Oracle® Communications Service Broker

Online Mediation Controller Implementation Guide

Release 6.1

E29452-02

May 2014

ORACLE®

Oracle Communications Service Broker Online Mediation Controller Implementation Guide, Release 6.1

E29452-02

Contents

4 Setting Up Orchestrated Charging Mediation

5 Using Degraded Mode

6 Offloading Subscriber Usage

7 Using the Event Notification Framework

8 Setting Up RADIUS Mediation for Accounting

9 Setting Up RADIUS Mediation for Authentication and Authorization

10 Setting Up Diameter Ro Mediation

11 Monitoring Online Mediation Controller

12 Using the Balance Manager

13 Configuring the Announcement Player Application

14 Redirecting Sessions

15 Configuring the Home Zones Application

16 Configuring the Threshold Notification Application

17 Implementing Overload Protection

A Diameter Ro to BRM Opcode Mapping

Preface

This document describes how to install, configure, and use the Oracle Communications Service Broker Online Mediation Controller.

Audience

This document is intended for system administrators, and system integrators who will set up or administer Service Broker OMC.

This documentation is based on the assumption that you are already familiar with:

- Service Broker concepts. For more information see *Oracle Communications Service Broker Release 6.1 Concepts Guide*
- The operating system on which your system is installed
- Telecommunications networks and protocols, especially Diameter and RADIUS
- Oracle Communications Billing and Revenue Management concepts. For more information, see the *Oracle Communications Billing and Revenue Management Release 7.5* documentation set.
- Oracle Communications Elastic Charging Engine concepts. For more information, see *Oracle Communications Elastic Charging Engine Documentation*.

Documentation Accessibility

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

Access to Oracle Support

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

Related Documents

For more information, see the following documents in the Oracle Communications Service Broker Release 6.1 documentation set and in the Oracle Communications Billing and Revenue Management Release 7.5 and Oracle Communications Elastic Charging Engine documentation sets:

- *Oracle Communications Service Broker Concepts Guide*

- *Oracle Communications Service Broker Signaling Server Units Configuration Guide*
- *Oracle Communications Service Broker Modules Configuration Guide*
- *Oracle Communications Service Broker Installation Guide*
- *Oracle Communications Service Broker Subscriber Store User's Guide*
- *Oracle Communications Service Broker Orchestration User's Guide*
- *Oracle Communications Service Broker System Administrator's Guide*
- *Oracle Communications Service Broker Security Guide*
- *Oracle Communications Service Broker Release Notes*
- (Optional) *Oracle Communications Billing and Revenue Management Release 7.5 Installation Guide*
- (Optional) *Oracle Communications Billing and Revenue Management Release 7.5 System Administrator's Guide*
- (Optional) *Oracle Communications Elastic Charging Engine Documentation*

Downloading Oracle Communications Documentation

Oracle Communications Service Broker documentation is available from the Oracle Software Delivery Web site:

`http://edelivery.oracle.com/`

Additional Oracle Communication documentation is available from Oracle Technology Network:

`http://www.oracle.com/technetwork/index.html`

1

Online Mediation Controller Overview

This chapter provides an overview of the Oracle Communications Online Mediation Controller.

Before you read this chapter, you should be familiar with Oracle Communications Service Broker concepts and architecture. See *Oracle Communications Service Broker Concepts Guide*, for more information.

About Online Mediation Controller

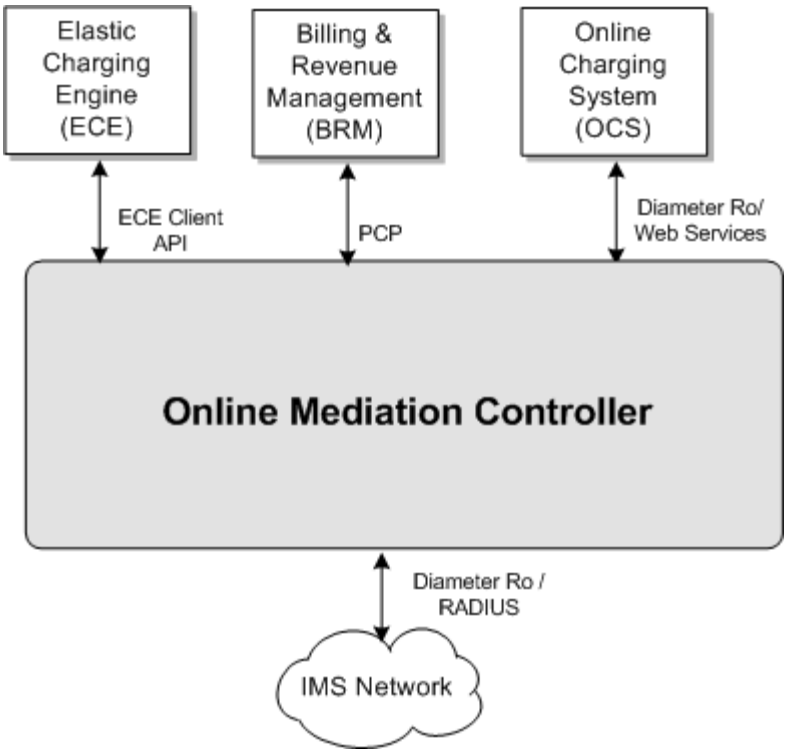
Online Mediation Controller acts as a single interface for online charging systems (OCSs) to the IMS network, mediating Diameter and RADIUS protocol messages to the OCS for rating, authentication, and authorization.

Online Mediation Controller provides network mediation for Oracle Communications Billing and Revenue Management (BRM), Oracle Communications Elastic Charging Engine (ECE) and third party OCSs. Online Mediation Controller supports concurrent connections to more than one OCS in the same domain, allowing mediation of messages to different OCSs.

When integrated with third party OCSs, Online Mediation Controller communicates through either a Diameter Ro interface, if supported, or a Web Services (HTTP, SOAP) interface. Integrations with BRM use the Portal Communications Protocol (PCP) for communication. When integrated with ECE, Online Mediation Controller mediates network charging requests using the ECE client API.

The integrations to the different OCS options are shown in Figure 1–1.

Figure 1–1 Online Charging Service Delivery to the Network



Online Mediation Controller also extends the OCS functionality traditionally associated with balance management and rating, with additional charging reliant features. In Degraded and Offline modes, Online Mediation Controller functions as one or more local online charging systems to handle requests when the external OCS is either unavailable or under excessive load.

- In Degraded Mode, Online Mediation Controller handles charging requests when the OCS is unavailable. See "Degraded Mode" for more information.
- In Offline Mode, Online Mediation Controller can service charging requests from low risk users and record usage for replay to the external OCS during off peak hours. See "Offloading Safe Subscriber Requests" for more information.

The following sections introduce key concepts and features of the Service Broker Online Mediation Controller.

Complementary Applications

Online Mediation Controller provides a Session Abstraction Layer (SAL) interface that you use to implement applications that extend your OCS functionality.

Applications that you implement reside on the session path inside Online Mediation Controller, optionally orchestrated and invoked with other external applications, and active during session setup and execution.

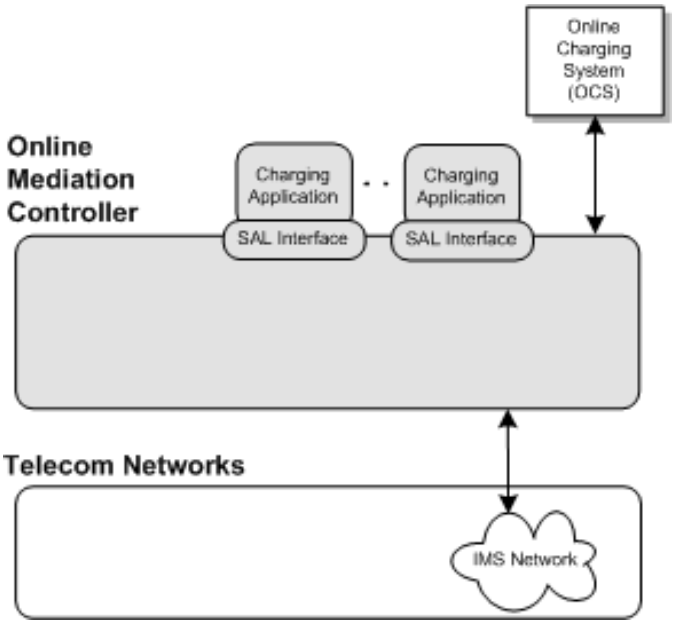
Applications can use any session parameter such as called party, call duration and charging information, to run their business logic, and affect session setup and execution synchronously. For example, an application can play an announcement when the subscriber balance goes down below a threshold, asking the subscriber to top up his account.

Applications can also invoke additional asynchronous activities in your system, by publishing events to external entities through a Web services API. For example, an application can notify your Operational Support Systems (OSS) when a subscriber tops up his account. The session itself is not affected by the notification, but your OSS can trigger asynchronous activities such as sending a notification email to the subscriber.

See O*racle Communications Service Broker Configuration and Runtime MBean Java API Reference*, for more information about the SAL interface.

Figure 1–2 shows applications implemented inside a Online Mediation Controller domain, that you combine, using service orchestration logic, with online charging services provided by the OCS.

Figure 1–2 Complementary Charging Applications



Subscriber Store

You can control service delivery for subscribers, specifying which applications to invoke and in what order, by basing orchestration logic conditions on the subscriber state, service type, geographical location and other charging request attributes, such as usage counters, stored in the Online Mediation Controller Subscriber Store.

You extend the subscriber lifecycle by defining the subscriber states your implementation requires. Each state implies a certain set of privileges and prohibitions. For example, you can configure Online Mediation Controller to charge **Active** subscribers, or redirect **Suspended** subscribers to top up their accounts. You can also publish and consume state transition notifications and actively request state transitions.

Online Mediation Controller provides two possible repositories for use as the Subscriber Store:

- **Local Subscriber Store** - Online Mediation Controller stores and updates subscriber profile data in a locally configured Oracle Berkeley or Oracle Enterprise 11g Database configured during domain creation. See *Oracle Communications*

Service Broker Subscriber Store User's Guide, for more information on the Local Subscriber Store.

- **BRM-based Subscriber Store** - Online Mediation Controller uses subscriber profile data sourced from the BRM database. See "Configuring the Subscriber Store", for more information on the BRM Subscriber Store.

Event Notification Framework

Complementary applications use the Event Notification Framework to publish events. The type of event and its timing are application-specific.

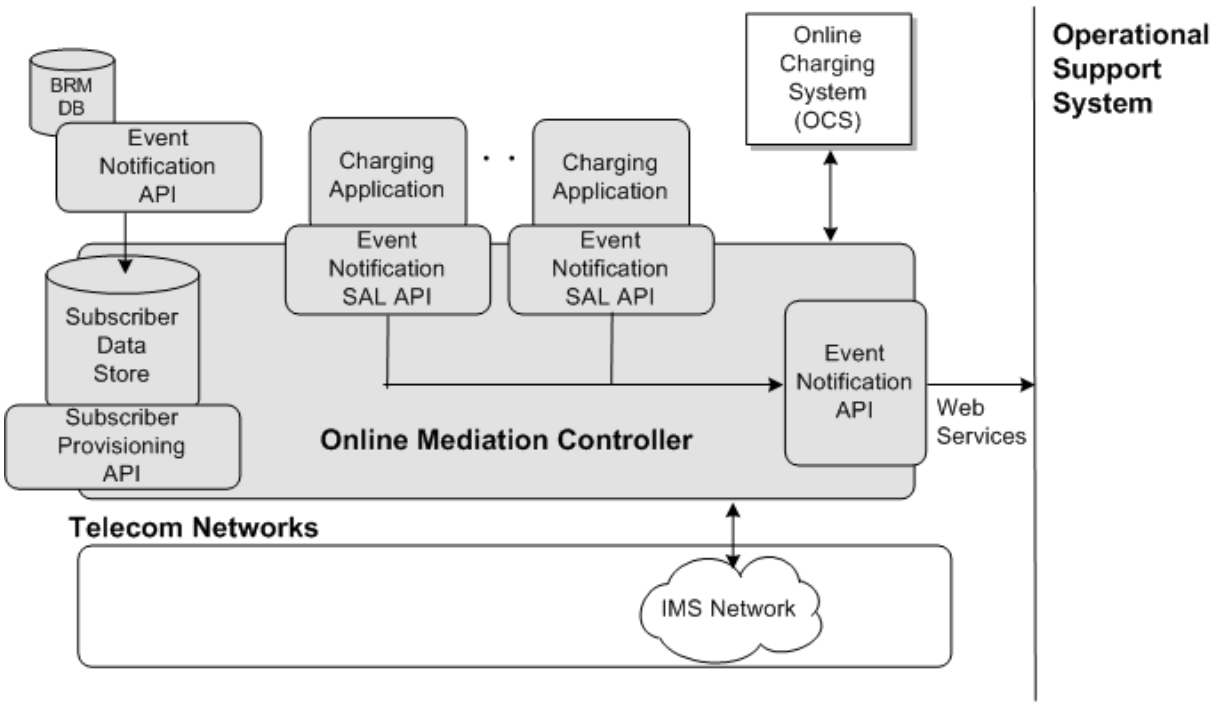
Events can be targeted to internal Online Mediation Controller components, or to external systems, such as your OSS. External systems use an asynchronous Web services Event Notification API to consume these events.

