oracle Beehive users' free/busy information
- Collaborate with Oracle Beehive users by e-mail

Oracle Beehive Coexistence Service

The Oracle Beehive Coexistence Service is deployed in an OC4J container (BEEAPP) that is part of every Oracle Beehive deployment. The service is automatically enabled when Oracle Beehive is installed and is required to implement the Oracle

Collaboration Coexistence Gateway. The Oracle Beehive Coexistence Service relays data between the Oracle Connector for Exchange and Oracle Beehive.

Oracle Beehive Users

Users with Oracle Beehive as their data repository for all content types. These users can do the following:

- Use all available Oracle Beehive services
- Collaborate with other Oracle Beehive users
- Collaborate with coexisting third-party users as though they were Oracle Beehive users

Oracle Change Notification Service for Exchange

The Oracle Change Notification Service for Exchange is an Oracle Collaboration Coexistence Gateway service that is responsible for the following tasks:

- Detecting changes to free busy information for coexisting third-party users
- Dispatching changes to Oracle Connector for Exchange

Oracle Communication Service

The Oracle Communication Service plays a vital role in information exchange between Oracle Beehive and Microsoft Exchange. The Oracle Communication Service is responsible for the transmission of all message types between the two systems. It distinguishes between different message types, including e-mails, events, and contacts, to facilitate the transmission of information between Oracle Beehive and Microsoft Exchange.

Oracle Connector for Exchange

The Oracle Connector for Exchange is an Oracle Collaboration Coexistence Gateway service that is responsible for the following tasks:

- Converting outgoing messages to a format accepted by Oracle Beehive and dispatching the delivery to the Oracle Communication Service
- Delivering meeting requests, responses, cancellations, task requests, responses, and updates arriving from Oracle Beehive to Microsoft Exchange users
- Updating free/busy information

Public Free/Busy Folder

The public free/busy folder is a shared folder that resides on the Microsoft Exchange server. Its purpose is to store availability information for all Active Directory individuals, including Active Directory users and contacts. The availability information that is stored in the folder is populated by Microsoft Exchange clients, such as Microsoft Outlook.

Concepts

This section defines Oracle Collaboration Coexistence Gateway concepts that will be used throughout this module.

Note: An understanding of the ["Terminology"](#) section is required to fully understand the concepts outlined in this section.

Cross-Scheduling

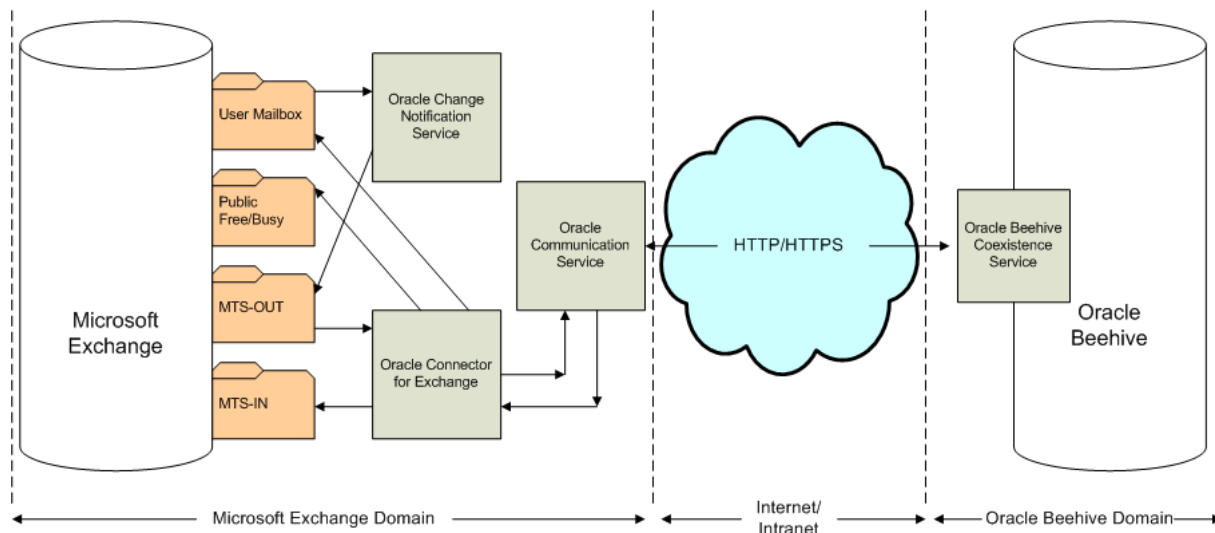
This method of coexistence allows Oracle Beehive users and third-party system users to collaborate with each other transparently. This consists of event scheduling and task assignment, including free/busy lookup and post-invitation event updates. In this method of coexistence, users in either system are unaware of the existence of the other system, and can collaborate transparently.

Architectural Overview

This section includes a diagram of the architectural overview specific to Oracle Collaboration Coexistence Gateway.

Figure 15–1 depicts key components of Oracle Collaboration Coexistence Gateway in conjunction with Microsoft Exchange, including services, third-party software, and information flow:

Figure 15–1 Oracle Collaboration Coexistence Gateway Architecture



Installation and Post-Installation Configuration

This section includes important information and cross-references that are necessary to install and configure the Oracle Collaboration Coexistence Gateway solution. This section contains the following topics:

- [Installation](#)
- [Post-Installation Configuration](#)
- [Post-Installation Tasks](#)

Installation

The *Oracle Beehive Installation Guide for Microsoft Windows* contains all of the information that you will need related to preparing for and installing the components of the Oracle Collaboration Coexistence Gateway solution:

- For information about what you will require before beginning the installation, see "Oracle Collaboration Coexistence Gateway" in Module 3 of the *Oracle Beehive Installation Guide for Microsoft Windows*.

- For information about the installation sequence, and installation of the Oracle Collaboration Coexistence Gateway, see "Oracle Collaboration Coexistence Gateway Install Help" in Module 6 of the *Oracle Beehive Installation Guide for Microsoft Windows*.

Note: After installing the components of the Oracle Collaboration Coexistence Gateway solution, a Remote Coexistence System must be configured in the Oracle Beehive deployment.

Follow the steps outlined in the "[Configuring a Coexistence System on Oracle Beehive](#)" section to complete the installation.

Follow the steps outlined in the "[Post-Installation Configuration](#)" section to complete the configuration.

Post-Installation Configuration

After installing Oracle Beehive, Oracle Connector for Exchange, and Oracle Change Notification Service for Exchange; and "[Configuring a Coexistence System on Oracle Beehive](#)", additional post-installation configuration must be completed before the Oracle Collaboration Coexistence Gateway is fully functional.

Complete all of the tasks in this list to get the Oracle Collaboration Coexistence Gateway up and running:

1. Adjust quota restrictions on Microsoft Exchange. For additional information about this task, refer to "[Adjusting Quota Restrictions for the Oracle Connector for Exchange Mailbox](#)".
2. If your Microsoft Exchange deployment does not allow users to receive e-mails from external addresses, the Microsoft Exchange administrator must reconfigure Microsoft Exchange to allow incoming external e-mails.
3. Configure a relay server on Oracle Beehive. For additional information about this task, refer to "[Setting Up E-mail Relay Routing](#)" in [Chapter 8, "Managing Oracle Beehive E-mail"](#).
4. Make sure your local domains are configured in the E-mail Service. You can do this with Oracle Beekeeper, or by running the following beectl command:

```
beectl> append_value --component _EmailService --name LocalEmailPatterns
--value "*@<yourdomain>"
```

Where <yourdomain> represents your local domain.

Note: For the configuration changes to take effect, you must run the `activate_configuration beectl` command.

5. Set the CoexUserLocal E-mail Service property to FALSE using the following beectl command:

```
beectl> modify_property --component _EmailService --name CoexUserLocal --value
FALSE
```

Note: For the configuration changes to take effect, you must run the `activate_configuration beectl` command.

6. If your Microsoft Exchange and Oracle Beehive deployments have local users with the same e-mail domain, set the `AcceptInvalidLocalUsers` E-mail Service property to `TRUE`. You can do this using Oracle Beekeeper, or by using the following `beectl` command:

```
beectl> modify_property --component _EmailService:TransportProperties --name
AcceptInvalidLocalUsers --value TRUE
```

Post-Installation Tasks

This section is a compilation of information related to post-installation administration tasks that may or may not be required after installing Oracle Connector for Exchange and Oracle Change Notification Service for Exchange. Read each preamble carefully, because not all deployment scenarios will require the post-installation tasks described in this section. For a list of required post-installation tasks, refer to ["Post-Installation Configuration"](#).

This section includes the following topics:

- ["Adjusting Quota Restrictions for the Oracle Connector for Exchange Mailbox"](#)
- ["Installing the Oracle Change Notification Service for Exchange on the Same Host as Microsoft Exchange"](#)
- ["Redirecting E-mail Back to Microsoft Exchange for Delivery to Oracle Beehive"](#)

Adjusting Quota Restrictions for the Oracle Connector for Exchange Mailbox

During the installation of Oracle Connector for Exchange, a mailbox is created for the service. This mailbox will be given the following name: `Oracle-coexConnExch <host>`. This folder should not be subject to default mail quota policies imposed on regular user folders.

For Oracle Connector for Exchange to function as it was intended, the Microsoft Exchange administrator must remove all mailbox quota policies that may be enforced on this folder before attempting a coexistence deployment.

Installing the Oracle Change Notification Service for Exchange on the Same Host as Microsoft Exchange

While installing Oracle Connector for Exchange on the same computer as Microsoft Exchange, a prompt appears giving you the "Option to Configure Oracle Change Notification Service". If you select **No** to this option and complete the installation, you will not be able to install Oracle Change Notification Service for Exchange on that computer using the Oracle Universal Installer.