Using the Event Notification Framework, Online Mediation Controller can be integrated with external Business Intelligence (BI) systems, basing business logic on complementary application events. For example, BRM may fire an event when a subscriber reaches a predefined monthly usage threshold to update the subscriber's profile, driving further business logic on the OSS side such as data throttling.

OSS applications consuming Online Mediation Controller events may also use those events as triggers to change Online Mediation Controller behavior. OSS applications make changes to Online Mediation Controller using Web services APIs. For example, if an OSS application consumes an event noting that a subscriber has reached a predefined monthly credit threshold, then the OSS application can use the Subscriber Provisioning Web services API to suspend the subscriber account.

Figure 1–3 shows how complementary applications use the Event Notification API to publish events to external systems.

Figure 1–3 The Event Notification API



The Event Notification Framework is thereby a Online Mediation Controller extension point in which you can extend business logic, and link online activities related to charging and service delivery with other activities happening in your OSS.

Degraded Mode

Degraded Mode is activated when the OCS cannot respond to charging requests directed at it from Online Mediation Controller. This may happen when the OCS fails or during an OCS maintenance window.

Degraded Mode guarantees service continuity while the OCS is unavailable. During that time, Online Mediation Controller ensures service continuity by responding to charging requests and generating and storing call data records (CDRs) for subscriber activity.

When the OCS resumes processing, Online Mediation Controller forwards the CDRs to the OCS which rates and updates subscriber account balances accordingly.

See "Using Degraded Mode", for more information.

Offloading Safe Subscriber Requests

Online Mediation Controller can offload charging requests from the external OCS based on session attributes such as service ID, location, or resource counter value. Online Mediation Controller, acting as a local OCS, answers the qualifying requests with a preconfigured response and generates CDRs for each session. CDRs are stored for playback to the external OCS during off-peak time periods.

Offloaded generated CDRs from the same subscriber can also be compressed before playback for efficiency.

Using Offline Mode allows service providers to optimize OCS utilization and reduce the possibility of OCS failure due to overloading.

See "Offloading Subscriber Usage", for more information.

Overload Protection

Online Mediation Controller contains a configurable, built-in overload protection mechanism preventing failures due to excessive request load.

See "Implementing Overload Protection", for more information.

Session Redirection

Online Mediation Controller contains a configurable Redirection application that redirects subscriber sessions based on low or empty resource responses received from an online charging system. For example, the Redirection application sends sessions with expiring resources to the Announcement Player application, notifying the subscriber of their remaining balance.

See "Redirecting Sessions", for more information.

User Interaction Framework

You use the User Interaction Framework to contact subscribers with charging-related information. Online Mediation Controller supports these channels for user interaction:

- Short Message Service (SMS): You can send SMS messages to subscribers using the Short Message Peer to Peer (SMPP) protocol. For example, you can send an SMS message to a subscriber when his account is being activated, after he uses a service for the first time.
- Announcements: You can use the User Interaction Framework to play mid-call announcements to either the calling party or called party.

Monitoring Online Mediation Controller

You can monitor Online Mediation Controller using a collection of runtime MBeans. Online Mediation monitors provide information on the performance of processing modules, request orchestration and connected network entities.

See "Monitoring Online Mediation Controller", for more information about monitoring Online Mediation Controller.

2

Installing and Configuring Online Mediation Controller

This chapter explains how to install and configure Oracle Communications Online Mediation Controller.

Online Mediation Controller Hardware and Software Requirements

Online Mediation Controller does not have additional hardware needs beyond those listed in *Oracle Communications Service Broker Installation Guide*. See *Oracle Communications Service Broker Installation Guide*, for more information on hardware requirements.

Install Online Mediation Controller by completing the procedures for installation and domain creation as described in the chapter in *Oracle Communications Service Broker Installation Guide*.

A typical Online Mediation Controller implementation may also contain the following components:

- An online charging system (OCS) such as Oracle Communications Billing and Revenue Management (BRM) or Oracle Communications Elastic Charging Engine (ECE).
- Network components or simulators that generate traffic for Online Mediation Controller to mediate.
- An internal or external data source functioning as a subscriber store.
- A client workstation containing a supported Web browser and other development tools such as a MBean browser or other test tools.

Installing and Deinstalling Online Mediation Controller

Online Mediation Controller is a Service Broker component installed by selecting the **Online Mediation Controller 6.1.0.0** option during installation. For details on installing and deinstalling Online Mediation Controller, see *Oracle Communications Service Broker Installation Guide*.

Creating Domains, and Managed and Administration Servers

An Online Mediation Controller implementation must have an Administration Server and one or more Managed Servers in the same domain. A single Online Mediation Controller node can contain both one Administration Server and one Managed Server. The Online Mediation Controller installation includes domain configuration scripts.

See the chapter on setting up domains in the *Oracle Communications Service Broker Installation Guide*, for more information.

The system topology required by your implementation needs differs depending on intended Online Mediation Controller use. For example, configure a basic development system with the Administration Console and Online Mediation Controller processing and signalling domains on the same physical server in a unified domain. Production systems may include multiple managed servers for load balancing and separate domains for processing and signalling. For additional information on configuring Online Mediation Controller domains and servers, see *Oracle Communications Service Broker Administrator's Guide*.

Configuring Online Mediation Controller

Oracle Communications Online Mediation Controller Implementation Guide contains the information needed to configure Online Mediation Controller.

See "Online Mediation Controller Overview", for an overview of Online Mediation Controller including references to the procedures used for configuration in this guide.

3

Configuring the Subscriber Store

This chapter describes how to use subscriber store information with Oracle Communications Online Mediation Controller.

About the Subscriber Store

To deliver subscriber-specific services, Online Mediation Controller maintains a subscriber profile for each subscriber in the Subscriber Store. The Subscriber Store uses either data populated from the Subscriber Provisioning API (Local Subscriber Store) or data sourced from an Oracle Communications Billing and Revenue Manager (BRM) database. You use only one of these options.

This chapter primarily describes the BRM-based Subscriber Store. See *Oracle Communications Service Broker Subscriber Store User's Guide*, for more information on using the Local Subscriber Store.

About Subscriber Profiles

A subscriber profile contains subscriber-specific information used by Online Mediation Controller's built-in applications. For example, the Balance Manager application must know a subscriber's default language to compose SMS messages sent to a subscriber. Online Mediation Controller uses subscriber profiles in determining subscriber state and offline charging eligibility.

Subscriber profiles in the Subscriber Store include:

- The service profile.
- An orchestration logic that defines how to route subscriber sessions through applications.
- An extensible subscriber lifecycle that you use to define subscriber states and state transitions.
- Usage counter and threshold information.

Orchestration logic in the subscriber profile is stored in iFC format. Oracle recommends that you use the Online Mediation Controller Orchestration Studio to graphically create orchestration logic.

While conditions in the orchestration logic are mostly based on session information, they can also depend on the following subscriber data available in the Subscriber Store:

- Degraded Mode state. See "Using Degraded Mode".
- Offline Mode eligibility. See "Offloading Subscriber Usage".

For example, you can configure the orchestration logic to route a session to the OCS only if the subscriber state is **Active**. You can also determine whether a subscriber's session can be routed to an offline mode local OCS based on the remaining prepaid minutes in their BRM-sourced resource counter balance.

In Online Mediation Controller, you must configure the Orchestration Engine (OE) to use the Subscriber Store as the source for subscriber profiles.

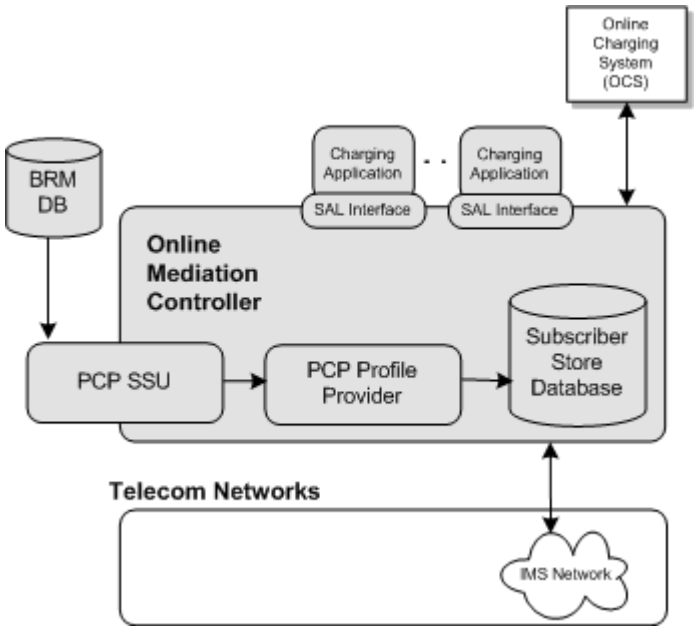
Using the BRM Subscriber Store

Service providers implementing both Online Mediation Controller and BRM can use BRM subscriber data as the Subscriber Store. The *PCP Profile Provider* reads BRM subscriber data using the PCP Signaling Server Unit (SSU) and caches the information for Online Mediation Controller use. BRM remains the system of record for subscriber data when using the PCP Profile Adapter.

The Subscriber Store updates BRM subscriber data by listening for Oracle Advanced Queuing (AQ) notification messages generated by the BRM database. When Online Mediation Controller receives a message it checks if any of the changes apply to users with active sessions. If any such users are found the PCP Profile Provider initiates a refresh of their Subscriber Store data for affected users.

Figure 3–1 shows the flow of data read by the PCP Profile Provider from the BRM database to the Subscriber Store.

Figure 3–1 The BRM Subscriber Store



See "Subscriber Profile Data Model", for additional information on the attributes retrieved and mapped from BRM into the Subscriber Store.

See "Configuring the BRM Subscriber Store", for information on configuring the BRM Subscriber Store.

See "PCP Profile Provider Data Mapping", for information on field mapping between BRM and the Subscriber Store.

Configuring the BRM Subscriber Store

To configure Online Mediation Controller to use the BRM database as the Subscriber Store source:

1. Verify that the PCP persistence bundle is installed and active. See "Configuring the PCP Profile Provider", for more information.
2. Create a connection pool for the BRM database. See the information on creating database connections in the *Oracle Communications Service Broker Installation Guide* for more information on setting up the BRM JDBC connection.
3. Configure the SSU connection to BRM. See the chapter on configuring a PCP Signaling Service Unit in *Oracle Communications Service Broker Signaling Server Units Configuration Guide*, for more information.
4. Configure the Subscriber Store Provider containing the queue from where BRM-generated notifications for updates to subscriber profile data are published. See "Configuring a Subscriber Store Provider", for more information.
5. Populate and maintain the Subscriber Store with user information.
6. Configure the OE to use the Subscriber Store as a source of subscriber profiles.

   Configure the OE as described in "Configuring the Orchestration Engine" in *Oracle Communications Service Broker Orchestration User's Configuration Guide*. Use the following configuration data, specifically:

   - In the General tab, in the **Subscriber Profile Receiver** list, select **OlpCustomInfoReceiver**.
   - In the Custom OLP tab, in the **Custom OPR Name** field, enter **UP OPR**.

Configuring the PCP Profile Provider

To configure the Subscriber Store to use PCP Profile Provider in the Administration Console:

1. Expand **Domain Management**.
2. Click **Packages**.
3. Remove the existing local Subscriber Store persistence bundle from the domain using the **Uninstall** button. By default, it is:

   oracle.ocsb.app.rcc.service.subscriber_store.providers.store.provider
4. Install and start the PCP Profile Provider with a level of 260 by clicking **Install** after selecting this bundle:

   oracle.ocsb.app.rcc.service.subscriber_store.providers.pcp.jar
5. Make sure the run level for the package matches that of the following package:

   oracle.ocsb.app.rcc.service.subscriber_store.core
6. Configure the database schema for the Subscriber Store using the following SQL script:

   *Oracle_home***/ocsb/admin_server/scripts/database/subscriber_store.sql**

   To use the script to configure your database schema, run the script using a SQL client tool, such as sqlplus, or interface provided by your database management system.

7. If the managed servers in the domain are not already configured to connect to the database, configure the connections in **Data Store** under **Domain Management**.

   For the Subscriber Store database, the connection pool name value in the JDBC configuration should be the name of your BRM database that you set in the "Configuring a Subscriber Store Provider" section. For example, **oracle_driver**.

   See *Oracle Communications Service Broker Installation Guide* for more information about configuring data storage.

8. Configure the PCP signaling server units (SSU) to connect to BRM.

   See the chapter on configuring PCP signal server units in *Oracle Communications Service Broker Signaling Server Units Configuration Guide*.

Configuring a Subscriber Store Provider

Configure the Subscriber Store Provider indicating the connection pool and queue on which to listen for BRM generated Oracle AQ notifications.

1. If the Administration Console is not already in offline mode, click the **Switch to OFFLINE mode** icon at the top of the page.
2. Click **Lock & Edit**.
3. Expand **OCSB** and then **Domain Management**.
4. Expand **Subscriber Store**.
5. Click **Providers**.
6. In the **BRM** tab, add the Subscriber Store Provider information as a new **Notifications** entry by clicking **New** and providing the following information:
   a. Enter a unique **Name** for the provider.
   b. In the **Connection Pool Reference** field, enter the name of the JDBC connection pool created for the BRM database. This value must be identical to the connection pool value.
   c. In the **Queue Owner** field enter the BRM database user with access to the notification queue.
   d. In the **Queue Name** field, enter the name of the notification queue as defined in the BRM database.
   e. In the **Number of Sessions field**, enter the number of connections to open to the Provider. This value cannot be larger than the connection pool size.
7. Click **OK**.

   The new provider instance appears in the list of Notifications.
8. Click **Commit**.

Subscriber Profile Data Model

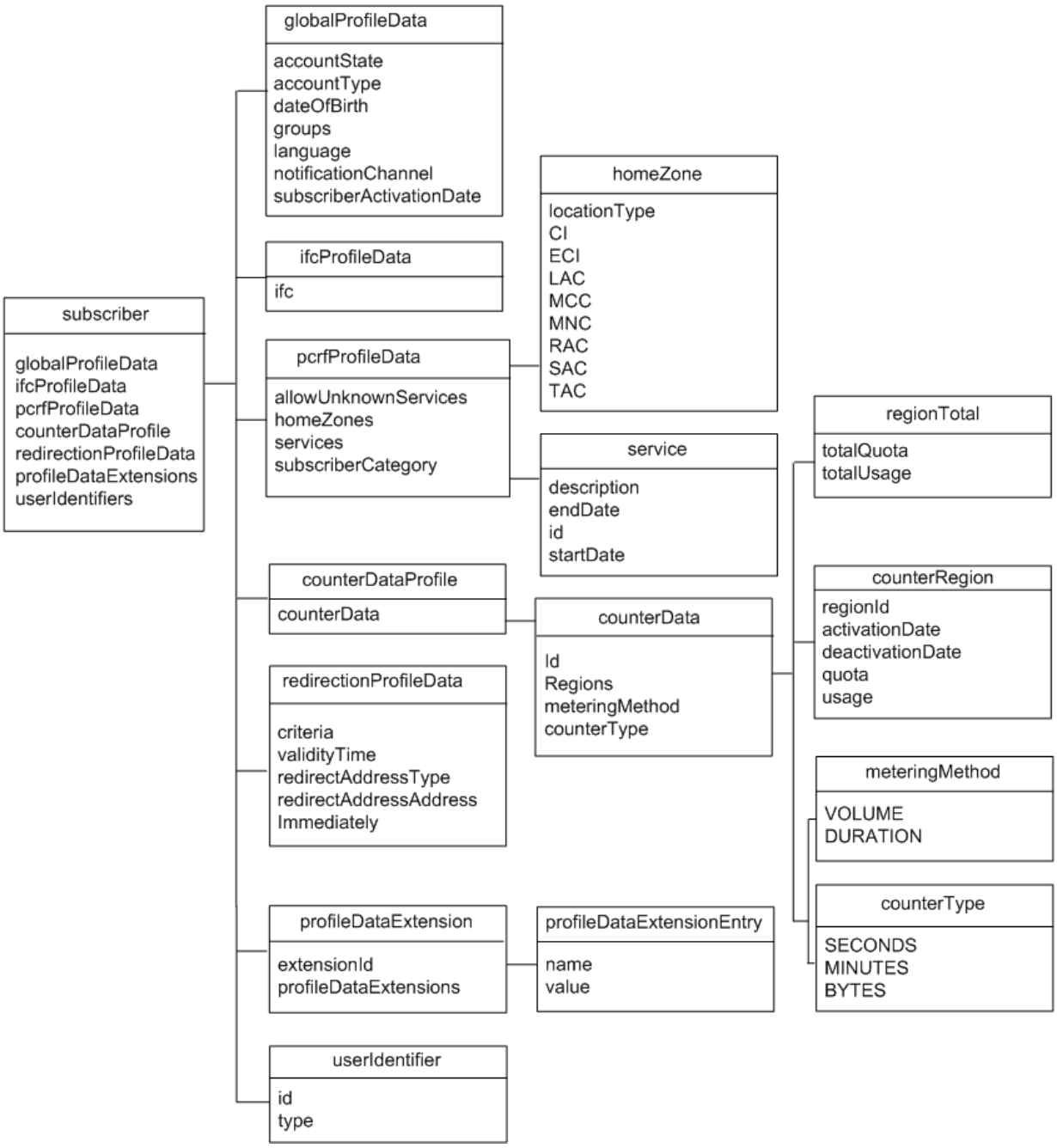
The subscriber profile data model defines the elements of a subscriber profile. The model includes general information for the subscriber along with feature-specific information, such as Policy Controller data. Figure 3–2 illustrates the subscriber profile data model.

The subscriber profile data model is defined by an XML Schema file. You can access the schema at the following default location:

http://*host*:*port*/soap/SubscriberProvisioning?xsd=1

where *host* is the host name or IP address and *port* is the server port number you specified as the server address.

Figure 3–2 Subscriber Profile Data Model



As shown in Figure 3–2, the top-level data elements that compose the subscriber profile are:

- globalProfileData Element
- ifcProfileData Element

- pcrfProfileData Element
- counterProfileData Element
- redirectionProfileData Element
- profileDataExtension Element
- userIdentifier Element

The following sections provide information on the data elements of the subscriber profile.

globalProfileData Element

The **globalProfileData** element defines general properties for a subscriber. Multiple Online Mediation Controller applications use this data. This element contains the following elements:

- **AccountState**: The account lifecycle state that indicates the status of the account, such as active or inactive.
- **AccountType**: The account type from the options prepaid, postpaid, or hybrid. Hybrid accounts can use both prepaid and postpaid charging methods.
- **DateOfBirth**: The birth date of the subscriber. The date is in *YYYYMMDD* format. For example, **19910128**.
- **Groups**: Group identifier for the subscriber.
- **Language**: The language to use for the subscriber. Online Mediation Controller applications that interact with subscribers can use this attribute to select a response or adapt a message for the subscriber.
- **NotificationChannel**: The notification channel used to send notifications to the user.
- **SubscriberActivationDate**: The date, in *YYYYMMDD* format, when the subscriber account was first activated, such as **20111231**.

ifcProfileData Element

The **ifcProfileData** element contains the iFC logic for the subscriber. The iFC defines the service chain for a particular subscriber, which services are invoked and the order in which they are invoked.

pcrfProfileData Element

The **PcrfProfileData** element contains subscriber data used by the Policy Controller. **PcrfProfileData** includes the following elements:

- **AllowUnknownServices**: Whether the subscriber is permitted to access services that are not specifically allocated to that subscriber, as indicated by the **services** field.
- **HomeZone**: The home, non-roaming network zone for the subscriber. Policy Controller administrators can apply charging rates or service access decisions based on whether the subscriber is using the network from their home zone.

  The **HomeZone** element is made up of a **LocationType**, which identifies the protocol type in which the location information is represented, such as IEEE-802.11a or 3GPP-GERAN. Depending on the location type, it also includes elements that identify the location as specified for that type.

Possible values for **LocationType** can be: **CGI**, **SAI**, **RAI**, **TAI**, **ECGI**, **TAI_ECGI**. Depending on the value of **LocationType**, the following additional elements may also be present:

- **CI**
- **ECI**
- **LAC**
- **MCC**
- **MNC**
- **RAC**
- **SAC**
- **TAC**

For example, if the location type is **CGI**, then the location-type specific parameters that should be set are: **CI**, **LAC**, **MCC**, and **MNC**.

For more information on the location identification protocol, see the following 3GPP specifications:

- TS 29.060: http://www.3gpp.org/ftp/Specs/html-info/29060.htm
- TS 29.061: http://www.3gpp.org/ftp/Specs/html-info/29061.htm
- TS 29.274: http://www.3gpp.org/ftp/Specs/html-info/29274.htm

- **Services** contains one or more **Service** elements. The **Service** defines the services this subscriber can access. Each service has the following fields:
  - **description**: A description of the service.
  - **endDate**: The date, in YYYYMMDD format, on which the service availability ends for an individual subscriber. Along with the **startDate** value, this value enables you to make services available to subscribers within a specific date range.
  - **id**: The identity of the Policy Controller application described by this **service** element. See *Oracle Communications Service Broker Policy Controller Implementation Guide* for more information.
  - **startDate**: The date, in YYYYMMDD format, on which service availability begins. Along with the **endDate** value, this value enables you to make services available to subscribers within a specific date range.
- **SubscriberCategory**: An application-defined service level for a given subscriber, such as gold, silver, or bronze. The Policy Controller rules determine how a category dictates the level of service access for the subscriber.

counterProfileData Element

The **counterProfileData** element contains subscriber usage data represented as one or more **counterData** elements. Each **counterData** element identifies a resource or usage balance and its associated **regions** defined by a set of consecutive value ranges. See "Understanding Counter Regions", for an example of how **regions** can be used in the Subscriber Store.

CounterData contains the following:

- **Id**: A unique string identifier for the counter.

- **Region(s)**: One or more consecutive **CounterRegion** resource balance value ranges used to categorize a subscriber's usage status. The value of a subscriber's resource counter determines the region that the subscriber is currently in.

  The **CounterRegion** element consists of the following:

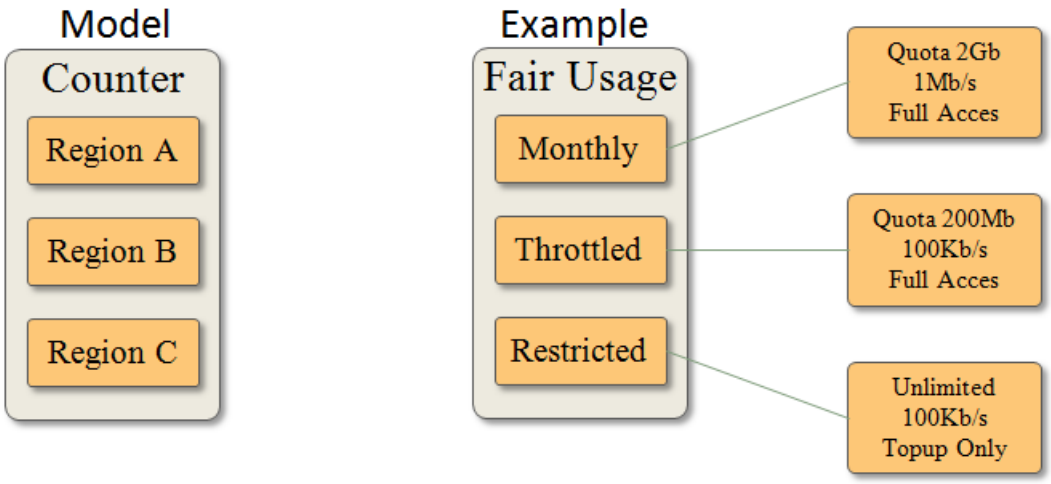
  - **regionID**: A unique string identifier for the region.
  - **activationDate**: The date on which the region became active. For example, **20120712**.
  - **deactivationDate**: The date on which the region will become inactive. For example, **20120815**.
  - **quota**: The available quota for the region if this region has a quota.
  - **usage**: The currently used resource amount for the region.
- **MeteringMethod**: Indicates a how counter tracks subscriber usage. The following methods are supported:
  - **VOLUME**: The amount of resource usage. For example, bytes.
  - **DURATION**: The length of time a resource is used. For example, seconds or minutes.
- **CounterType**: Indicates the type of metric used to track subscriber usage. The following types are supported:
  - **SECONDS**
  - **MINUTES**
  - **BYTES**
- **RegionTotal**:
  - **totalQuota**: The total amount of resource quota available in a region.
  - **totalUsage**: A subscriber's total resource usage within a region.

Understanding Counter Regions

The following sample scenario provides an example of how counter regions are used in the Subscriber Store.

In this example, counter regions represent a fair usage scenario for a data subscriber. The subscriber's service provider implements a data plan including reduced (throttled) data speeds triggered by the subscriber's total data usage across the thresholds as shown in Figure 3–3.

Figure 3–3 Counter Regions Example



As the subscriber's usage accumulates in the fair usage counter, he moves through the **Monthly**, **Throttled**, and **Restricted** counter regions. The subscriber exists in only one region at a time.

Online Mediation Controller applications can use the subscriber's current region to enforce the correct data throttling and access limitations. The bandwidth and access values in the example are not part of the counter data model, but rather part of the rules enforced based on the data model.

redirectionProfileData Element

The **redirectionProfileData** element defines subscriber redirection behavior. The Service Broker Redirection Application uses this data to redirect sessions requiring subscriber attention, such as a low balance, to an alternative destination application. Redirection data stored in a subscriber's profile overrides the default redirection behavior configured in the charging IM.

See the chapter on the Redirection Application in *Oracle Communications Service Broker Online Mediation Controller Implementation Guide* for more information.