If after choosing not to install Oracle Change Notification Service for Exchange on the same computer as Microsoft Exchange and Oracle Connector for Exchange, you then want to install Oracle Change Notification Service for Exchange on that same computer, follow these instructions:

1. On the computer hosting the Oracle Connector for Exchange and Microsoft Exchange, navigate to the `C:\oracle\product\<version>\exconnector_1\beehive\collabcoex_connector`.

Where `<version>` represents the folder with the version number of Oracle Connector for Exchange.

2. Run the following command:

```
coexctl.exe install_eventsink --install_directory
```



```
C:\oracle\product\<version>\exconnector_1\beehive\collabcoex_connector --admin_
account <WindowsUserAccount> --admin_password <psw>
```

Where:

- <version> represents the folder with the version number of Oracle Connector for Exchange
- <WindowsUserAccount> represents a Windows user account, with the rights and privileges outlined in the "Windows User Account" section of the *Oracle Beehive Installation Guide for Microsoft Windows*.
- <psw> represents the <WindowsUserAccount> user's password

If the installation was successful, the following message will be returned:

```
EventSink registered successfully.
```

3. Start the Oracle Change Notification Service for Exchange. For instructions on starting the Oracle Change Notification Service for Exchange, refer to ["Starting Oracle Change Notification Service for Exchange"](#).

Note: If this method is used to install Oracle Change Notification Service for Exchange, the Oracle Universal Installer cannot be used to uninstall it. To uninstall Oracle Change Notification Service for Exchange using the command-line tool, run the following command:

```
coexctl.exe uninstall_eventsink
```

Redirecting E-mail Back to Microsoft Exchange for Delivery to Oracle Beehive

By default, Microsoft Exchange redirects all messages directed towards coexisting Oracle Beehive users to the Coexistence Connector, which is then responsible for delivering those messages to Oracle Beehive. However, the following limitations apply when the Coexistence Connector sends the e-mails to Oracle Beehive itself:

- SMTP SSL/TLS is not supported
- SMTP Authentication is not supported
- Receipt notifications are not supported (although delivery and non-delivery reports are supported)

Beginning with Oracle Beehive Release 1 (1.5), Oracle recommends configuring the Coexistence Connector to redirect all e-mail messages addressed to Oracle Beehive users sent from Microsoft Exchange, back to Microsoft Exchange for delivery.

By having Microsoft Exchange deliver the messages, MIME conversion will be performed by Microsoft Exchange and not by the Coexistence Connector.

If you do not make this configuration change, the listed limitations apply, and you will have to configure e-mail sending options in two locations. Having Microsoft Exchange deliver e-mails enables you to configure e-mail sending, retry intervals, and other Microsoft Exchange features through the usual Microsoft Exchange UI.

Example Redirection Configuration

Suppose that one of the domains of your Oracle Beehive deployment is `example.com`. You configure the system in the following way:

1. The Coexistence Connector is configured to replace all `example.com` addresses in the e-mail envelope with `special.example.com`, and then redirect it to Exchange for delivery.
2. Microsoft Exchange is configured to send messages addressed to `special.example.com` to Oracle Beehive. The message forwarded from the Coexistence Connector will therefore be delivered by Exchange directly to Oracle Beehive.
3. The Oracle Beehive E-mail Service is configured with a pre-resolution rule, such that upon receipt of the message, Oracle Beehive will rewrite `special.example.com` addresses back to `example.com` before delivering the message to the Oracle Beehive user.

To enable this feature, you must add one or more pre-resolution rule for incoming e-mails in Oracle Beehive to rewrite addresses in the e-mail envelopes. A simple rule that rewrites `special.example.com` to `example.com` is listed in [Example 15–1, "Simple Rule for Rewriting E-mail Domains"](#). The rule is implemented using a new (in Oracle Beehive Release 1 (1.5)) Mail Service feature, `RewriteDomainAction`, which can perform regular expressions on the e-mail domain.

Caution: Rewrite actions are run in sequence. This means rewrite result can be (mistakenly) matched by subsequent actions. You should carefully craft the matching patterns to avoid this.

Example 15–1 Simple Rule for Rewriting E-mail Domains

```
<?xml version="1.0" encoding="UTF-8"?>
<java version="1.5.0_16" class="java.beans.XMLDecoder">
  <object class="oracle.ocs.mail.service.ruleengine.RuleEngine">
    <void property="name">
      <string>5e8f2f1b59e7de1f:-bd35894:11d64fcfda8:-8000</string>
    </void>
    <void property="rules">
      <array class="oracle.ocs.mail.service.ruleengine.Rule" length="2">
        <void index="0">
          <object
class="oracle.ocs.mail.service.ruleengine.mail.actions.RewriteDomainAction">
            <void property="name">
              <string>rewrite domain rule</string>
            </void>
            <void property="pattern">
              <string>(special.) (example.com)</string>
            </void>
            <void property="replaceString">
              <object class="java.util.ArrayList">
                <void method="add">
                  <string>\2</string>
                </void>
              </object>
            </void>
          </object>
        </void>
        <void index="1">
          <object class="oracle.ocs.mail.service.ruleengine.TopLevelAction">
            <void property="argumentType">
              <string>oracle.ocs.mail.service.mom.MsgContentAndEnvelope</string>
            </void>
            <void property="group">
```

```

        <string>THE_RULE</string>
    </void>
    <void property="name">
        <string>THE_RULE</string>
    </void>
    <void property="rootRule">
        <string>rewrite domain rule</string>
    </void>
    </object>
</void>
</array>
</void>
</object>
</java>

```

If you need to rewrite multiple domains (because your Oracle Beehive users have more than one valid e-mail domain), you can write a more complex rule. An example of a composite rule that rewrites multiple domains is shown in [Example 15-2, "Composite Rule for Rewriting Multiple E-mail Domains"](#). In this example, `special.example.com` is rewritten to `example.com`, and `special.example.anotherdomain.com` is rewritten to `example.anotherdomain.com`.

Example 15-2 Composite Rule for Rewriting Multiple E-mail Domains

```

<?xml version="1.0" encoding="UTF-8"?>
<java version="1.5.0_16" class="java.beans.XMLDecoder">
  <object class="oracle.ocs.mail.service.ruleengine.RuleEngine">
    <void property="name">
      <string>5e8f2f1b59e7de1f:62dfa7e3:11d6f346721:-8000</string>
    </void>
    <void property="rules">
      <array class="oracle.ocs.mail.service.ruleengine.Rule" length="4">
        <void index="0">
          <object
class="oracle.ocs.mail.service.ruleengine.mail.actions.RewriteDomainAction">
            <void property="name">
              <string>rewrite domain rule</string>
            </void>
            <void property="pattern">
              <string>(special.)(example.com)</string>
            </void>
            <void property="replaceString">
              <object class="java.util.ArrayList">
                <void method="add">
                  <string>\2</string>
                </void>
              </object>
            </void>
          </object>
        </void>
        <void index="1">
          <object
class="oracle.ocs.mail.service.ruleengine.mail.actions.RewriteDomainAction">
            <void property="name">
              <string>rewrite domain rule 2</string>
            </void>
            <void property="pattern">
              <string>(special.)(example.)(anotherdomain.com)</string>
            </void>

```

```
<void property="replaceString">
  <object class="java.util.ArrayList">
    <void method="add">
      <string>\2</string>
    </void>
    <void method="add">
      <string>com</string>
    </void>
  </object>
</void>
</object>
</void>
<void index="2">
  <object class="oracle.ocs.mail.service.ruleengine.CompositeRule">
    <void property="children">
      <array class="java.lang.String" length="2">
        <void index="0">
          <string>rewrite domain rule</string>
        </void>
        <void index="1">
          <string>rewrite domain rule 2</string>
        </void>
      </array>
    </void>
    <void property="mode">
      <string>SEQUENCE</string>
    </void>
    <void property="name">
      <string>rewrite domains</string>
    </void>
  </object>
</void>
<void index="3">
  <object class="oracle.ocs.mail.service.ruleengine.TopLevelAction">
    <void property="argumentType">
      <string>oracle.ocs.mail.service.mom.MsgContentAndEnvelope</string>
    </void>
    <void property="group">
      <string>THE_RULE</string>
    </void>
    <void property="name">
      <string>THE_RULE</string>
    </void>
    <void property="rootRule">
      <string>rewrite domains</string>
    </void>
  </object>
</void>
</array>
</void>
</object>
</java>
```

Once you have written the XML to configure your Oracle Beehive E-mail Service pre-resolution rule, perform the following steps to apply it on your Oracle Beehive instance:

Note: In the following procedure, if you have already configured pre-resolution rules, they will be overwritten with the new rules created in your XML file. You can use the `beectl list_properties` command to see the current value of the `PreResolutionRules` property; if it contains a configuration that you need to preserve, contact your Oracle Support representative for assistance.

1. Use the `beectl list_components` command to find the identifier of your E-mail Service Transport Properties component:

```
beectl> list_components
```

Search the results for the `EmailService.TransportProperties` component, and copy the corresponding ID. For example:

```
-----+-----
-
EmailService.TransportPropert | e7ffb5e7-757c-46a1-961d-b3b12fd4ad14
ies                           |
-----+-----
-
```

2. Use the `beectl modify_property` command to upload the XML file:

```
beectl> modify_property --component <Transport_properties_ID> --name
PreResolutionRules --file <XML_File>
```

3. Validate and activate your proposed configuration using the `beectl activate_configuration` command:

```
beectl> activate_configuration
```

Once the changes are applied in Oracle Beehive, run the following for each Oracle Beehive primary e-mail domain, on each Coexistence Connector machine:

```
$ORACLE_HOME\beehive\collabcoex_connector\coexctl.exe set_property --file $ORACLE_
HOME\beehive\collabcoex_connector\conf\ConnectorConfiguration.xml --section
smtpRedirection --property <example.com> --value <special.example.com>
```

Substitute the correct domain names for the `--property` and `--value` values.