The following fields are included in the redirectionProfileData element:

- **criteria**: Indicates the type of message triggering redirection supplied by the online charging system. Supported values are **FUI**, **NO_MONEY_NOTIFICATION**, and **LOW_CREDIT_NOTIFICATION**.
- **redirectAddressType**: Defines the address type of the address specified in the element. Supported values are **IPV4**, **IPV6**, **URL**, and **SIP_URI**.
- **redirectAddressAddress**: The address of the redirection server.
- **validityTime**: The allowed time, in seconds, remaining for a redirected subscriber to access network resources.
- **immediately**: Indicates whether redirection should happen **IMMEDIATELY** or **NON_IMMEDIATELY** (after the value specified in **Validity Time**). When the type is set to NO_MONEY, this value must be set to IMMEDIATELY.

profileDataExtension Element

The **ProfileDataExtension** element stores custom data elements in the profile. For example, the Policy Controller rule engine can make policy decisions based on dynamically defined subscriber attributes.

The **ExtensionId** value identifies the Online Mediation Controller component that uses the data extension. For the Policy Controller, for example, the **ExtensionId** value is **pcrf**. Each data extension entry consists of one or more name-value pairs, as specified by the Online Mediation Controller component that uses it.

userIdentifier Element

The **UserIdentifier** element contains the unique identifier for a subscriber on a particular network. A subscriber can belong to multiple networks, for instance, if they own multiple devices used to access different networks. Therefore, a subscriber can have multiple **UserIdentifier** elements.

Each **UserIdentifier** element consists of the identifier value and the type of the identifier. The type represents the system in which the ID belongs. It may have one of the following values:

- **END_USER_E164**: The subscriber's identity in ITU-T E.164 format, as defined in recommendations E164 and CE164.
- **END_USER_IMSI**: The subscriber's identity in International Mobile Subscriber Identity format, as defined in ITU-T recommendations E212 and CE212.
- **END_USER_SIP_URI**: The subscriber's identity in a SIP network.
- **END_USER_NAI**: The subscriber's identity in a mobile IP Network Address Identifier format.
- **END_USER_PRIVATE**: The subscriber's private identity in a credit-control server.
- **END_USER_GLOBAL_UID**: A unique global user ID generated by Online Mediation Controller to identify subscribers internally. Online Mediation Controller generates this ID automatically when you create a subscriber profile.

The value for each identifier type varies by the network type. For example, the identifier for the URI type should be in the form of a SIP URL, such as **sip:username@example**.

PCP Profile Provider Data Mapping

Online Mediation Controller maps BRM subscriber profile data to Subscriber Store attributes used in determining a subscriber's state or service usage eligibility. For more information about BRM data to Subscriber Store mapping, see the chapter on using the Subscriber Provisioning API in *Oracle Communications Service Broker Subscriber Store User's Guide*.

The following tables describe the mapping of Subscriber Store profile data fields to the BRM Flist fields used in the BRM database.

Table 3–1 describes the **GlobalProfileData** fields mapping between the Subscriber Store and BRM database.

Table 3–1 GlobalProfileData Fields Mapping

GlobalProfileData Field	Type	Flist Field Name	Flist Field Type
AccountState	String	globalProfileData.accountState	String
NotificationChannel	String	globalProfileData.notificationChannel	String
Groups	List	globalProfileData.groups.[n]	N/A
AccountType	String	globalProfileData.accountType	String
SubscriberActivationDate	String	globalProfileData.subscriberActivationDate	String
DateOfBirth	String	globalProfileData.dateOfBirth	String

Table 3–2 describes the **IfcProfileData** field mapping between the Subscriber Store and BRM database.

Table 3–2 IfcProfileData Field Mapping

IfcProfileData Field	Type	Flist Field Name	Flist Field Type
Ifc	String	ifcProfileData.ifc	String

Table 3–3 describes the **PcrfProfileData** field mapping between the Subscriber Store and BRM database.

Table 3–3 PcrfProfileData Fields Mapping

PcrfProfileData Field	Type	Flist Field Name	Flist Field Type
SubscriberCategory	String	pcrfProfileData.subscriberCategory	String
HomeZones	List	pcrfProfileData.homeZone.[n]	N/A
Home.GeographicLocationType	Enum	pcrfProfileData.homeZone.[n].geographicalLocationType	String
Home.LocationData	Map	pcrfProfileData.homeZone.[n].locationData	String
Home.LocationField	Enum	pcrfProfileData.homeZone.[n].locationField	String
Services	List	pcrfProfileData.services.[n]	N/A
Service.id	String	pcrfProfileData.services.[n].id	String
Service.description	String	pcrfProfileData.services.[n].description	String
Service.startDate	String	pcrfProfileData.services.[n].startDate	String
Service.endDate	String	pcrfProfileData.services.[n].endDate	String
AllowUnknownServices	Boolean	pcrfProfileData.allowUnknownServices	String

Table 3–4 describes the **CounterProfileData** field mapping between the Subscriber Store and BRM database

Table 3–4 CounterProfileData Fields Mapping

CounterProfileData Field	Type	Flist Field Name	Flist Field Type
Id	String	counterProfileData.id	String
CounterType	Enum	Mapped from FldGroupInfo	FldRumName
MeteringMethod	Enum	Derived from CounterType	FldRumName
Regions	List	counterProfileData.region.[n]	N/A

Table 3–4 (Cont.) CounterProfileData Fields Mapping

CounterProfileData Field	Type	Flist Field Name	Flist Field Type
CounterRegion.regiodId	String	counterProfileData.region.[n].regionId	String
CounterRegion.value	Long	counterProfileData.region.[n].value	String
CounterRegion.activationDate	String	counterProfileData.region.[n].activationDate	String
CounterRegion.deactivationDate	String	counterProfileData.region.[n].deactivationDate	String

Table 3–5 describes the **ProfileDataExtension** field mapping between the Subscriber Store and BRM database

Table 3–5 ProfileDataExtension Fields Mapping

ProfileData Extension Field	Type	Flist Field Name	Flist Field Type
Id	String	profileDataExtension.[n].id	String
Data	Map	profileDataExtension.[n].data.[n]	N/A
Data.key	String	profileDataExtension.[n].data.[n].key	String
Data.value	String	profileDataExtension.[n].data.[n].value	String

4

Setting Up Orchestrated Charging Mediation

Oracle Communications Online Mediation Controller provides a way for online charging systems (OCS) to charge subscribers and accounts, for services that they use in the IMS network.

This chapter describes how you set up and configure orchestrated charging mediation.

About Orchestrated Charging Mediation

An OCS allows communications service providers to charge their subscribers, in real time, based on service usage. An OCS authenticates subscribers and maintains accounts that subscribers use to pay for services that they use. The OCS can influence, in real time, the service rendered to a subscriber and therefore needs direct interaction with the communications network.

Online Mediation Controller provides an OCS with a front-end interface to the network. Online Mediation Controller communicates with the network through its native protocols, exposing one unified interface to the OCS.

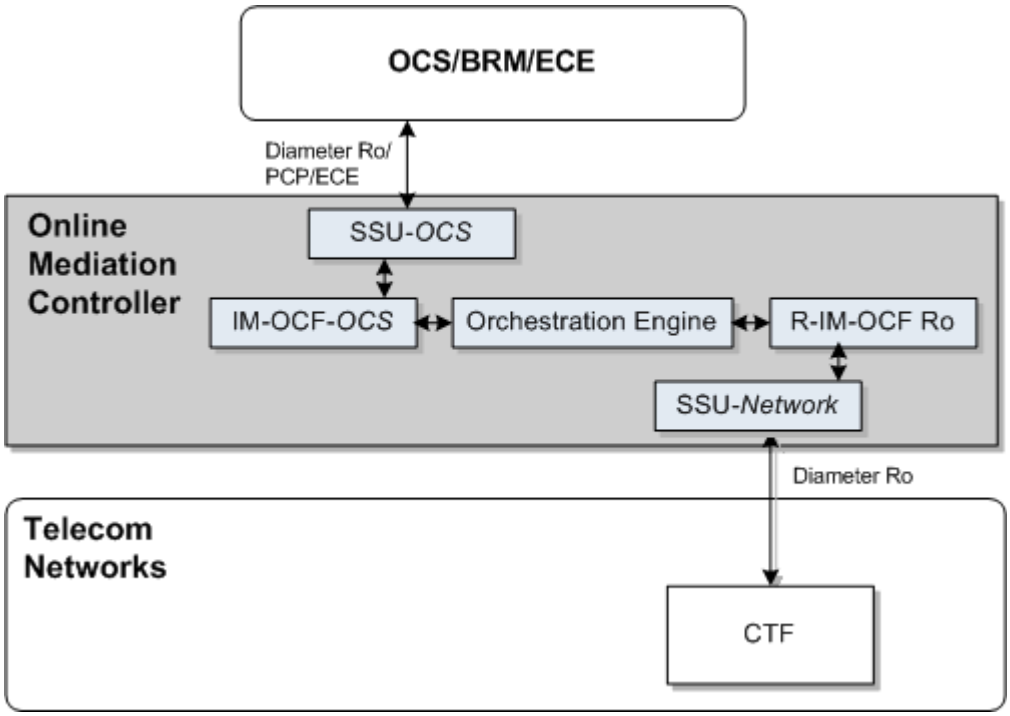
The interface is based on either a standard Diameter Ro, Portal Communications Protocol (PCP) or Oracle Communications Elastic Charging Engine (ECE) API. You can use any of the three interfaces, depending on the type of OCS implemented. For integration with BRM the PCP interface is used. For integration with the ECE, the ECE API is used.

Figure 4–1 shows the application-facing interworking modules, network-facing interworking modules, and the Signaling Server Units (SSUs) that you need to deploy and configure to apply an online charging service.

You use:

- RIM-OCF, to charge services in the IMS network
- Depending on the type of OCS implemented, use one of the following to interfaces to process charging requests:
  - OCS supporting a standard Diameter Ro interface: IM-OCF-Ro
  - Oracle Communications BRM: IM-OCF-PCP
  - Oracle Communications ECE: IM-OCF-ECE

Figure 4–1 Orchestrated Charging Mediation



To charge subscribers for voice sessions, data sessions and other service usage, in real time, you need to configure your network to route sessions and service usage events to Online Mediation Controller.

When you route subscriber sessions and events through Online Mediation Controller, you configure the subscriber's orchestration logic in the Subscriber Store to forward sessions to the OCS. You can also configure Online Mediation Controller to forward sessions to additional applications, thereby applying more services in your network to sessions, in addition to the charging service applied by the OCS.

Improving Performance in Diameter Only Environments

Interworking Modules (IMs) communicate with the Orchestration Engine (OE) using a proprietary internal protocol. This protocol allows Online Mediation Controller to act as a mediator between different protocols.

When a network-facing IM receives a session from the network, the IM translates the session from its original protocol to the Online Mediation Controller internal protocol. Then the application-facing IM translates the session from the internal protocol to the protocol of the interface that this IM provides. The protocol translation allows Online Mediation Controller to perform mediation between different protocols.

For example, when you use Online Mediation Controller as a mediation solution between Diameter and SIP networks, the Online Mediation Controller internal protocol allows an IM which provides the interface for Diameter and an IM which provides the interface for SIP to communicate with each other using a common internal language.

In online charging solutions using the Diameter protocol for communication between both the network and OCS, Online Mediation Controller performance can be improved by eliminating the translation from Diameter to the Online Mediation

Controller internal protocol and from the Online Mediation Controller internal protocol back to Diameter.

You can instruct Online Mediation Controller to encapsulate Diameter messages as they are to the body of the Online Mediation Controller internal protocol. In this case, Online Mediation Controller does not perform any protocol translation, but uses the internal protocol for tunnelling purposes only. This mode is known as **Diameter-based Orchestration Mode**.

You activate the Diameter-based orchestration mode in an R-IM-OCF using Java MBeans. See "Connecting to the IMS Network" for more information on how to activate the Diameter-based orchestration mode.

Configuration Workflow

To set up an end-to-end configuration for online charging perform the following steps:

1. To charge sessions and events in the IMS domain, connect Online Mediation Controller to the IMS network. See "Connecting to the IMS Network".
2. Connect Online Mediation Controller to your OCS. Depending on the type of OCS in your system, do one of the following:
   - OCS supporting a standard Diameter Ro interface: See "Connecting to an OCS through Diameter Ro".
   - BRM using PCP: See "Connecting to BRM through PCP".
   - ECE using the ECE API: See "Connecting to ECE Using the ECE API".
3. Route subscriber sessions to the OCS using orchestration logic. See "Adding the OCS to the Service Orchestration Chain" for more information.

Connecting to the IMS Network

To connect Online Mediation Controller to the IMS network:

1. Define Online Mediation Controller as a Diameter node and configure how other Diameter entities access it, as described in the section "Creating a Diameter Node" in "Configuring Diameter Signaling Server Units" in *Oracle Communications Service Broker Signaling Server Units Configuration Guide*.

   In the Administration Console:

   a. In the navigation tree, expand **OCSB**, and then **Signaling Tier**.

   b. Select **SSU Diameter**.

   c. In the **DIAMETER** tab, select the **Diameter Configuration** tab.

   d. You can either use the default node or create a new node by clicking **Add** on the bottom of the list of existing Diameter nodes.

   e. In the **General** tab, in the **Name** field, enter a unique name for the Diameter node.

   f. In the **Realm** field, enter the realm name that other Diameter nodes use to access Online Mediation Controller.

   g. In the **Port** field, enter the port number that signaling servers use to listen to Diameter traffic.

h. Leave the **Address**, **Host** and **Target** fields blank to apply the configuration to all signaling servers in the Signaling Domain and have them all provide a Diameter network channel on the same port.

i. Click **Apply**.

Note: If you run multiple signaling servers on the same physical system, define each signaling server as a different Diameter node which listens on a different port. Otherwise, the Diameter SSU running on all signaling servers using the same port will result in network traffic collisions.

---

2. Deploy the R-IM-OCF module as described in the discussion on setting up R-IM-OCF in *Oracle Communications Service Broker Modules Configuration Guide*.

   In the Administration Console:

   a. In the navigation tree, expand **OCSB**.

   b. Expand **Processing Tier**, and then **Interworking Modules**.

   c. Click **IM Management**.

   d. In the **IM Management** tab, click **New**. The New dialog box appears.

   e. From the **Type** list, select **RIMOCF**.

   f. In the **Name** field, enter a module instance name. For example, rimocfro_ instance.

   g. Click **OK**.

3. Configure the R-IM-OCF module as described in the discussion on setting up R-IM-OCF in *Oracle Communications Service Broker Modules Configuration Guide*.

4. If you want to activate the Diameter-based orchestration mode, in the **CallHandlingMBean** whose object name is **oracle:type=oracle.axia.cm.ConfigurationMBean,name=com.convergin.wcs.osgi.im.rocf,version=***MBean_Version***,name0=RIMOCF,name1=ModuleInstance[***ModuleInstance_Number***],name2=CallHandling**, set the **DiameterBasedOrchestrationMode** attribute to **true**.

5. Configure the Diameter SSU to accept incoming Diameter requests as described in the discussion on configuring the Diameter SSU in *Oracle Communications Service Broker Signaling Server Units Configuration Guide*.

   Specifically, add a new incoming routing rule to route incoming Ro requests to the R-IM-OCF module that you created in step 2.

   In the Administration Console:

   a. In the navigation tree, expand **OCSB**, and then **Signaling Tier**.

   b. Select **SSU Diameter**.

   c. In the **SSU Diameter** tab, select the **Routing** tab.

   d. On the bottom of the list of existing routes, click **Add**. The New dialog box appears.

   e. In the **Name** field, enter a name for the new routing rule.

   f. Click **Apply**.

g. In the **Incoming Routing Rules** subtab, in the **Priority** field, leave the default value.

h. In the **Module Instance** field, enter ssu:*r-im-ocf-module-name*.RIMOCF@*domain-id*, where *r-im-ocf-module-name* is the name you gave to the R-IM-OCF module in step 2, and *domain-id* is the name of the Processing Domain where you deployed the R-IM-OCF.

   If your deployment includes only one Processing Domain, then set *domain-id* to ocsb. For example, `rimocfro_instance.RIMOCF@ocsb`.

i. Click **Apply**.

Specify the criteria that Ro requests have to meet so that the Diameter SSU forward them to R-IM-OCF:

a. In the **SSU Diameter** tab, select the **Routing** tab.

b. In the list of existing routes, select the route for which you want to specify incoming routing criteria.

c. Click the **Incoming Routing Criteria** subtab.

d. Click **New**. The New dialog box appears.

e. In the **Name** field, enter a name for the new criteria.

   In the **Attribute** field, select the AVP whose value the Diameter SSU checks in incoming request.

   In the **Value** field, enter the value that the AVP should match for the Diameter SSU to route incoming requests into Online Mediation Controller.

f. Click **OK**.

6. Activate the R-IM-OCF module that you deployed and configured in steps 2 and 3.

   In the Administration Console:

   a. In the navigation tree, expand **OCSB**.

   b. Expand **Processing Tier** and then **Interworking Modules**.

   c. Click **IM Management**.

   d. In the **IM Management** tab, select the **R-IM-OCF** module in the table.

   e. Click **Activate**.

Connecting to an OCS through Diameter Ro

To connect Online Mediation Controller to an OCS:

1. In the Diameter SSU, configure the OCS instances as Diameter peers, as described in "Configuring the Diameter SSU" in *Oracle Communications Service Broker Signaling Server Units Configuration Guide*.

   At a minimum, it is recommended to establish connection with two peers per realm.

   In the Administration Console:

   a. In the navigation tree, expand **OCSB**, and then **Signaling Tier**.

   b. Select **SSU Diameter**.

c. In the **DIAMETER** tab, in the **Diameter Configuration** tab, in the list of existing Diameter nodes, select the node for which you want to configure peers.

d. Click the **Peers** subtab.

e. Click **New**. The **New** dialog appears.

f. In the **Address** field, enter the IP address or DNS name of the peer.

g. In the **Host** field, enter the Destination-Host AVP value identifying the peer. You will refer this value later when configuring outgoing Diameter routes.

h. In the **Port** field, enter the listen port number of the peer node

i. In the **Protocol** field, enter the protocol used to communicate with the peer: tcp or sctp.

j. Check the **Watchdog** check box if the peer supports the Diameter Tw watchdog timer interval.

k. Click **Apply**.

l. Repeat steps e through k for each OCS instance in your system.

2. In the Diameter SSU, define the OCS as a Diameter destination. You can define two or more destinations having different Destination-Host AVPs, that share the same alias, thereby adding a level of redundancy, treating all destinations as one logical destination, and balancing the load among destinations.

a. In the navigation tree, expand **OCSB**, and then **Signaling Tier**.

b. Select **SSU Diameter**.

c. In the **SSU Diameter** tab, select the **Outbound Destinations** tab.

d. Click **New**. The New dialog box appears.

e. In the **Name** field, enter a name for the OCS.

f. In the **Alias** field, enter an alias that you want to assign to the destination OCS. You can enter the same alias later when you define more destination OCSs, to have many destination OCSs share load.

g. In the **Destination Host** and **Destination Realm** fields, enter the destination host and destination realm of the peers running the OCS. Use the destination host of the peers you defined in step 1.

h. Click **OK**.

i. Repeat steps d through h for each additional destination OCS that you want to configure.

3. Deploy the IM-OCF-Ro module as described in "Managing Interworking Modules" in *Oracle Communications Service Broker Modules Configuration Guide*.

In the Administration Console:

a. In the navigation tree, expand **OCSB**.

b. Expand **Processing Tier**, and then **Interworking Modules**.

c. Click **IM Management**.

d. In the **IM Management** tab, click **New**.

e. From the **Type** list, select **IMOCF**.

f. In the **Name** field, enter a module instance name. For example, **imocfro_instance**.

g. Click **OK**.

4. Configure the IM-OCF-Ro instance as described in "Configuring IM-OCF-Ro" in *Oracle Communications Service Broker Modules Configuration Guide*.

   Specifically, you need to define the destination OCS that the IM-OCF-Ro instance communicates with. You can either specify Destination-Host and Destination-Realm AVPs, or you can use the alias of a destination that you defined in step 1.

   In the Administration Console:

   a. In the navigation tree, expand **OCSB**.

   b. Expand **Processing Tier**, and then **Interworking Modules**.

   c. Click **IM Management**.

   d. Select the **IM-OCF-Ro** module.

   e. In the **Configuration** tab, select the **Diameter Credit Control Application** tab, and then the **AVPs** tab.

   f. In the **Destination-Realm AVP** field, enter the alias that you assigned to the destination OCS that you defined in step 1. Alternatively, in the **Destination-Realm AVP** and in the **Destination-Host AVP** fields, enter the values that the IM-OCF-Ro must set in the Destination-Host and Destination-Realm AVPs of outgoing Diameter request, to route requests to the destination OCS.

   g. Click **Apply**.

5. Activate the IM-OCF-Ro module that you deployed and configured in steps 3 and 4.

   In the Administration Console:

   a. In the navigation tree, expand **OCSB**.

   b. Expand **Processing Tier** and then **Interworking Modules**.

   c. Click **IM Management**.

   d. In the **IM Management** tab, select the IM-OCF-Ro module in the table.

   e. Click **Activate**.

Connecting to BRM through PCP

To connect Online Mediation Controller to BRM using PCP:

1. Create BRM connection pools in the PCP SSU, as described in "Defining Connection Pools" in the chapter "Configuring the PCP Signaling Server Unit" in *Oracle Communications Service Broker Signaling Server Units Configuration Guide*.

   See also "About Connection Pooling" in *Oracle Communications Billing and Revenue Management System Administrator's Guide*.

2. Secure the BRM connection pools that you created in step 1, as described in "Securing Connection Pools" in the chapter "Configuring the PCP Signaling Server Unit" in *Oracle Communications Service Broker Signaling Server Units Configuration Guide*.

In the Administration Console:

a. In the navigation tree, expand **OCSB**, and then **Signaling Tier**.

b. Select **SSU PCP**.

c. In the **PCP** tab, select the **Credential Store** tab.

d. In the **Password** area, in the **Key** field, enter the ID of the connection pool that you want to secure. This should be the Pool ID that you assigned to the connection pool when you created the connection pool in step 1.

e. In the **Password** area, in the **Password** field, enter the password of the BRM client application account used by the connection pool to access BRM. This should be the password of the account that you configured in the **BRM CM Login ID** field when you initially defined the connection pool.

f. In the **Password** area, uncheck the **One-way** check box.

g. In the **Password** area, click **Set Password**.

h. Repeat steps d through f for each connection pool that you want to secure.

3. Define destination BRM applications, as described in "Defining PCP Network Entities" in the chapter "Configuring the PCP Signaling Server Unit" in *Oracle Communications Service Broker Signaling Server Units Configuration Guide*.

4. Deploy the IM-OCF-PCP module as described in "Managing Interworking Modules" in *Oracle Communications Service Broker Modules Configuration Guide*.

   In the Administration Console:

   a. In the navigation tree, expand **OCSB**.

   b. Expand **Processing Tier**, and then **Interworking Modules**.

   c. Click **IM Management**.

   d. In the **IM Management** tab, click New.

   e. From the **Type** list, select **IMOCFPCP**.

   f. In the **Name** field, enter a module instance name. For example, imocfpcp_ instance.

   g. Click **OK**.

5. Configure the IM-OCF-PCP instance as described in "Configuring IM-OCF PCP" in *Oracle Communications Service Broker Modules Configuration Guide*.

   Specifically, you need to define the destination OCS that the IM-OCF-PCP module communicates with. You can either specify Destination-Host and Destination-Realm AVPs, or you can use an alias of a destination that you defined in step 1.

   In the Administration Console:

   a. In the navigation tree, expand **OCSB**.

   b. Expand **Processing Tier**, and then **Interworking Modules**.

   c. Click **IM Management**.

   d. Select the IM-OCF-PCP module node.

   e. In the **Configuration** tab, select the **Diameter Credit Control Application** tab, and then the **AVPs** tab.

f. In the **Destination-Realm AVP** field, enter the alias that you assigned to the destination BRM connection pool that you defined in step 1. Alternatively, in the **Destination-Realm AVP** and in the **Destination-Host AVP** fields, enter the values that the IM-OCF-PCP must set in the Destination-Host and Destination-Realm AVPs of outgoing Diameter request, to route requests to the destination BRM.

g. Click **Apply**.

6. Activate the IM-OCF-PCP instance that you deployed and configured in steps 4 and 5.

   In the Administration Console:

   a. In the navigation tree, expand **OCSB**.

   b. Expand **Processing Tier** and then **Interworking Modules**.

   c. Click **IM Management**.

   d. In the **IM Management** tab, select the IM-OCF-PCP module in the table.

   e. Click **Activate**.

Connecting to ECE Using the ECE API

To connect Online Mediation Controller to ECE:

1. In the Administration Console:

   a. In the navigation tree, expand **OCSB**, and then **Signaling Tier**.

   b. Select **SSU ECE**.

   c. In the **ECE** tab, select the **Coherence** tab.

2. Populate the ECE Protocol Adapter values used to connect to the ECE OCS:

Table 4–1 ECE OCS Configuration Parameters

Name	Type	Description
Coherence Cluster Name	String	Specifies the name for the ECE cluster to join.
JMX Management read-only	Boolean	Specifies whether the MBeans exposed by this cluster node allow operations that modify run-time attributes.
Coherence Log File Name	String	Specifies the name of the Coherence log file.
Coherence Log Level	Integer	Specifies which logged messages will be output to the log destination.
Use ECE Well Known Address	Boolean	Specifies whether to use the ECE OCS Well Known Address.
Well Known Address 1 (ip:port)	String	Specifies the first Well Known Address of the ECE cluster if Multicast networking is not in use.
Well Known Address 2 (ip:port)	String	Specifies the second Well Known Address of the ECE cluster if Multicast networking is not in use.
Multicast Address (ip:port)	String	The multicast address and port of the ECE OCS.
Multicast TTL	Integer	Specifies the time to live for multicast packets.

For information about the ECE Coherence configuration values, see *Oracle Communications Elastic Charging Engine Administration Guide*.

For information about Oracle Coherence, see the *Oracle Coherence Knowledge Base Home* at:

http://coherence.oracle.com/display/COH/Oracle+Coherence+Knowledge+Base+Home

3. Select the **General** tab to set the **Request default timeout**.
4. Configure the IM-OCF-ECE instance as described in "Configuring IM-OCF ECE" in *Oracle Communications Service Broker Modules Configuration Guide*.