Administering Oracle Collaboration Coexistence Gateway

This section explains how to perform various administrative tasks related to Oracle Collaboration Coexistence Gateway, and includes the following topics:

- [Oracle Beehive Coexistence Service Administrative Tasks](#)
- [Oracle Connector for Exchange Administrative Tasks](#)
- [Oracle Change Notification Service for Exchange Administrative Tasks](#)

Oracle Beehive Coexistence Service Administrative Tasks

This section describes how to perform various administrative tasks related to the Oracle Beehive Coexistence Service. 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*.

This section includes the following topics:

- [Configuring a Coexistence System on Oracle Beehive](#)
- [Listing Oracle Beehive Remote Coexistence Systems](#)
- [Modifying Oracle Beehive Remote Coexistence Systems](#)
- [Provisioning Users for Coexistence](#)
- [Viewing the Registration State of Coexisting Users](#)
- [Deprovisioning Users from an Oracle Beehive Remote Coexistence System](#)

Configuring a Coexistence System on Oracle Beehive

This section includes information about adding a coexistence system using the `add_coexistence_system` command.

The `add_coexistence_system` command takes either a `--url`, or both a `--host_name` and `--port`. You must use the `--url` option if you need to customize the URI portion of the URL. Otherwise, you can use either method of specifying the host.

Optionally, the `--use_https` parameter can be used along with the `--host_name` and `--port` parameters to specify that the HTTPS protocol will be used.

The following examples show how to format input using each method:

```
beectl add_coexistence_system --host_name example.com --port 80 --sitekey
```

This command generates a URL of

```
http://example.com:80/coexConnector/CoexMLPort
```

```
beectl add_coexistence_system --host_name example.com --port 80 --use_https  
--sitekey
```

This command generates a URL of

```
https://example.com:80/coexConnector/CoexMLPort
```

```
beectl add_coexistence_system --url http://example.com --sitekey
```

This command generates a URL of

```
http://example.com:80/coexConnector/CoexMLPort
```

```
beectl add_coexistence_system --url https://example.com/uri --sitekey
```

This command generates a URL of `https://example.com:80/uri`

Follow these steps to add a coexistence system to your Oracle Beehive deployment:

1. Determine the fully qualified domain name and port of your Oracle Connector for Exchange installation.

2. Navigate to the `ORACLE_HOME/bee hive/bin` directory on the system hosting the Oracle Beehive deployment.
3. Run the following command:

```
beectl> add_coexistence_system {--url
http://<OracleConnectorForExchangeHost>:<port>/coexConnector/CoexMLPort |
--host_name <coexistence_system_name> --port <port_number> --use_https }
--sitekey [<obfuscated_sitekey> --obfuscated] [--timezone_alias_namespace
<timezone_alias_namespace>]
```

Notes: The argument passed to the `--sitekey` option is the password defined during the installation of the Oracle Connector for Exchange. For more information about the Site Key value, see "Oracle Collaboration Coexistence Gateway Install Help" in Module 6 of the *Oracle Beehive Installation Guide for Microsoft Windows*.

The argument passed to the `--url` option is displayed in the summary screen of the Oracle Connector for Exchange installation.

Where:

- `<OracleConnectorForExchangeHost>` represents the fully qualified domain name of the system hosting Oracle Connector for Exchange
- `<port>` represents the HTTP port of your Oracle Connector for Exchange installation
- `<coexistence_system_name>` represents the name that you want to give to the Remote Coexistence System
- `<obfuscated_sitekey>` represents the password defined during the installation of the Oracle Connector for Exchange, which you have obfuscated using the `beectl obfuscate` command
- `<timezone_alias_namespace>` is a string that corresponds to a time zone namespace in Oracle Beehive and is used to map time zones from Exchange to Oracle Beehive and vice versa. By default this is set to MSFT. Unless advised to do so by Oracle support, do not change this value.

Listing Oracle Beehive Remote Coexistence Systems

Oracle Beehive administrators may require details about the Oracle Beehive Remote Coexistence Systems configured on their deployment. This section includes information about listing all configured Remote Coexistence Systems using the `list_coexistence_systems` command.

Run the following command to list the configured Remote Coexistence Systems on your Oracle Beehive deployment:

```
beectl> list_coexistence_systems
```

The command will return output similar to the following example:

```
-----
Coexistence remote collaboration system details.
-----
```

```
Name                : Exchange
URL                 : http://10.156.42.99:7777/coexConnector/CoexMLPort
Timezone alias namespace: MSFT
```

For additional details, use the `--show_more` option:

```
beectl> list_coexistence_systems --show more
```

Coexistence remote collaboration system details.

```
Id                :
121A:2A16:cors:56F748517E90371EE040578CF71854800000000186A6
Name              : Exchange
URL               : http://10.156.42.99:7777/coexConnector/CoexMLPort
Timezone alias namespace: MSFT
-----
```

Configuration properties status.

```
Modified On       : 9/19/08 10:17 AM
Last synchronization : 9/19/08 10:17 AM
Propagation state  : IN-SYNC
-----
```

In addition to the information provided by the `list_coexistence_systems` command, you can use the `beectl list_coexistence_connectors` command to show information about individual coexistence connectors within a Remote Coexistence System:

```
beectl> list_coexistence_connectors
```

List of coexistence connectors for remote collaboration systems.

```
Connector Name      : Oracle-coexConnExch (tmcoex1)
Routing List        : RGC426AE:*;1;OCS:*;1;
Relay Address Type   : RGC426AE
-----
```

Modifying Oracle Beehive Remote Coexistence Systems

Oracle Beehive administrators may need to modify certain properties of an Oracle Beehive Remote Coexistence System. This section includes information about modifying a Remote Coexistence System using the `modify_coexistence_system` command. Depending on what properties require modification, different options are required.

In the following example, the Remote Coexistence System listed in the "[Listing Oracle Beehive Remote Coexistence Systems](#)" section, named `TMCOEX2`, is being assigned a new URL:

```
beectl> modify_coexistence_system --select_by_name TMCOEX2 --url
http://new.example.com
```

You can also update the Remote Coexistence System's `sitekey` attribute.

See Also: For more information about the `modify_coexistence_system` command, including available options and syntax, see "modify_coexistence_system" in Module 2 of the *Oracle Beehive Administrator's Reference Guide*.

Provisioning Users for Coexistence

Before users can take advantage of the Oracle Collaboration Coexistence Gateway, Oracle Beehive administrators must provision users for coexistence with a Remote Coexistence System. This section includes information about provisioning users for coexistence with a Remote Coexistence System using the `modify_coexistence_profile` command.

Prerequisites

Before provisioning users for coexistence with a Remote Coexistence System using the `modify_coexistence_profile` command the following conditions must be met:

- The user account must exist on Oracle Beehive. For information about provisioning users on Oracle Beehive, refer to ["Provisioning User Accounts Using beectl"](#) in the ["Configuring and Managing Oracle Collaboration Coexistence Gateway"](#) module.
- Oracle recommends that Oracle Beehive and Microsoft Exchange users have the same primary e-mail address. Ensure that the every user's Microsoft Active Directory SMTP address property is set to the Oracle Beehive e-mail address. This address should be set as primary.

Example 15–3 Provisioning a User for Coexistence

In this example, `user2` is being provisioned for coexistence with Remote Coexistence System `TMCOEX2`.

Using the `BEEHIVE` argument for the `--accessible_system` option indicates an Oracle Beehive user who will be coexisting:

```
beectl> modify_coexistence_profile --user user=user2 --accessible_system BEEHIVE
```

Coexistence profile has been created and processing is initiated.

Using the `EXCHANGE` argument for the `--accessible_system` option indicates that the user is a Coexisting Third Party User (the user can access Microsoft Exchange only):

```
beectl> modify_coexistence_profile --user user=user2 --accessible_system BEEHIVE
--accessible_system EXCHANGE
```

See Also: For more information about the `modify_coexistence_profile` command, including available options and syntax, see ["modify_coexistence_profile"](#) in Module 2 of the *Oracle Beehive Administrator's Reference Guide*.

Post-Provisioning Notes

After running the `modify_coexistence_profile` command, the registration process will begin. To view whether the registration process has completed refer to ["Viewing the Registration State of Coexisting Users"](#).

Manually Importing User Data

To avoid or delay the registration process of coexistence users (which normally occurs automatically), you can run the `beectl modify_coexistence_profile` command with the `--no_data_import` option. You can then use the `beectl import_coexistence_data` command to manually trigger user data import.

You might wish to do this for any of the following reasons:

- You want to customize date ranges and data types for a subset of users
- You want to trigger the import at a later time, and not immediately when users are registered for coexistence
- If the automated data import failed or partially succeeded, you can reattempt the import manually
- You want to extend the date ranges to get more data in the past

To register a coexistence user without importing that user's data into Oracle Beehive, use the `--no_data_import` option, such as in the following example:

```
beectl> modify_coexistence_profile --user user=<user_login_id> --accessible_system
BEEHIVE --no_data_import
```

To manually import coexistence user data into Oracle Beehive, use the `beectl import_coexistence_data` command:

```
beectl> import_coexistence_data [--user <user_cen> | --email <user_email> ]
[--data_type <type>] [--timerange_start <start>] [--relative_timerange_start
<relative_start>] [--timerange_end <end>] [--relative_timerange_end <relative_
end>]
```

Use the `--data_type` option to specify one or more particular types of user data to import. Valid types are MEETING, TASK, CONTACT or MESSAGE.

Use the `--timerange_start` and `--timerange_end` options to specify an absolute range of time for which all data should be imported. You can use a timestamp, for example 2007-10-01T12:00:00Z. You can also use the keywords NOW (to specify a timerange beginning or ending with the current time) or INFINITY (to specify that there should be no limit).

Use the `--relative_timerange_start` and `--relative_timerange_end` options to specify a relative range of time (in days) for which all data should be imported. Specify an integer for each option. A negative integer represents a number of days prior to today, while a positive integer represents a number of days in the future (from today).

See Also: For more information about the `import_coexistence_data` command, including available options and syntax, see "import_coexistence_data" in Module 2 of the *Oracle Beehive Administrator's Reference Guide*.

Viewing the Registration State of Coexisting Users

After a user has been provisioned for coexistence using the steps outlined in ["Provisioning Users for Coexistence"](#), Oracle Beehive will attempt to register the user. The user must be registered for coexistence before being able to coexist. Although this will occur automatically, it may take some time.

To view the registration state of coexisting users use the `list_coexistence_profiles` command:

```
beectl> list_coexistence_profiles
```

Output similar to the following will be returned:

```
-----+-----+-----+-----
User Name      | Accessible systems | State      | Import state
-----+-----+-----+-----
```

user1	BEEHIVE	COMPLETED	NONE
user2	BEEHIVE, EXCHANGE	COMPLETED	CREATED
user3	EXCHANGE	COMPLETED	COMPLETED

Five possible values will appear in the `State` column:

- `CREATED` indicates that the registration request has been created.
- `IN-PROGRESS` indicates that the registration request has been sent to the Oracle Beehive Remote Coexistence System, but the Oracle Beehive Coexistence service is waiting for a response.
- `COMPLETED` indicates that the registration request acknowledgement has been received from the Oracle Beehive Remote Coexistence System and the provisioning, and registration process has completed.
- `DELETED` indicates that the user was once provisioned for coexistence, but is no longer.
- `REJECTED` indicates that the specified system(s) were not possible for a given user. For example, if you specify `EXCHANGE` as one of the systems for a user that does not exist in the Microsoft Exchange server, the registration process will fail and the profile will be set to `REJECTED`.

The value in the `State` column must be `COMPLETED` before a user has been fully provisioned for coexistence.

To view the additional information about the registration state of coexisting users, use the `list_coexistence_profiles` command with the `--show_more` option:

```
beectl> list_coexistence_profiles --show_more
```

The following columns will be displayed:

- User Name
- Accessible systems
- State
- Last Modification
- Import State
- Last import

To view complete information about coexisting users, use the `list_coexistence_profiles` command with the `--show_all` option:

```
beectl> list_coexistence_profiles --show_all
```

Output similar to the following will be displayed for each user:

```
=====
User:user2
Accessible system: BEEHIVE, EXCHANGE
Profile process state: COMPLETED
Profile modifiedon: 8/5/08 7:34:07 AM

Data Type: CONTACT
GUID: guid-F14819DF1B5F4DA683E1C9393DFF106E000000000003
State: CREATED
```

```
Latest process time: NOT-PROCESSED
Received entity: 0
Expected entity: 0
Time range start: -INFINITY
Time range end: INFINITY

Data Type: MESSAGE
GUID: guid-F14819DF1B5F4DA683E1C9393DFF106E000000000002
State: CREATED
Latest process time: NOT-PROCESSED
Received entity: 0
Expected entity: 0
Time range start: 7/6/08 2:28:08 PM UTC
Time range end: INFINITY

Data Type: TASK
GUID: guid-F14819DF1B5F4DA683E1C9393DFF106E000000000001
State: CREATED
Latest process time: NOT-PROCESSED
Received entity: 0
Expected entity: 0
Time range start: 7/6/08 2:28:08 PM UTC
Time range end: INFINITY

Data Type: MEETING
GUID: guid-F14819DF1B5F4DA683E1C9393DFF106E000000000000
State: CREATED
Latest process time: NOT-PROCESSED
Received entity: 0
Expected entity: 0
Time range start: 7/6/08 2:28:06 PM UTC
Time range end: INFINITY
=====
```

Deprovisioning Users from an Oracle Beehive Remote Coexistence System

If a user has been provisioned for coexistence, and coexistence is no longer needed thereafter, the user can be deprovisioned. This section includes information about provisioning users for coexistence with a Remote Coexistence System using the `delete_coexistence_profile` command.

Example 15–4 *Deprovisioning a User for Coexistence*

In this example, `user2` is being deprovisioned for coexistence from Remote Coexistence System `TMCOEX2`:

```
beectl> delete_coexistence_profile --user user=user2
```

Coexistence profile has been deleted and processing is started.

See Also: For more information about the `delete_coexistence_profile` command, including available options and syntax, see "delete_coexistence_profile" in Module 2 of the *Oracle Beehive Administrator's Reference Guide*.

Post-Deprovisioning Notes

After running the `delete_coexistence_profile` command, the deprovisioning process will begin. To view the state of the coexistence profile refer to ["Viewing the Registration State of Coexisting Users"](#).

Oracle Connector for Exchange Administrative Tasks

This section explains how to stop and start Oracle Connector for Exchange. The following topics are covered in this section:

- [Best Practices for Stopping and Starting Oracle Connector for Exchange](#)
- [Stopping Oracle Connector for Exchange](#)
- [Starting Oracle Connector for Exchange](#)
- [Stopping the BEECONNECTOR OC4J Instance](#)
- [Starting the BEECONNECTOR OC4J Instance](#)
- [Configuring Oracle Connector for Exchange to use HTTPS](#)

When installed in the default location, the command line tool to stop and start Oracle Connector for Exchange is located in the

`C:\oracle\product\<version>\exconnector_1\beehive\collabcoex_connector` folder. This section assumes that you have navigated to the above mentioned directory from the Windows command line prompt.

Best Practices for Stopping and Starting Oracle Connector for Exchange

Oracle Connector for Exchange works in tandem with the BEECONNECTOR OC4J instance. Both are installed on the computer where Oracle Connector for Exchange was deployed.

Due to the interdependency between Oracle Connector for Exchange and the BEECONNECTOR OC4J instance, Oracle recommends stopping and starting the service and instance in a specific order.

Stopping When stopping the Oracle Connector for Exchange, Oracle recommends following these steps:

1. Stop the BEECONNECTOR OC4J instance. For details about stopping the BEECONNECTOR OC4J instance, see ["Stopping the BEECONNECTOR OC4J Instance"](#).
2. Stop Oracle Connector for Exchange. For details about stopping Oracle Connector for Exchange, see ["Stopping Oracle Connector for Exchange"](#).

Starting When starting the Oracle Connector for Exchange, Oracle recommends following these steps:

1. Start Oracle Connector for Exchange. For details about starting Oracle Connector for Exchange, see ["Starting Oracle Connector for Exchange"](#).
2. Start the BEECONNECTOR OC4J instance. For details about starting the BEECONNECTOR OC4J instance, see ["Starting the BEECONNECTOR OC4J Instance"](#).

Stopping Oracle Connector for Exchange

Before stopping the Oracle Connector for Exchange, the BEECONNECTOR OC4J instance must be stopped. For details about stopping the BEECONNECTOR OC4J instance, see ["Stopping the BEECONNECTOR OC4J Instance"](#).

To stop Oracle Connector for Exchange, run the following command:

```
coexctl.exe stop_connector
```

Note: Oracle Connector for Exchange can also be stopped in the Windows Services panel, by selecting and stopping "Oracle Coexistence Connector for Exchange" and "Oracle Coexistence Administration Service".

Starting Oracle Connector for Exchange

To start Oracle Connector for Exchange, run the following command:

```
coexctl.exe start_connector
```

Note: Oracle Connector for Exchange can also be started in the Windows Services panel, by selecting and starting "Oracle Coexistence Connector for Exchange" and "Oracle Coexistence Administration Service".

Stopping the BEECONNECTOR OC4J Instance

The BEECONNECTOR OC4J instance works in tandem with the Oracle Connector for Exchange. It is installed on the same computer as Oracle Connector for Exchange, and can be stopped using the following instructions:

1. Click the **Start** button on the Windows computer hosting Oracle Connector for Exchange
2. Select **Programs**
3. Select **Oracle - coex**
4. Click on **Stop SOA suite**

Starting the BEECONNECTOR OC4J Instance

The BEECONNECTOR OC4J instance works in tandem with the Oracle Connector for Exchange. The BEECONNECTOR OC4J instance is installed on the same computer as Oracle Connector for Exchange, and can be started using the following instructions:

Note: The Oracle Connector for Exchange should be started before starting the BEECONNECTOR OC4J instance. For details about starting Oracle Connector for Exchange, see ["Starting Oracle Connector for Exchange"](#).

1. Click the **Start** button on the Windows computer hosting Oracle Connector for Exchange
2. Select **Programs**
3. Select **Oracle - coex**

4. Click on **Start SOA suite**

Configuring Oracle Connector for Exchange to use HTTPS

If Oracle Beehive was configured to use HTTPS, the Oracle Connector for Exchange must be configured to use the Oracle Beehive HTTPS URL.

Follow these steps to configure Oracle Connector for Exchange to use the Oracle Beehive HTTPS URL:

1. Stop the BEECONNECTOR OC4J Instance. For information about stopping the BEECONNECTOR OC4J Instance, refer to [Stopping the BEECONNECTOR OC4J Instance](#).
2. Using the command prompt on the computer hosting Oracle Connector for Exchange, navigate to the C:\oracle\product\<version>\exconnector_1\beehive\collabcoex_connector directory.

Where <version> represents the installed version of Oracle Connector for Exchange.