   Specifically, you need to define the destination OCS that the IM-OCF-ECE module communicates with. You can either specify Destination-Host and Destination-Realm AVPs, or you can use an alias of a destination that you defined in step 1.

   In the Administration Console:

   a. In the navigation tree, expand **OCSB**.
   b. Expand **Processing Tier**, and then **Interworking Modules**.
   c. Click **IM Management**.
   d. Select the **IM-OCF-ECE** module.
   e. In the **Configuration** tab, select the **Diameter Credit Control Application** tab, and then the **AVPs** tab.
   f. In the **Destination-Realm AVP** field, enter the coherence cluster name you defined in step 21. Alternatively, in the **Destination-Realm AVP** and in the **Destination-Host AVP** fields, enter the values that the IM-OCF-ECE must set in the Destination-Host and Destination-Realm AVPs of outgoing Diameter request, to route requests to ECE.
   g. Click **Apply**.
5. Activate the IM-OCF-ECE instance that you deployed and configured in steps 4 and 5.

   In the Administration Console:

   a. In the navigation tree, expand **OCSB**.
   b. Expand **Processing Tier** and then **Interworking Modules**.
   c. Click **IM Management**.
   d. In the **IM Management** tab, select the **IM-OCF-ECE** module in the table.
   e. Click **Activate**.

Adding the OCS to the Service Orchestration Chain

To route subscriber sessions to the OCS:

1. Depending on your implementation, create an appropriate orchestration logic that routes network sessions through the OCS in your system. Use the Orchestration Studio to create the orchestration logic. See *Oracle Communications Service Broker Orchestration User's Guide*.
2. Assign the orchestration logic you created in step 1 to subscribers. Use the Subscriber Provisioning API to provision the iFC source of the orchestration logic in subscribers' IfcProfileData. See "Using the Subscriber Provisioning API" in *Oracle Communications Service Broker Subscriber Store User's Guide*, for more information.

Setting Orchestrated Charging Mediation in Offline or Degraded Mode

For information on setting up orchestrated mediation in Degraded Mode, see the discussion about the orchestrated mediation in "Using Degraded Mode".

For information on setting up orchestrated mediation in Offline Mode, see the discussion about the orchestrated mediation in "Offloading Subscriber Usage".

5

Using Degraded Mode

This chapter describes how to configure and use the Degraded Mode in Oracle Communications Online Mediation Controller.

About Degraded Mode

Online Mediation Controller relies on an external online charging system (OCS), such as Oracle Communications Billing and Revenue Management (BRM) or Oracle Communications Elastic Charging Engine (ECE), to make service authorization decisions and apply charges to subscriber accounts.

Online Mediation Controller and the OCS depend on network connectivity during normal operations. Network disruption or hardware failure in the OCS may affect the ability of users to access services if charging requests cannot be processed.

Degraded mode is an Online Mediation Controller operating mode ensuring service continuity for subscribers if integrated instances of BRM or ECE become unavailable. Online Mediation Controller actively monitors the health of BRM and ECE. If the BRM or ECE becomes unavailable for any reason, Online Mediation Controller temporarily assumes the charging function and acts as a local OCS. Online Mediation Controller also provides a default RADIUS service access decision while in degraded mode.

While acting on behalf of BRM or ECE, Online Mediation Controller applies a configurable default authorization decision to incoming service requests. Online Mediation Controller also records user activity information, which it stores in the Subscriber Store in the form of charging data records (CDRs). Online Mediation Controller replays the charging records to the BRM or ECE when the instance is available again.

Degraded mode operation is transparent to the OCS. Replayed CDRs are processed by the OCS in the same manner as real-time requests. Accordingly, the BRM and ECE require no special configuration to support degraded mode.

Online Mediation Controller supports degraded mode for external systems through the following IM types:

- IM-OCF PCP
- IM-OFCF PCP
- IM-OCF ECE
- IM-OFCF ECE

About Degraded Mode Triggers

When degraded mode is enabled, Online Mediation Controller monitors the availability of the external OCS and automatically assumes the functions of the OCS in response to:

- The OCS failing to acknowledge or reply to a request during an active session.
- The OCS not replying to a configurable heartbeat check.

When degraded mode is triggered during an active session, only the affected session is transferred to degraded mode. The session is processed in degraded mode until completed. Other active sessions and new sessions (except those associated with the same user that is being handled in degraded mode) continue to be handled by the external OCS.

If in-order CDR replay is enabled in the Administration Console, any new sessions associated with a user who is already in degraded mode are handled in degraded mode as well. This ensures that the activities recorded in all CDRs for that user are replayed in the order in which they occurred, even if that user is accessing the network from different devices.

If a heartbeat check failure triggers degraded mode, Online Mediation Controller moves all active sessions to degraded mode and handles all new sessions in degraded mode.

Online Mediation Controller continues to monitor the availability of the external OCS. When the OCS becomes available again, Online Mediation Controller resumes using it for new sessions. Existing sessions continue to be processed in degraded mode until completed.

You can also trigger degraded mode manually in the Administration Console. This is useful when you know in advance of system downtime, for example, for planned maintenance.

About Configuring Degraded Mode

By default, degraded mode is disabled. That is, Online Mediation Controller does not monitor the availability of an external OCS or automatically assume the OCS role if it becomes unavailable.

To enable degraded mode, you need to configure degraded mode operation settings, along with the settings related to degraded mode in the SSU Diameter and IMs that interact with the external OCS, as described in this chapter.

Degraded mode relies on the Subscriber Store (the repository of subscriber information) for certain capabilities. For example, the Subscriber Store enables Online Mediation Controller to correlate multiple sessions initiated on different devices to a single subscriber. This allows Online Mediation Controller to replay all CDRs associated with a subscriber in order, even for sessions that were originated on different devices. Online Mediation Controller performs default charging behaviors configured in each IM when the Subscriber Store cannot be reached.

Also, for degraded mode to work properly with orchestrated mediation, Online Mediation Controller must be configured to retrieve iFC user orchestration profiles from the Subscriber Store. Degraded mode does not work if Online Mediation Controller retrieves iFCs using the LSS mechanism. See *Oracle Communications Service Broker Subscriber Store User's Guide* for information on how to set up the Subscriber Store.

The general steps for configuring degraded mode for are:

1. Configure CDR persistence. See "Configuring CDR Persistence" for more information.
2. See "Configuring the Signaling Tier for Degraded Mode", for information on configuring the signaling tier settings for the SSU for use with the local OCS. For more information, see *Oracle Communications Service Broker Signaling Server Units Configuration Guide*.
3. Configure local OCS properties, the settings applicable to the internal Online Mediation Controller component that performs the function of the external OCS during degraded mode. See "Configuring Local OCS Properties" for more information.
4. Create an IM-OCF instance for the local OCS. See "Creating and Configuring IM-OCF for the Local OCS", for more information. Additional information on creating and configure IMs can be found in *Oracle Communications Service Broker Modules Configuration Guide*.
5. Add degraded mode settings to the IM that connects to the external OCS. See "Configuring IM Degraded Mode Settings", for more information.
6. Add a branching condition to the orchestration logic for the IMs that routes messages to one or the other IM based on the degraded mode status of the session. The status is indicated in degraded mode header of the message.

   If the value of this header is true, the iFC routes the message to the IM of the local OCS. If false, it routes the message to the IM of the external OCS. See *Oracle Communications Service Broker Orchestration User's Guide*, for more information on configuring orchestration.
7. Configure the default client authentication decision when Online Mediation Controller is acting on behalf of the external OCS. See "Configuring a Default Service Access Decision" for more information.
8. Configure CDR replay settings. "Configuring Degraded Mode CDR Replay Behavior" for more information.
9. Configure external OCS monitoring. See "Configuring External OCS Monitoring" for more information.
10. Configure whether degraded mode is triggered manually. See "Triggering Degraded Mode Manually", for more information.
11. Configure manual CDR replay options if applicable. See "Replaying Charging Data Records Manually", for more information.
12. Configure service unit counters. See "Configuring Service Unit Counters" for more information.

Configuring CDR Persistence

You can use either Oracle Berkeley DB or Oracle 11g Enterprise Database for CDR storage. Install and instantiate the database before domain creation when using Oracle 11g Enterprise Database. Set the CDR persistence mechanism to your choice of database when creating a Online Mediation Controller domain.

The default persistence mechanism is Oracle Berkeley DB. To enable Berkeley DB storage, you only need to configure the file storage location. To use Oracle Database 11g, you need to change the persistence package installed in the domain and then configure the database connection settings.

The following steps provide an overview of how to configure CDR persistence. For more information on data storage, see the information on configuring data storage in the *Oracle Communications Service Broker Installation Guide*.

Using Oracle Database 11g Persistence

To use Oracle 11g Database for CDR storage, follow these steps:

1. Prepare the database for CDR storage by running this SQL script: **degraded_mode_cdr_store.sql**.

   The script is located in this directory:

   *Oracle_home***/ocsb61/admin_server/scripts/database**

2. Open the Administration Console.
3. Expand **Domain Management**, then select **Packages**.
4. Remove the existing persistence bundle from the domain by selecting this package and clicking **Uninstall**:

   oracle.ocsb.app.rcc.service.degraded_mode.persistence.bdb

5. Install the Oracle 11g Database bundle in the domain by selecting this package and clicking **Install**:

   oracle.ocsb.app.rcc.service.degraded_mode.persistence.database.jar

6. Ensure that the start level for the package matches that of this package:

   oracle.ocsb.app.rcc.service.degraded_mode.core

7. Configure the database connection for each Managed Server, as described in the *Oracle Communications Service Broker Installation Guide*.

For more information on how to perform these steps, see the *Oracle Communications Service Broker Installation Guide*.

Using Oracle Berkeley DB File-Based Persistence

The default persistence package in the domain used for CDR storage is the Berkeley DB file-based persistence package. Therefore, to implement Berkeley DB for CDR storage, you only need to configure the storage location settings for the managed servers in your domain.

See *Oracle Communications Service Broker Installation Guide* for information on configuring Berkeley DB settings.

Configuring the Signaling Tier for Degraded Mode

To use degraded mode, you need to configure a route to the local online charging server (OCS). The local OCS is the internal Online Mediation Controller component that acts as the proxy for the unavailable OCS when operating in degraded mode. When degraded mode is active, the Signaling Server Unit (SSU) directs requests to the local OCS rather than the external OCS.

To configure routing to the local OCS:

1. Start the Administration Console.
2. In the navigation tree, expand **OCSB**.
3. Expand **Signaling Tier**.
4. Click **SSU Diameter**.

5. In the **SSU Diameter** tab, click the **Outbound Destinations** subtab.
6. Click **New**.
7. In the dialog box, enter values for the following fields:
   - **Name**: A name for the local OCS definition in the configuration.
   - **Alias**: An alias for the local OCS destination, such as **localocs**.
   - **Destination Host**: The host name of the system on which the Diameter SSU runs. This host contains the local OCS. For example, **ro.server.example.com**.
   - **Destination Realm**: The destination realm for the local OCS. This should be set as the same destination realm used for the Diameter SSU in your deployment. For example, **us.example.com**.

     Note: The destination host and realm you specify in this tab must match the values defined in the **Local OCF** node located in the **Degraded Mode** configuration folder. See "Configuring Local OCS Properties", for more information.

   - **Weight**: An integer defining the relative amount of traffic sent to the destination host. This value is compared to weights specified in other destination hosts when more than one destination host is configured.
8. Click **Apply**.

The new local OCS configuration appears in the outbound destinations list.

Now configure the peer and route configuration in SSU Diameter, as follows:

1. In the navigation tree, expand **OCSB**.
2. Expand **Signaling Tier**.
3. Click **SSU Diameter**.
4. Click the **DIAMETER** tab.
5. With the **default** node selected, configure the following settings of the **General** tab:
   - **Realm**: The destination realm for the local OCS. This should be the same value as you configured for the destination realm for the local OCS, such as **us.example.com**.
   - **Host**: The host name of the system on which the local OCS runs. This should be the same value as you configured for the Destination Host for the local OCS, such as **ro.server.example.com**.

     Note: The destination host and realm you specify in this tab must match the values defined in the **Local OCF** node located in the **Degraded Mode** configuration folder. See "Configuring Local OCS Properties", for more information.

   Use the **default** node for the local OCS configuration.

   Configure other general settings for the route as needed. For more information, see the chapter on SSU Diameter in *Oracle Communications Service Broker Signaling Server Units Configuration Guide*.

6. Click the **Routes** tab.
7. Click **Add** to create a route.
8. Enter a **Name** for the route.
9. Click **Apply**.
10. Specify the following settings of the new route:
    - **Realm**: The destination realm for the local OCS. This should be the same value as you configured for the destination realm for the local OCS, such as **us.example.com**.
    - **Application ID**: The application ID of the Diameter application. Enter **4**, which represents Diameter Ro.
    - **Action**: The routing action to perform for the local OCS. Set to **relay**.
11. Click **New** under the KeyID Host table.
12. In the **Host** field, specify the host name of the system on which the local OCS runs. This should be the same value as you configured for the destination host for the local OCS, such as **ro.server.example.com**.
13. Click **Apply**.
14. Click the **Peers** tab.
15. Create a new local OCS Diameter peer as described in the chapter on configuring SSU Diameter in *Service Broker Signaling Server Units Configuration Guide*.
16. Select the check box next to your newly created peer. The peer represents the local OCS.
17. Click **Update**.
18. Specify the peer settings as follows:
    - **Host**: The host name of the system on which SSU Diameter runs. This address is configured in the **General** settings located at the following location: **OCSB**, then **Signaling Tier**, then **SSU Diameter**, then the **DIAMETER** tab, then the **Diameter Configuration** subtab. See the chapter on configuring SSU Diameter in the *Oracle Communications Service Broker Signaling Server Units Configuration Guide*, for more information
    - **Address**: The IP address SSU Diameter is configured to listen on. This address is configured in the **General** settings located at the following location: **OCSB**, then **Signaling Tier**, then **SSU Diameter**, then the **DIAMETER** tab, then the **Diameter Configuration** subtab. See the chapter on configuring SSU Diameter in the *Oracle Communications Service Broker Signaling Server Units Configuration Guide*, for more information.
    - **Port**: The port number on which the SSU Diameter listens for Diameter traffic. The default value is **3588**.
    - **Protocol**: The network protocol to use for the local OCS. Select either **tcp** or **sctp**.

    The **Watchdog** check box can remain disabled with a **FALSE** value.
19. Click **Apply**.

The signaling tier is now configured to route Diameter traffic to the local OCS.

Configuring Local OCS Properties

The local OCS settings define properties for the Online Mediation Controller component that performs the functions of the external OCS when unavailable.

Verifying the Degraded Mode Local OCS Configuration

By default, the Online Mediation Controller installation creates the degraded mode local OCS. Additional local online charging systems are created and managed in the **Local OCF Configuration** tab.

Verify that the default degraded mode local OCS is configured:

1. In the navigation tree, expand **OCSB**.
2. Expand **Processing Tier**.
3. Expand **Applications**.
4. Expand **Degraded Mode**.
5. Click **Local OCF**.
6. Ensure there is an entry for **localocs-Degraded**.

Configuring Local OCS Properties

Configure local OCS degraded mode properties as follows:

1. In the navigation tree, expand **OCSB**.
2. Expand **Processing Tier**.
3. Expand **Applications**.
4. Expand **Degraded Mode**.
5. Click **OCS**.
6. In the **Degraded mode enabled OCS** field, specify the external OCS that the local OCS supports:
   - **BRM**: Degraded mode is configured for use with Oracle Communications Billing and Revenue Management.
   - **ECE**: Degraded mode is configured for use with Oracle Communications Elastic Charging Engine
7. Click **Apply**.

Creating and Configuring IM-OCF for the Local OCS

Online Mediation Controller requires configuration of an instance of IM-OCF used with Degraded Mode.

1. Create an IM-OCF instance for the local OCS. See the chapter on setting up IM-OCF in *Oracle Communications Service Broker Modules Configuration Guide* for more information.
2. In the **Degraded Mode** tab of the IM-OCF for the local OCS, specify the following settings:
   - **CDR Mode**: Set to **ALWAYS**.
   - **Local-OCF Alias**: Set to the alias of the local OCF.

Optionally, configure other degraded mode settings for the IM as described in "Common IM Degraded Mode Settings".

3. If triggering degraded mode manually, leave the **Destination-Realm-AVP** in the **Diameter Credit Control Application** tab of the IM-OCF blank.

4. Add a branching condition to the orchestration logic for the IMs that routes messages to one or the other IM based on the degraded mode status of the session. The orchestration logic branching condition is required if you trigger degraded mode manually.

   Example 5–1 shows a sample orchestration IFC configuration rule file. In this example, The Degraded Mode status is indicated in the **x-wcs-degraded-mode** header of the message. If the value of this header is true, the iFC routes the message to the IM of the local OCS. If false, the iFC routes the message to the IM of the external OCS.

   For more information on configuring orchestration, see *Oracle Communications Service Broker Orchestration User's Guide*.

Example 5–1 Sample Orchestration IFC Configuration

```
<IFCs>
            <InitialFilterCriteria>
                <Priority>0</Priority>
                <TriggerPoint>
                    <ConditionTypeCNF>0</ConditionTypeCNF>
                    <SPT>
                        <ConditionNegated>0</ConditionNegated>
                        <Group>0</Group>
                        <SIPHeader>
                            <Header>x-wcs-degraded-mode</Header>
                            <Content>true</Content>
                        </SIPHeader>
                    </SPT>
                </TriggerPoint>
                <ApplicationServer>
                    <ServerName>imocf.IMOCF@ocsb</ServerName>
                    <DefaultHandling>1</DefaultHandling>
                    <Extension>
                    </Extension>
                </ApplicationServer>
                          </InitialFilterCriteria>
                          <InitialFilterCriteria>
                <Priority>1</Priority>
                <TriggerPoint>
                    <ConditionTypeCNF>0</ConditionTypeCNF>
                    <SPT>
                        <ConditionNegated>1</ConditionNegated>
                        <Group>0</Group>
                        <SIPHeader>
                            <Header>x-wcs-history</Header>
                            <Content>id=0;.*</Content>
                        </SIPHeader>
                    </SPT>
                </TriggerPoint>
                <ApplicationServer>
                    <ServerName>imocfpcp.IMOCFPCP@ocsb</ServerName>
                    <DefaultHandling>1</DefaultHandling>
                    <Extension>
                    </Extension>
```

```
        </ApplicationServer>
    </InitialFilterCriteria>
</IFCs>
```

Configuring IM Degraded Mode Settings

To configure degraded mode for orchestrated mediation:

1. Configure the degraded mode settings in each OCS IM deployed in your system. To access the degraded mode settings in the IM:
   a. In the Processing Tier tree, expand **Interworking Modules**.
   b. Click the IM instance of the external OCS for which you want to enable degraded mode.
   c. In the **Configuration** tab, click the **Degraded Mode** subtab.
   d. Configure the degraded mode settings shown in "Common IM Degraded Mode Settings" as appropriate for the external OCS. For information on the IM configuration, see *Oracle Communications Service Broker Modules Configuration Guide*.

Common IM Degraded Mode Settings

The IMs that support degraded mode have common configuration settings related to degraded mode operation. This section provides an overview of the common settings. For more information about a particular type of IM, see *Oracle Communications Service Broker Modules Configuration Guide*.

The common degraded mode IM settings are:

- **On OCF Failure**: Specifies the module behavior when there is an Online Charging Function (OCF) Failure. This setting is overwritten by the OCF AVP settings for **Credit Control Failure Handling** and **Realtime Required**.
  - **ALWAYS_REFUSE**: Use when degraded mode is disabled. Online Mediation Controller writes no CDRs to storage.
  - **USE_LOCAL_REFUSE**: Use when degraded mode is activated. Online Mediation Controller writes CDRs to the Subscriber Store for replay at a later time. If the Subscriber Store is unavailable, service is refused.
  - **USE_LOCAL_GRANT**: Use when degraded mode is activated. Online Mediation Controller writes CDRs to the Subscriber Store for replay at a later time. If the Subscriber Store is unavailable, permit service.
- **CDR Mode**: The mode in which Online Mediation Controller writes CDRs to local storage, from these options:
  - **Normal**: Online Mediation Controller writes CDRs only after it assumes the functions of the external OCS.
  - **History**: After it assumes the functions of the external OCS, Online Mediation Controller writes CDRs that reflect the history of each active session, including for requests received before assuming the active OCS role.
  - **Always**: Online Mediation Controller writes CDRs always.
- **CDR Writer Impl**: The internal class that performs CDR writing. This can be one of the following values:

  - **oracle.ocsb.app.rcc.ocfproxy.cdrwriter.CDRWriterImpl**: Writes CDRs to the degraded mode service. This is the standard implementation to use for degraded mode processing.
  - **com.convergin.common.diameter.ocfproxy.CDRWriterLogImpl**: Writes information to the log only. This is useful for testing.

- **CDR Writer Service** The CDR writer service that Online Mediation Controller uses. Enter **oracle.ocsb.app.rcc.service.dmode.DegradedModeService** to use the default service. If no value is provided, NULL will be passed.
- **Degraded Mode Timer**: The period, in milliseconds, that Online Mediation Controller waits for a response from the online charging server. If the online charging server does not respond within the specified period, the user session is switched to degraded mode.
- **Local-OCF Alias**: Specifies the alias of the local OCS. This alias is mapped to the destination host and destination realm of the local online charging server as defined in the configuration of Diameter SSU outbound routing rules. See the discussion of Diameter SSU in *Oracle Communications Service Broker Signaling Server Units Configuration Guide* for more information.
- **Local-OCF External Protocol**: Specifies the protocol that Online Mediation Controller uses to communicate with the local online charging server.
- **Degraded Mode Error Codes**: The error response code from the external OCS that indicates a failure for which degraded mode should be activated. Specify as comma-delimited integers.

  For example:

  5012,5023

Configuring a Default Service Access Decision

Online Mediation Controller can perform the role of a RADIUS server, applying service authorization and authentication decisions to incoming requests. Online Mediation Controller acts as the access enforcement point for network services.

When presented with a user request for a service, Online Mediation Controller retrieves the service authorization information for a given user from the external OCS. If the user is authorized to access the service, Online Mediation Controller also retrieves that user's credentials from the OCS. Online Mediation Controller uses the credentials to validate those in the incoming request and permits or denies access to a service for the user based on the results.

You can configure a default RADIUS authentication decision for Online Mediation Controller to apply when it is in degraded mode.

To configure the default credit authorization decision:

1. In the navigation tree, expand **OCSB**.
2. Expand **Processing Tier**.
3. Click **RADIUS Mediation**.
4. In the **degraded-mode-behavior** field, enter one of the following values:
   - **reject**: Blocks all requests.
   - **accept**: Permits all requests.

- **discard**: Silently discards all requests, which the client typically perceives as a server unavailable error.

5. Click **Apply**.

Configuring Degraded Mode CDR Replay Behavior

The CDR replay configuration settings determine how Online Mediation Controller replays CDRs when an external OCS becomes available again.

This section describes how to configure properties associated with CDR replay. See "Replaying Charging Data Records Manually" for information about manually replaying CDRs.

To configure CDR replay behavior:

1. Start the Administration Console.
2. In the navigation tree, expand **OCSB**.
3. Expand **Processing Tier**.
4. Expand **Applications**.
5. Expand **Degraded Mode**.
6. Click the **Replay** node.
7. In the **Control** tab, set the **Enable Manual Replay** field to either of the following values:
   - **TRUE**: Disables automatic CDR replay. In this case, a Online Mediation Controller administrator must manually initiate the CDR replay process when an OCS resumes availability.
   - **FALSE**: Enables automatic CDR replay. In this case, Online Mediation Controller triggers CDR replay when it detects that an OCS has resumed availability.

     The default value is **FALSE**.
8. Click the **Degraded CDR Replay** tab.
9. In the **Degraded CDR Replay** pane, configure these settings:
   - **Enable In-Order replay**: If **TRUE**, Online Mediation Controller ensures that outstanding requests are replayed to the external OCS in the order in which they occurred. With in-order replay, Online Mediation Controller takes into account requests in new sessions for the same subscriber. That is, while there are any pending CDRs for a given subscriber, all new sessions for that subscriber are handled in degraded mode to ensure in-order processing of CDRs across devices. The default value is **FALSE**.
   - **Replay rate**: To avoid over-burdening the OCS when it restarts, you can control the rate at which Online Mediation Controller submits CDRs to the external OCS using this attribute. This value determines the number of CDRs replayed per second after the OCS resumes operation. The default value is **10000**.
   - **Replay Trigger Interval**: The minimum amount of time, in milliseconds, between replay trigger events. You can use this value to pace the replay of CDRs to ensure that the external OCS is not overwhelmed with requests when its service is resumed. The default value is **2000**.

- **Replay Trigger Delay**: The time to wait, in milliseconds, before beginning CDR replay after the OCS returns to online mode. This value is respected only when **Enable Manual Replay** is set to false. The default value is **2000**.
- **Replay Failure Threshold**: The number of timeout events during CDR replay after which Online Mediation Controller pauses CDR replay. It resumes CDR replay after the configured interval. The default value is **5**.
- **Max number of CDRs to fetch**: The maximum number of CDRs retrieved from the data store. The default value is **1000**.
- **Delete CDRs**: Whether Online Mediation Controller should delete the CDRs from the CDR store after they have been successfully replayed. The default value is **FALSE**.

10. Click **Apply** to save your changes.

Configuring External OCS Monitoring

Online Mediation Controller monitors an external OCS to determine its availability. You can configure the following aspects of how Online Mediation Controller monitors the OCS:

- Heartbeat Interval
- Heartbeat Failure Threshold

The **Heartbeat Interval** determines how frequently Online Mediation Controller sends a health check message to the external OCS.

To configure the heartbeat check interval, follow these steps:

1. In the navigation tree, expand the **OCSB** node.
2. Expand **Processing Tier**.
3. Expand **Applications**.
4. Expand **Degraded Mode**.
5. Click **Replay**.
6. In the configuration pane, click the **Heartbeat** tab.
7. In the **Heartbeat Interval** field, enter how frequently, in milliseconds, the degraded mode application checks the health of the external OCS. The default is **10000** milliseconds.
8. Configure the **Heartbeat Timeout** value.