3. Run the following command:

```
coexctl.exe set_property --file C:\oracle\product\<version>\beehive\collabcoex_connector\conf\OGWEConfiguration.xml --section communicationservice --property endpoint --value "https://<OracleBeehiveHost>:<OracleBeehivePort>/coexService/CoexMLPort"
```

Where:

- <OracleBeehiveHost> represents your Oracle Beehive hostname
 - <OracleBeehivePort> represents the HTTPS listening port of your Oracle Beehive installation
4. Start the BEECONNECTOR OC4J Instance. For information about starting the BEECONNECTOR OC4J Instance, refer to [Starting the BEECONNECTOR OC4J Instance](#).

Oracle Change Notification Service for Exchange Administrative Tasks

This section explains how to stop and start the Oracle Change Notification Service for Exchange from the command line.

When installed in the default location, the command line tool to stop and start the Oracle Change Notification Service for Exchange is located in the c:\oracle\product\<version>\exconnector_1\beehive\collabcoex_connector folder. This section assumes that you have navigated to the above mentioned directory from the Windows command line prompt.

Stopping Oracle Change Notification Service for Exchange

To stop the Oracle Change Notification Service for Exchange, run the following command:

```
coexctl.exe stop_eventsink
```

Note: Oracle Change Notification Service for Exchange can also be stopped in the Windows Services panel, by selecting and stopping "Oracle Change Notification Service for Exchange".

Starting Oracle Change Notification Service for Exchange

To start the Oracle Change Notification Service for Exchange, run the following command:

```
coexctl.exe start_eventsink
```

Note: Oracle Change Notification Service for Exchange can also be started in the Windows Services panel, by selecting and starting "Oracle Change Notification Service for Exchange".

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](#)

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 16-2

- Whenever you have already scheduled system downtime, you should use the opportunity to perform a full closed (cold) backup. See ["Performing a Closed \(Cold\) Backup of Oracle Beehive"](#) on page 16-4
- 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 16-5

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](#)
- [Performing a Closed \(Cold\) Backup of Oracle Beehive](#)
- [Performing an Online \(Hot\) Backup of Oracle Beehive Database](#)

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

Shut down 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/bee hive/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:

- Closed (cold) backup

A closed 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 a Closed \(Cold\) 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. Oracle Beehive requires the database to be in ARCHIVELOG mode., so an online backup is a viable option.

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 a Closed (Cold) Backup of Oracle Beehive

It is recommended to take a full closed (cold) backup of all mid-tiers before going into production. From that point onwards, a full closed backup should be taken after every Oracle Beehive patch is applied to the system. Also a full closed backup is recommended every time a change impacting connectivity to configuration data like change to database connect string, schema passwords etc. is implemented as this has the potential to hamper usability of older backups. Apart from that, if there are any scheduled outage windows, then a full closed backup of all mid-tiers should be performed too. Use archival tool of your choice like tar, cpio, zip to take the backup of the entire mid-tier ORACLE_HOME and the oraInventory.

Here is a very simple example demonstrating a closed backup of 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 SQLplus
4. Backup 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%'
        ( 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 a Closed \(Cold\) 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 a Closed (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.

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

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 system in the event of a failure in the primary system.

This module contains the following topics:

- [Prerequisites](#)
- [Configuring Oracle Beehive for Disaster Recovery](#)
- [Performing Role Transitions](#)
- [Tuning Options](#)

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, as your primary database.

Configuring Oracle Beehive for Disaster Recovery

These configuration steps show how to take advantage of service relocation 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.

Database-Only versus Full Deployment

In an Oracle Beehive 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.

Example Deployments

Figure [Figure 17-1](#) 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 17-1 Oracle Beehive Full-Deployment Disaster Recovery Scenario

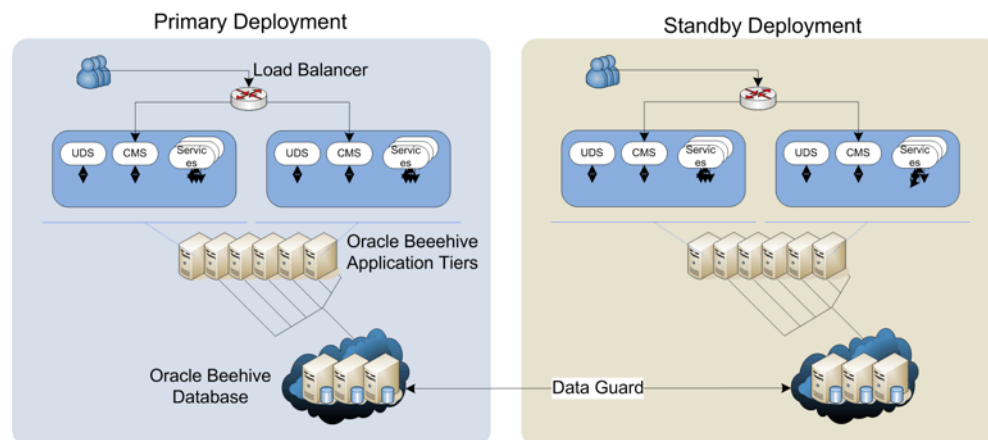
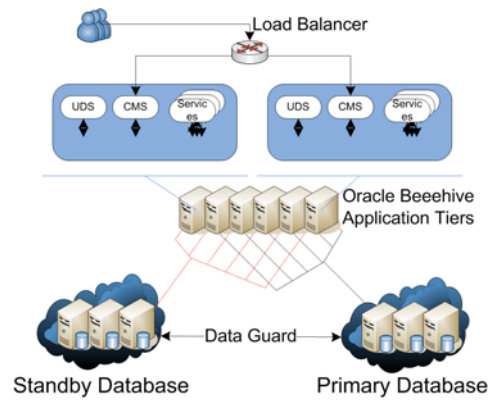


Figure [Figure 17-2](#) shows a typical database-only deployment. In this scenario, there is a standby database that duplicates the primary multi-node database, but there are no standby Application tiers. The primary Application tiers remain active and connect to the standby database. This example provides database redundancy and failover in the event of a failure in the primary database, but there is no provision for Application-tier failover.

Figure 17-2 Oracle Beehive Database-Only Recovery Scenario

To configure Oracle Beehive for use in these scenarios, perform the following procedures:

- [Recording Environment Details](#)
- [Configuring Service Relocation and Database Triggers](#)
- [Configuring Oracle Beehive Application Tiers](#)

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_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 10g 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_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. [Create Database Services Startup Trigger](#)
3. [Setup FAN ONS Publisher and Database Role Change Trigger](#)

Create Managed Services

Perform the following steps to create managed services:

1. If you are using a RAC database deployment, add a CRS-managed bootstrap service for the primary and standby databases by running the following commands:

For the primary, run the following command on any primary RAC node:

```
srvctl add service -s beehivedg -d pr_orcl1 -r pr_orcl1,pr_orcl2
```

For the standby, run the following command on any standby RAC node:

```
srvctl add service -s beehivedg -d stby_orcl1 -r stby_orcl1,stby_orcl2
```

Note: This service name must be different than the name used in the database parameter `SERVICE_NAMES`. In this example, `beehivedg` is the service name.

2. If you are using a single-instance database deployment, create the relocation 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;
/
```

Create Database Services Startup Trigger

When a switchover or failover to a standby database occurs, service relocation and affinity services or other CRS-managed services are not automatically started. Perform the following steps to create a trigger to start up database-related services. (In a full deployment scenario, you do not need to perform this procedure.)

1. If you have a multiple instance deployment, create a script to start CRS services. [Example 17-1](#) shows an example script, which includes using affinity services, for a multiple instance deployment. [Example 17-2](#) shows a similar script which does not include using affinity services:

Note: Instructions for setting up affinity services are included in the RAC post-install steps in the *Oracle Beehive Installation Guide* for your platform.

Example 17-1 Example CRS-Managed Service Startup Script With Affinity Services *StartupService.sh*

```
#!/bin/ksh
export ORACLE_BASE=/u01/app/oracle
export ORACLE_HOME=$ORACLE_BASE/product/11.1.0/db
export LD_LIBRARY_PATH=$ORACLE_HOME/lib:$ORACLE_HOME/ctx/lib
export PATH=$ORACLE_HOME/bin:$ORACLE_HOME/opmn/bin:/usr/local/bin:/bin:/usr/bin
export TIMESTAMP=`date '+%y%m%d%H%M%s'`
export LOGFILE=/tmp/StartupService_${TIMESTAMP}.log

echo "Starting $0 at `date`" > $LOGFILE
HOSTNAME=`hostname | awk -F. '{print $1}'`
case $HOSTNAME in
    pr_db1)
        export DB_UNIQUE_NAME=bhha
        export ORACLE_SID=bhha1
        export AFF_SVC=aff1
        ;;
    pr_db2)
        export DB_UNIQUE_NAME=bhha
        export ORACLE_SID=bhha2
        export AFF_SVC=aff2
        ;;
    stby_db1)
        export DB_UNIQUE_NAME=bhha
        export ORACLE_SID=bhha1
        export AFF_SVC=aff1
        ;;
    stby_db2)
        export DB_UNIQUE_NAME=bhha
        export ORACLE_SID=bhha2
        export AFF_SVC=aff2
        ;;
    *) echo "Node $HOSTNAME not identified or not defined for database bhha " >>
    $LOGFILE
    exit ;;
esac

echo "`date` : Starting CRS managed database service $AFF_SVC" >> $LOGFILE
srvctl start service -d $DB_UNIQUE_NAME -s $AFF_SVC -i $ORACLE_SID >> $LOGFILE
2>&1
srvctl start service -d $DB_UNIQUE_NAME -s beehivedg -i $ORACLE_SID >> $LOGFILE
```

```
2>&1
echo "`date` : Start of CRS managed database service $AFF_SVC complete" >>
$LOGFILE
```

Example 17–2 Example CRS-Managed Service Startup Script Without Affinity Services StartupService.sh

```
#!/bin/ksh
export ORACLE_BASE=/u01/app/oracle
export ORACLE_HOME=$ORACLE_BASE/product/11.1.0/db
export LD_LIBRARY_PATH=$ORACLE_HOME/lib:$ORACLE_HOME/ctx/lib
export PATH=$ORACLE_HOME/bin:$ORACLE_HOME/opmn/bin:/usr/local/bin:/bin:/usr/bin
export TIMESTAMP=`date '+%y%m%d%H%M%S'`
export LOGFILE=/tmp/StartupService_${TIMESTAMP}.log

echo "Starting $0 at `date`" > $LOGFILE
HOSTNAME=`hostname | awk -F. '{print $1}'`

echo "`date` : Starting beehivedg database service" >> $LOGFILE
srvctl start service -d $DB_UNIQUE_NAME -s beehivedg -i $ORACLE_SID >> $LOGFILE
2>&1
echo "`date` : Start of beehivedg database service complete" >> $LOGFILE
```

2. Create a database startup trigger on the primary database (Oracle Data Guard will subsequently replicate your trigger on the standby database). For example, for a multiple instance deployment:

```
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_scheduler.create_job(
            job_name=>'start_dbsvc',
            job_type=>'executable',
            job_action=>'<ORACLE_HOME>/bin/StartupService.sh',
            enabled=>TRUE
        );
    END IF;
END;
/
```

job_action should specify the location of the script you created in Step 1

Replace <ORACLE_HOME> with the full path of your Oracle Database Oracle home.

For a single instance deployment (in which you do not need a script):

```
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;
```

```
END;
/
```

3. To test the script, 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, there is no switchover or failover between database deployments, so you can skip this step.)

See Also: Pages 11-13 of "Client Failover in Data Guard Configurations for Highly Available Oracle Databases: Oracle Database 10g 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):

```
bhha peer=stby_orcl
bhha peer=pr_orcl
bhha service=beehivedg location=pr_db1,pr_orcl1:pr_db2,pr_orcl2
bhha service=aff1 location=pr_db1,pr_orcl1
bhha service=aff2 location=pr_db2,pr_orcl2
bhha service=beehivedg location=stby_db1,stby_orcl1:stby_db2,stby_orcl2
bhha service=aff1 location=stby_db1,stby_orcl1
bhha 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. 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

In a database-only deployment, configure your Oracle Beehive servers by performing the following configuration steps:

Note: You only need to perform these steps on one Application tier.

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 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
```

3. Activate the Configuration on the first Beehive Application tier by running the following commands:

```
beectl activate_configuration  
beectl modify_local_configuration_files  
beectl start --all
```

4. 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.

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
2. Follow the steps in MetaLink Note 751600.1
3. Start the Application tiers in your standby deployment
4. Once the standby Oracle Beehive system is up and running, you can redirect user traffic to the standby deployment

Performing a Failover

To perform a database-only switchover 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 switchover operation in a full-deployment scenario, perform the following steps:

1. Stop Oracle Beehive on the primary site application nodes
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. Start the Application tiers in your standby deployment
4. Once the standby Oracle Beehive system is up and running, you can redirect user traffic to the standby 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 10g 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 "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 17-3](#), 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 17–3 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 <X.X.X.X>.

2. Implement the listener.ora changes by running the following command:

```
lsnrctl reload
```

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](#)
- [ChangingLog Levels](#)
- [Searching Through Logs](#)
- [Managing Logs](#)

Logging Architecture

Oracle Beehive logs are located in the following directory:

`$ORACLE_HOME/bee hive/logs`

A variety of directories and files will appear within the `$ORACLE_HOME/bee hive/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/bee hive/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.

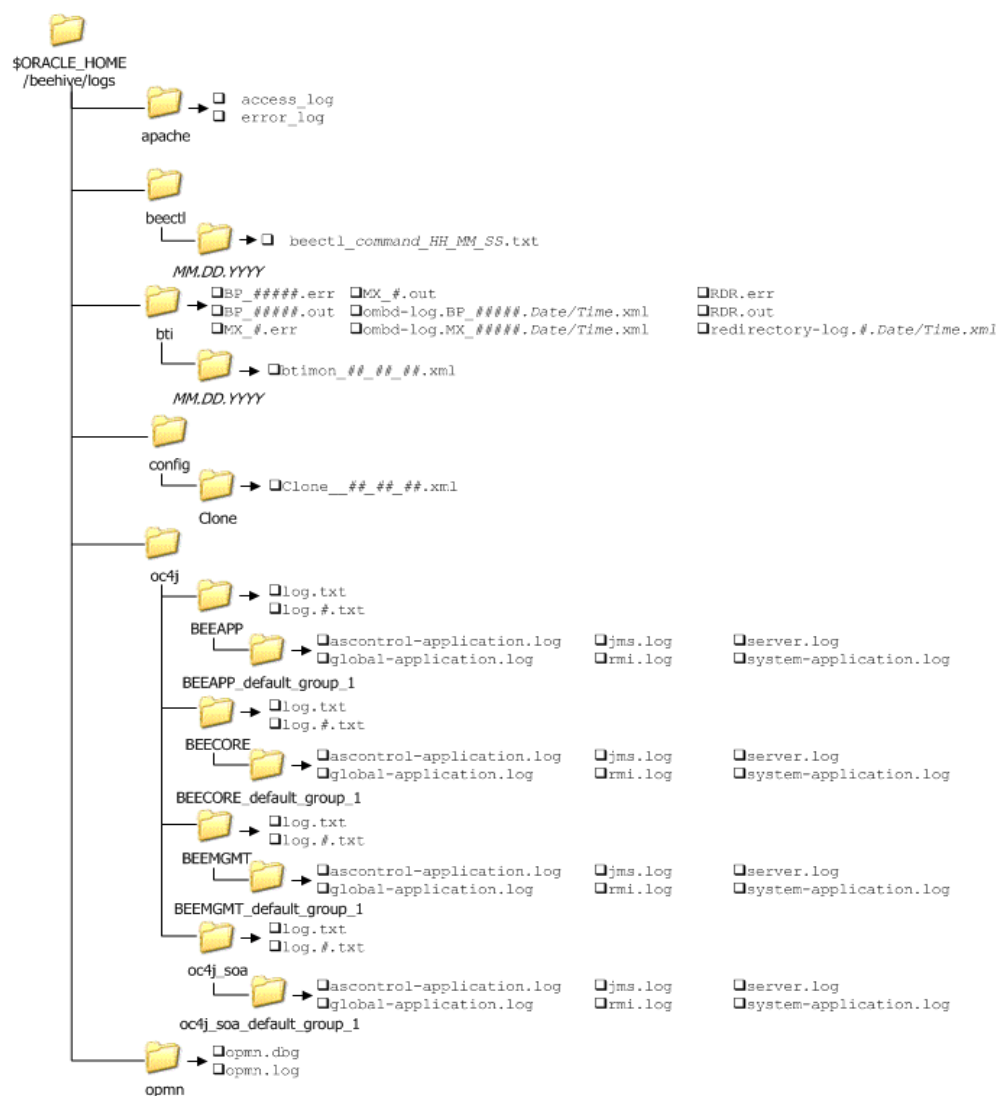
Within the `$ORACLE_HOME/bee hive/logs/oc4j` directory, sub-directories are named after managed components and will typically include a `log.txt` file and a series of files named `log.txt.#`. The `#` in `log.txt.#` represents an archived version of the log file. The most recent log file is always `log.txt`.

For more information about log archiving options refer to the ["Managing Logs"](#) section.

Figure 18–1 outlines what you might typically expect to see in the `$ORACLE_HOME/bee hive/logs` directory.

Note: The directory structure on your Oracle Beehive deployment may have additional or fewer files and directories.

Figure 18–1 Oracle Beehive Log Directory Structure



Note: The `$ORACLE_HOME/bee hive/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 18–1 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 18–1 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 18–2](#) 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 18–3](#) explains the sequence of errors that appear in [Example 18–2](#).

Table 18–2 Explanation of Error Message Fields in [Example 18–2](#)

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 18–2 (Cont.) Explanation of Error Message Fields in [Example 18–2](#)

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 18–1, "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 <i>Oracle Beehive Administrator's Reference Guide</i>

Example 18–2 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_compuid: 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 18–2](#) 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 18–3](#) explains the sequence of errors that appear in [Example 18–2](#).

Table 18–3 Explanation of Error Message Fields in [Example 18–2](#)

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 18–1, "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 18–3 (Cont.) Explanation of Error Message Fields in Example 18–2

Number	Name	Description
9	Description	<p>A description of the error message. This message will often include Oracle Beehive error code IDs, suspected causes and recommended actions.</p> <p>For a list of Oracle Beehive error code IDs, refer to "Oracle Beehive Error Codes" in <i>Oracle Beehive Administrator's Reference Guide</i></p>

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 18–4 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 18–4 Log Level Values Set vs. Values that Appear in the Oracle Beehive Logs

Value Used to Set Using <code>beectl</code>	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 18-4](#).

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/bee hive/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 18-4](#).

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/bee hive/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/bee hive/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 18–3 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 18–3](#) 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 18–4 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 18–4](#) 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/bee hive/logs/oc4j/BEEAPP directory. The `--display_source` option instructs the command to return the file in which the record appears.

Example 18–5 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 18–5](#) 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*.

Index

A

- access control, 13-1 to 13-13
 - about, 13-1
 - explicit access control, 13-1
 - ACEs, 13-9
 - creating and managing, 13-9
 - assigned roles, 13-8
 - implicit, 13-2
 - privileges, 13-2
 - default, 13-3
 - managing, 13-3
 - roles, 13-3, 13-4
 - deleting, 13-9
 - modifying, 13-8
 - role definitions, 13-5
 - sensitivities, 13-9
 - about, 13-2
 - creating and managing, 13-10
 - team workspaces, 6-20
- access control entities (ACEs), 13-9
 - creating and managing, 13-9
- Access Control Service
 - managing, 5-3
- ACEs (access control entities), 13-9
 - creating and managing, 13-9
- adding, 15-12
- administration, overview of, 1-1
- Alarm Service, 2-4, 5-14
- attachment blocking (e-mail), 8-15
- Audit Service, 2-4
 - managing, 5-4
- auditing
 - policies, 12-16
 - example, 14-9
- Authentication Service
 - managing, 5-4
- Authentication Services, 2-4
- auto attendant
 - administration commands, 11-28
 - configuring, 11-26
 - installing, 11-26
- auto attendants, 11-2

B

- backing up, 16-2
 - cold backup, 16-4
 - hot backup, 16-5
- backup and recovery, 16-1 to 16-8
 - backups, 16-2
 - cold backup, 16-4
 - hot backup, 16-5
 - introduction, 16-1
 - recovering, 16-6
- BEEadmin account
 - about, 3-18
- BEEAPP, 2-4
- BEECONNECTOR OC4J Instance
 - starting, 15-20
 - stopping, 15-20
- BEECORE, 2-4
- beectl
 - shell, 2-2
- beectl commands
 - activate_configuration, 7-5, 7-9, 7-11, 7-13, 7-14
 - add_category, 6-17
 - add_category_application, 6-19
 - add_coexistence_connector, 15-12
 - add_custom_user_property, 3-28
 - add_filesystem_reference, 6-24
 - add_group, 3-36
 - add_resource, 4-3
 - add_team_workspace, 6-14
 - add_user, 3-19, 3-21
 - example, 3-20
 - add_workspace_template, 6-13
 - delete_category, 6-18
 - delete_category_application, 6-19
 - delete_coexistence_profile, 15-18
 - delete_custom_user_property, 3-28
 - delete_filesystem_reference, 6-25
 - delete_resources, 4-7
 - delete_team_workspace, 6-16
 - delete_user, 3-27, 3-30
 - delete_workspace_template, 6-13
 - download_syncml_messages, 7-5
 - import_documents, 6-24
 - list_coexistence_connectors, 15-14
 - list_coexistence_systems, 15-13

- list_commands, 3-3
- list_components, 7-7, 7-12
- list_custom_user_properties, 3-28
- list_enterprises, 3-19
- list_filesystem_references, 6-25
- list_groups, 3-34
- list_organizations, 3-20
- list_properties, 7-8, 7-12
- list_resources, 4-4
- list_role_definitions, 6-19
- list_users, 3-13
- list_workspace_templates, 6-9, 6-12
- list_workspaces, 6-15
- modify_coexistence_connector, 15-14
- modify_coexistence_profile, 15-15
- modify_group, 3-36
- modify_personal_workspace, 6-9
- modify_property, 7-5, 7-9, 7-11, 7-13, 7-14
- modify_resources, 4-6, 4-7, 4-8
- modify_team_workspace, 6-16, 6-20
- modify_user, 3-25, 3-28, 3-31
- modify_workspace_template, 6-13
- restart, 2-2, 2-5
- start, 2-2, 2-4
- status, 2-3
- stop, 2-2, 2-5
- upload_client_applications, 7-7
- upload_device_profiles, 7-2
- using, 1-1
- BEEMGMT, 2-4
- best practices, 15-19
- BTI, 2-4

C

- CalDAV Service, 5-15
- categories
 - managing, 6-17
- Cisco hardware requirements, 11-3
- Client Management Service, 5-5
- coexctl, 15-20, 15-21
- coexistence
 - e-mail, 8-21
- Coexistence Connector
 - configuring, 15-12
 - listing, 15-13
 - modifying, 15-14
- Coexistence Service, 2-4, 5-21
- collaborative services
 - content management services
 - FTP Service, 5-11
 - WebDAV Service, 5-11
 - instant message services
 - Instant Message Service, 5-13
 - XMPP Service, 5-13
 - meetings services
 - transcoding service, 5-14
 - time management services
 - Alarm Service, 5-14
 - CalDAV Service, 5-15

- Resource Directory Service, 5-15
- Time Management Service, 5-15
- Time Zone Service, 5-15
- Zimbra services, 5-16
 - Zimbra Connector Service, 5-16
 - Zimbra UI Service, 5-16
- Conference Service, 2-4
- content management services, 5-11
 - FTP Service, 5-11
 - WebDAV Service, 5-11
- core services
 - event services
 - Event Service, 5-7
 - Object Event Publisher Service, 5-7
 - Policy Service, 5-7
 - User Directory Service, 5-8
 - Workflow Service, 5-10
- custom user properties, 3-28

D

- dead letter (e-mail), 8-17
- default provisioning policy
 - about, 3-3
- Device Management Service, 2-4, 5-5
- Discussions Service, 2-4, 5-11
- dispatch rules (e-mail), 8-19
 - about, 8-3
- dynamic groups, 3-34

E

- e-mail
 - attachment blocking, 8-15
 - coexistence, 8-21
 - configuring
 - about, 8-1
 - dead letter, 8-17
 - dispatch rules, 8-19
 - about, 8-3
 - endpoints
 - about, 8-3
 - setting and modifying, 8-18
 - introduction, 8-1
 - local users domain, 8-20
 - managing, 8-1 to 8-22
 - managing components of, 8-4
 - non-Beehive domains, 8-19
 - open relay (preventing), 8-6
 - relay configuration
 - VMS evaluators, 8-5
 - relay servers, 8-12
 - SMTP Server
 - about, 8-2
 - virtual mail servers
 - about, 8-2
 - virus scanning, 8-14, 8-15
 - whitelist, 8-10
- E-mail Service, 2-4, 5-12
- endpoints

- about, 8-3
- endpoints (e-mail)
 - setting and modifying, 8-18
- enterprise preference properties
 - voicemail, 11-3
- enterprise preferences
 - voicemail, 11-11
- enterprise services
 - mobility services
 - Mobile Data Synchronization service, 5-18
 - Mobile Device Management Service, 5-17
 - Mobile Mail Service, 5-18
 - Mobile Push Service, 5-18
 - subscription and notification services
 - Notification Delivery Service, 5-19
 - SMPP Delivery Service, 5-20
 - Subscription Service, 5-19
- enterprise users
 - about, 3-4
- error code severities, 18-3
- error message
 - examples, 18-3
- Event Service, 5-7
- Event Services, 2-4
- event services, 5-6
 - Event Service, 5-7
 - Object Event Publisher Service, 5-7
- events, 12-1 to 12-6
 - asynchronous events, 12-2
 - event subscriptions, 12-2
 - introduction, 12-1, 12-2
 - managing, 12-5
 - synchronous events, 12-2
 - workspace events, 6-5
- extended enterprise users
 - about, 3-4
- external contacts
 - about, 3-4

F

- facilities, 11-1, 11-12
 - creating, 11-12
 - sample XML, 11-13
- facility properties, 11-3
- fax message service, 5-12
- file system directories, 6-23
- files
 - FTP access, 6-25
 - managing in workspaces, 6-23
 - WebDAV access, 6-25
- FTP
 - access to workspaces, 6-25
- FTP Service, 2-4, 5-11

G

- getting started with Oracle Beehive, 1-1
- groups
 - about, 3-32

- creating, 3-35
- dynamic, 3-34
- inheritance, about, 3-41
- listing, 3-34
- managing, 3-32
- modifying, 3-35
- static, 3-34

H

- HTTPS configuration, 15-21

I

- Identity Provider Service
 - managing, 5-5
- Instant Message Service, 5-13
- Instant Message Services, 2-4
- instant message services, 5-13
 - Instant Message Service, 5-13
 - XMPP Service, 5-13

L

- log entries
 - understanding, 18-3
- log levels, 18-5
 - changing, 18-6 to 18-7
- logging
 - architecture, 18-1
- logging properties, listing, 18-5
- logs
 - archiving, 18-7
 - managing, 18-1 to 18-8
 - searching, 18-8

M

- managed components
 - determining the identifier, 2-3
 - services, 2-4
- Management Service, 5-7
- managing users
 - using beectl
 - about, 3-3
- meetings services, 5-13
 - transcoding service, 5-14
- Message Delivery Service, 2-4
- Mobile Data Sync Service
 - controlling MD5 authentication, 7-10
 - controlling synchronization ranges, 7-11
 - controlling synchronized data types, 7-9
 - listing configurable properties, 7-7
- Mobile Data Synchronization service, 5-18
- Mobile Device Management Service, 2-4, 5-17
- mobile device profile
 - adding a new device to a, 7-4
- mobile device profiles
 - customizing, 7-2
 - uploading, 7-2
- Mobile Mail Service, 2-4, 5-18

- listing configurable properties, 7-12
- maximum message size, 7-14
- number of e-mails pushed to a device, 7-13
- past e-mail push, 7-14
- Mobile Push Service, 2-4, 5-18
- Mobility Services
 - managing, 7-1 to 7-14
- mobility services, 5-17
 - Mobile Data Synchronization service, 5-18
 - Mobile Device Management Service, 5-17
 - Mobile Mail Service, 5-18
 - Mobile Push Service, 5-18
- MWI (Message Waiting Indicator), 11-22

N

- Notification Delivery Service, 5-19
- Notification Service, 2-4

O

- Object Event Publisher Service, 5-7
- oc4j_soa, 2-4
- OHS, 2-4
- open relay, 8-6
- Oracle Beehive
 - starting, 2-1 to ??
- Oracle Beehive Transport Infrastructure, 2-4
- Oracle Change Notification Service for Exchange
 - administrative tasks, 15-21
 - starting, 15-22
 - stopping, 15-21
- Oracle Collaboration Coexistence Gateway
 - Active Directory Contact, 15-2
 - Active Directory Exchange User, 15-2
 - Active Directory Mail Enabled User, 15-2
 - architecture, 15-4
 - Coexisting Third-Party User, 15-2
 - concepts, 15-3
 - configuring, 15-1 to 15-22
 - cross-scheduling, 15-4
 - installation, 15-4
 - introduction to, 15-1
 - managing, 15-1 to 15-22
 - Non-Coexisting Third-Party User, 15-2
- Oracle Change Notification Service for Exchange, 15-1, 15-3
- Oracle Communication Service, 15-1, 15-3
- Oracle Connector for Exchange, 15-1, 15-3
 - provisioning users, 15-15
- Public Free/Busy Folder, 15-3
- terminology, 15-2
- Oracle Connector for Exchange, 15-21
 - administrative tasks, 15-19
 - starting, 15-20
 - stopping, 15-19, 15-20
- Oracle HTTP Server, 2-4
- Organizations
 - about, 3-2

P

- participation mode, 6-20
- Personal Workspaces
 - about, 3-2
- Platform Service, 2-4, 5-21
- platform services
 - Coexistence Service, 5-21
 - platform services, 5-21
 - Platform Service, 5-21
 - Platform Web Service, 5-21
- Platform Web Service, 2-4, 5-21
- Policies
 - provisioning and deprovisioning
 - about, 3-3
- policies, 12-1 to 12-18
 - auditing policies, 12-16
 - example, 14-9
 - creating new, 12-16
 - custom, 12-16
 - deleting, 12-18
 - deprovisioning policy, 12-11
 - example, 12-12
 - editing, 12-17
 - introduction, 12-1, 12-4
 - managing, 12-6
 - password policy, 12-13
 - example, 12-14
 - provisioning policies
 - example, 12-8, 12-10
 - user attributes, 12-7
 - provisioning policy, 12-6
- Policy Service, 2-4, 5-7
- preference properties
 - voicemail, 11-3
- Presence Service, 5-8
- privileges
 - default, 13-3
- privileges (access control), 13-2
 - managing, 13-3
- provisioning users, 3-18

R

- recovering (from backup), 16-6
- relay (e-mail)
 - relay servers, 8-12
- relay configuration (e-mail)
 - VMS evaluators, 8-5
- Resource Directory Service, 2-4, 5-15
- resources
 - attributes, 4-2
 - booking characteristics options, 4-8
 - creating, 4-3 to 4-4
 - deleting, 4-6 to 4-7
 - introduction, 4-1
 - listing, 4-4 to 4-5
 - managing, 4-1 to 4-8
 - managing resource approvers, 4-7 to 4-8
 - modifying, 4-6
- restarting, 2-1 to ??

Roles

- about, 3-17
- roles (access control), 13-3, 13-4
 - assigned roles, 13-8
 - deleting, 13-9
 - modifying, 13-8
 - role definitions, 13-5

S

Search Service, 2-4, 5-19

sensitivities

- about, 13-2
- in team workspaces, 6-22
- sensitivities (access control), 13-9
 - creating and managing, 13-10

Services

- Access Control Service, 2-4
- CalDAV Service, 2-4
- Coexistence Service, 15-1, 15-2
- Management Service, 2-4
- Mobile Data Sync Service, 2-4
- Presence Service, 2-4

services

- about, 5-2
- collaborative services
 - content management services, 5-11
 - Discussions Service, 5-11
 - E-mail Service, 5-12
 - fax message service, 5-12
 - instant message services, 5-13
 - managing, 5-10
 - meetings services, 5-13
 - Time Management Services, 5-14
 - Voice Message Service, 5-16
 - Zimbra services, 5-16
- core services, 5-2
 - Access Control Service, 5-3
 - Audit Service, 5-4
 - Authentication Service, 5-4
 - Client Management Service, 5-5
 - Device Management Service, 5-5
 - event services, 5-6
 - Identity Provider Service, 5-5
 - Management Service, 5-7
 - managing, 5-2
 - Presence Service, 5-8
- enterprise services, 5-17
 - mobility services, 5-17
 - Search Service, 5-19
 - subscription and notification services, 5-19
 - Workspace Service, 5-20
- managing, 5-1 to 5-22
 - introduction to, 5-1
- platform services, 5-21
 - Coexistence Service, 5-21
 - platform services, 5-21
- service instances, 5-2
- SMPP Delivery Service, 5-20
- SMTP Server

about, 8-2

starting, 15-19

Starting and Stopping Oracle Beehive

overview, 2-1

static groups, 3-34

stopping, 2-1 to ??

subscription and notification services, 5-19

Notification Delivery Service, 5-19

SMPP Delivery Service, 5-20

Subscription Service, 5-19

Subscription Service, 5-19

system actor

about, 3-18

T

team workspaces

- access control, 6-20
- creating, 6-13
- creating and managing, 6-13
- deleting, 6-16
- examples, 6-14, 6-15, 6-19
- managing membership, 6-19
- modifying, 6-16
- modifying quota, 6-16
- roles, 6-21
- sensitivities, 6-22
- viewing, 6-15
- visibility, 6-20

template keys (workflows), 12-21

templates

- workspaces
 - about, 6-9
 - modifying, 6-12
- worspaces
 - creating, 6-13
- Time Management Service, 2-4, 5-15
- Time Management Services, 5-14
- time management services
 - Alarm Service, 5-14
 - CalDAV Service, 5-15
 - Resource Directory Service, 5-15
 - Time Management Service, 5-15
 - Time Zone Service, 5-15
- Time Zone Service, 2-4, 5-15
- transcoding service, 5-14

U

UDS requirements for voicemail, 11-3

user accounts

- about, 3-3
- account status
 - changing, 3-25
 - created, 3-25
 - delete in progress, 3-27
 - deleted, 3-27
 - disabled, 3-26
 - enabled, 3-26
 - locked, 3-26

- marked for delete, 3-27
 - unlocked, 3-26
- attributes, 3-6
- creating custom properties, 3-28
- custom properties, 3-28
- deleting, 3-29
- disabling, 3-30
- managing, 3-23
- modifying, 3-28
- provisioning
 - in bulk, 3-21
- User Directory Service, 2-4, 5-8
 - about, 3-2
- user IDs
 - about, 3-13
- user preference properties
 - voicemail, 11-4
- users
 - account status
 - about, 3-18
 - managing and provisioning, 3-1 to 3-41
 - provisioning, 3-18
 - using beectl, 3-19

V

- virtual mail servers
 - about, 8-2
- virus scanning, 8-14, 8-15
- VMS evaluator
 - open relay (preventing), 8-6
- VMS evaluators, 8-5
- Voice Message Service, 2-4, 5-16
- voicemail
 - auto attendant, 11-26
 - installing, 11-26
 - auto attendants, 11-2
 - Cisco hardware requirements, 11-3
 - configuring the enterprise, 11-11
 - configuring the GUI, 11-22
 - configuring the Message Waiting Indicator, 11-22
 - configuring voice service, 11-4
 - creating users, 11-12
 - enterprise preference properties, 11-3
 - inheritance rules, 11-4
 - enterprise preferences, 11-11
 - facilities, 11-1, 11-12
 - creating, 11-12
 - facility properties, 11-3
 - infrastructure, 11-2
 - managing
 - introduction, 11-1
 - preference properties, 11-3
 - UDS requirements, 11-3
 - user preference properties, 11-4
 - voicemail users
 - creating, 11-12
 - voicemail, managing, 11-1 to 11-28

W

- WebDAV
 - access to workspaces, 6-25
- WebDAV Service, 2-4, 5-11
- whitelist (e-mail), 8-10
- Workflow Service, 2-4, 5-10
- workflows, 12-1 to 12-28
 - introduction, 12-1, 12-4
 - managing, 12-19
 - tasks, 12-20
 - template keys, 12-21
 - templates, 12-20
 - types, 12-19
 - workflow template commands, 12-27
 - workflow templates
 - example, 12-23, 12-25
- Workspace Service, 2-4, 5-20
- Workspaces
 - team workspaces
 - examples, 6-19
- workspaces
 - about, 6-1
 - address books, 6-4
 - announcements, 6-5
 - events, 6-5
 - file system directories, 6-23
 - FTP access, 6-25
 - locking, 6-4
 - managing, 6-1 to 6-36
 - managing categories, 6-17
 - managing files, 6-23
 - messaging, 6-5
 - personal
 - about, 6-2
 - managing, 6-8
 - properties and controls, 6-3
 - team
 - about, 6-2
 - team workspaces
 - access control, 6-20
 - creating, 6-13
 - creating and managing, 6-13
 - deleting, 6-16
 - examples, 6-14, 6-15
 - managing membership, 6-19
 - modifying, 6-16
 - roles, 6-21
 - sensitivities, 6-22
 - viewing, 6-15
 - visibility, 6-20
 - templates, 6-9
 - about, 6-9
 - basic personal workspace template, 6-10
 - basic team workspace template, 6-10
 - community of practice template, 6-10
 - contents, 6-10
 - creating, 6-13
 - deleting, 6-13
 - examples, 6-36
 - modifying, 6-12

- project workspace template, 6-10
- trash, 6-5
- WebDAV access, 6-25

X

XML files

- examples

- example user provisioning XML file, 3-42

XMPP Service, 2-4, 5-13

Z

Zimbra Connector Service, 5-16

Zimbra services, 5-16

Zimbra UI Service, 5-16