   The Heartbeat Timeout value indicates the interval in milliseconds to wait for the heartbeat response. If the response is not received within this time, a heartbeat failure is indicated. You should set this value less than the Heartbeat Interval value.
9. Configure the **Heartbeat Failure Threshold**.

   The Heartbeat Failure Threshold value indicates the number of missed heartbeat intervals that trigger degraded mode. In some high-availability configurations the failure of an OCS node may cause a single missed heartbeat before another OCS node assumes processing of charging requests. In such cases, the heartbeat failure threshold should be set to greater than 1. The default value is **2**.

To configure the heartbeat failure threshold:

1. In the navigation tree, expand the **OCSB** node.

2. Expand **Processing Tier**.
3. Expand **Applications**.
4. Expand **Degraded Mode**.
5. Click **Replay**.
6. In the configuration pane, click the **Heartbeat** tab.
7. In the **Heartbeat Failure Threshold** field, enter the number of heartbeat check cycle failures before degraded mode is triggered. The default value is **2**.

Triggering Degraded Mode Manually

To manually trigger degraded mode, perform the following steps:

1. In the navigation tree, expand **OCSB**.
2. Expand **Processing Tier**.
3. Expand **Applications**.
4. Expand **DegradedMode**.
5. Select **Replay**.
6. In the **Control** tab, set the **Enable Manual Degraded Mode** value to **TRUE**.

   Setting this mode enables degraded mode to be activated manually.

   If **FALSE**, degraded mode can only be activated automatically, that is, in response to network events.
7. Click **Apply**.

After enabling manual degraded mode triggering, you can trigger degraded mode by invoking the **markSystemStatusAsDegraded** operation in the following runtime MBean:

oracle.ocsb.app.rcc.service.dmode.mbeanDegradedModeMBean

To access the runtime MBean, connect to the managed server JMX process using any JMX client software.

Replaying Charging Data Records Manually

By default, Online Mediation Controller replays CDRs automatically when it detects OCS recovery. Alternatively, you can disable automatic CDR replays, and instead invoke CDR replay manually when needed.

Configure the CDR replay behavior by performing the following steps:

1. In the navigation tree, expand **OCSB**.
2. Expand **Processing Tier**.
3. Expand **Applications**.
4. Expand **DegradedMode**.
5. Select **Replay**.
6. In the **Control** tab, set the **Enable Manual Replay** to one of the following values:
   - **TRUE**: Disables automatic CDR replay and enables manual replay.
   - **FALSE**: Enables automatic CDR replay.

7. Click **Apply**.

After enabling manual CDR replay, you can trigger CDR replay using the **startManualReplay** operation in the following runtime MBean:

oracle.ocsb.app.rcc.service.dmode.mbeanDegradedModeMBean

The operation does not replay all outstanding CDRs. It only replays the number of CDRs configured as the maximum number of CDRs to fetch in the replay configuration (1000, by default). To automatically replay all outstanding CDRs, you must create an MBean script that repeatedly invokes the **startManualReplay** operation.

See "Configuring Degraded Mode CDR Replay Behavior" for information on the maximum number of CDRs to fetch setting.

Configuring Service Unit Counters

Counters in the Online Mediation Controller configuration comprise the network service units that you can adapt to your Diameter application requirements.

By default, the predefined counters are based upon the requested service in the Diameter request. Specifically, they are based on the content of the REQUESTED-SERVICE-UNITS AVP, which can be:

- CC-Time is mapped to code 420
- CC-Total-Octets is mapped to code 421
- CC-Input-Octets is mapped to code 412
- CC-Output-Octets is mapped to code 414
- CC-Service-Specific-Units is mapped to code 417

To modify the default unit counter configuration:

1. In the navigation tree, expand **OCSB**.
2. Expand **Processing Tier**.
3. Expand **Applications**.
4. Expand **Degraded Mode**.
5. Select **LocalOCF**.
6. In the **General** tab, verify the implementation class that performs service unit calculations. The default is **oracle.ocsb.app.localocf.ruleengine.UnitCalculatorBasicImpl**. You should modify this value only if you have implemented a custom unit calculator.
7. Click the **Counters** tab.
8. In the Counters pane, modify the default counter configuration by selecting a counter from the list and clicking the **Update** button.
9. In the Update dialog box, enter values for these fields:
   - **type**: The AVP code number that represents the counter type.
   - **value**: The initial unit value for this counter type.
10. Click **OK**.

6

Offloading Subscriber Usage

This chapter describes the request offload functionality in Oracle Communications Online Mediation Controller.

About Offloading Subscriber Usage

Online Mediation Controller responds to offloaded subscriber authorization and charging requests without redirecting sessions to Oracle Communications Billing and Revenue Management (BRM). Online Mediation Controller assumes the role of the external online charging system (OCS) and responds with a default authorization for users deemed safe for offloading. Online Mediation Controller generates CDRs for the offloaded sessions for replay at a later time back to BRM to account for the offloaded sessions.

Service Providers configure the types of users or traffic types eligible for offloading. Offloading charging requests ensures that the OCS is not overwhelmed during peak hours. Online Mediation Controller delays sending offloaded CDRs to the OCS for rating until configurable off-peak hours.

Online Mediation Controller can compress CDRs from the same subscriber into a single record before playback. Doing so further increases the efficiency of rating offloaded subscriber usage.

Offline Mode operates independently from degraded mode. Each mode is configured to use its own instance of a local OCS.

About Configuring Offloading and CDR Replay

To enable offloading, you need to configure Online Mediation Controller settings including Signaling Server Unit (SSU) and Interworking Modules (IMs) components.

Offloading relies on the Subscriber Store, the Online Mediation Controller repository of end user information, for certain capabilities. For example, the Subscriber Store enables Online Mediation Controller to correlate multiple sessions initiated on different devices to a single end user. This allows Online Mediation Controller to replay all CDRs associated with the end user in order, even for sessions that were originated on different devices. Subscriber Store data, such as subscribed services, credit limits, and usage counters can also be used to determine offload eligibility.

Offloading supports orchestrated mediation deployments. Incoming Diameter Ro requests are checked by the Orchestration Engine (OE) to determine if they are offload eligible. The OE sends qualifying requests to the offloading local OCS. Orchestrated mediation of offloaded requests can include additional applications if required by your business logic.

Configuring Online Mediation Controller for Offloading

Offloading requires orchestrated mediation, in which charging-related traffic is handled by the OE and applicable IMs. The OE determines if a request is offload eligible by inspecting the request header or querying the Subscriber Store for a subscriber's resource balance.

Configuring orchestrated mediation for offloading involves the following general steps:

- Adding offloading settings to the IM that connects to the BRM instance
- Creating an IM-OCF instance for the offloading local OCS
- Adding a branching condition in the orchestration logic for the IMs for routing qualifying requests to the offloading local OCS.

See "Setting Up Orchestrated Charging Mediation", for more information on orchestrated mediation. For additional information on creating and configuring IMs, see *Oracle Communications Service Broker Modules Configuration Guide*.

The following procedure provides an overview of the orchestrated mediation configuration for offloading.

1. Configure CDR persistence. See "Configuring CDR Persistence" for more information.
2. Configure SSU settings for the local OCS (as described in "Configuring the Signaling Tier for Offloading") and for the external OCS. For more information on SSU configuration, see *Oracle Communications Service Broker Signaling Server Units Configuration Guide*.
3. Add offload settings to the IM instance that mediates requests to the external OCS. See "Configuring IM Offloading Settings", for more information.
4. Create an offloading local OCS. See "Configuring Local OCS Properties", for more information.
5. Create an IM-OCF instance for the local OCS used with offloading. See "Creating and Configuring IM-OCF for the Offloading Local OCS", for more information.
6. Add a branching condition to the orchestration logic for the IMs that routes messages to one or the other IM based on the offload eligibility of the session. The status is indicated in offload mode header of the message.

   If the value of this header is true, the iFC should route the message to the IM of the local OCS. If false, it should route the message to the IM of the external OCS. See *Oracle Communications Service Broker Orchestration User's Guide*, for more information on configuring orchestration.
7. Configure the offloaded CDR replay behavior. See "Configuring Offloaded CDR Replay Behavior", for more information. See "Replaying Charging Data Records Manually", if you want to perform manual CDR replay.
8. Configure the offloaded CDR compression if needed. See "Configuring Offloaded CDR Compression", for more information.
9. Configure service unit counters. See "Configuring Service Unit Counters" for more information.

Configuring CDR Persistence

You can use either Oracle Berkeley DB or Oracle 11g Database for offloaded CDR storage. The default persistence mechanism is Oracle Berkeley DB storage. To enable Berkeley DB storage, you only need to configure the file storage location. To use Oracle Database 11g, you need to change the persistence package installed in the domain and then configure the database connection settings.

CDRs generated by both degraded mode and offloading reside in the same database. See "Using Degraded Mode", for more information on the degraded mode.

The following steps provide an overview of how to configure CDR persistence. For more information on data storage, see the information on configuring data storage in the *Oracle Communications Service Broker Installation Guide*.

Using Oracle Database 11g Persistence

To use Oracle 11g Database for offloaded CDR storage, follow these steps:

1. Prepare the database for CDR storage by running this SQL script: **degraded_mode_cdr_store.sql**.

   The script is located in the following directory:

   *Oracle_home***/ocsb61/admin_server/scripts/database**

2. Open the Administration Console.
3. Select **Domain Management**, then **Packages**.
4. Remove the existing persistence package from the domain by selecting this package and clicking **Uninstall**:

   oracle.ocsb.app.rcc.service.degraded_mode.persistence.bdb

5. Install the database package in the domain selecting this file and clicking **Install**:

   oracle.ocsb.app.rcc.service.degraded_mode.persistence.database.jar

6. Ensure that the start level for the package matches that of this package:

   oracle.ocsb.app.rcc.service.degraded_mode.core

7. Configure the database connection for each Managed Server, as described in the *Oracle Communications Service Broker Installation Guide*.

For more information on how to perform these steps, see the *Oracle Communications Service Broker Installation Guide*.

Using Oracle Berkeley DB File-Based Persistence

The default persistence package used for offloaded CDR storage is the Oracle Berkeley DB file-based persistence package. Therefore, to implement Oracle Berkeley DB for CDR storage, you only need to configure the storage location settings for the managed servers in your domain.

See *Oracle Communications Service Broker Installation Guide* for information on configuring Berkeley DB settings.

Configuring the Signaling Tier for Offloading

To use offloading, you need to configure a route to the offloading local online charging server (OCS). The local OCS is the internal Online Mediation Controller component that responds to requests from offloaded subscribers. When offloading is active, the

Signaling Server Unit (SSU) directs requests to the local OCS rather than the external OCS.

To configure routing to the offloading local OCS:

1. In the navigation tree, expand **OCSB**.
2. Expand **Signaling Tier**.
3. Click **SSU Diameter**.
4. In the **SSU Diameter** tab, click the **Outbound Destinations** subtab.
5. Click **New**.
6. In the dialog box, enter values for the following fields:
   - **Name**: A name for the local OCS definition in the configuration.
   - **Alias**: An alias for the local OCS destination, such as **localocs**.
   - **Destination Host**: The host name of the system on which the Diameter SSU runs. This host contains the local OCS. For example, **ro.server.example.com**.
   - **Destination Realm**: The destination realm for the local OCS. This should be set as the same destination realm used for the Diameter SSU in your deployment. For example, **us.example.com**.

     Note: The destination host and realm you specify in this tab must match the values defined in the **Local OCF** node located in the **Degraded Mode** configuration folder for the offloading local OCS. See "Configuring Local OCS Properties", for more information.

   - **Weight**: An integer defining the relative amount of traffic sent to the destination host. This value is compared to weights specified in other destination hosts when more than one destination host is configured.
7. Click **Apply**.

The new local OCS configuration appears in the outbound destinations list.

Now configure the peer and route configuration in SSU Diameter, as follows:

1. In the navigation tree, expand **OCSB**.
2. Expand **Signaling Tier**.
3. Click **SSU Diameter**.
4. Click the **DIAMETER** tab.
5. With the **default** node selected, configure the following settings of the **General** tab:
   - **Realm**: The destination realm for the local OCS. This should be the same value as you configured for the destination realm for the local OCS, such as **us.example.com**.
   - **Host**: The host name of the system on which the local OCS runs. This should be the same value as you configured for the Destination Host for the local OCS, such as **ro.server.example.com**.

Note: The destination host and realm you specify in this tab must match the values defined in the **Local OCF** node located in the **Degraded Mode** configuration folder. See "Configuring Local OCS Properties", for more information.

Use the **default** node for the local OCS configuration.

Configure other general settings for the route as needed. For more information, see the chapter on SSU Diameter in *Oracle Communications Service Broker Signaling Server Units Configuration Guide*.

6. Click the **Routes** tab.
7. Click **Add** to create a route.
8. Enter a **Name** for the route.
9. Click **Apply**.
10. Specify the following settings of the new route:
    - **Realm**: The destination realm for the local OCS. This should be the same value as you configured for the destination realm for the local OCS, such as **us.example.com**.
    - **Application ID**: The application ID of the Diameter application. Enter **4**, which represents Diameter Ro.
    - **Action**: The routing action to perform for the local OCS. Set to **relay**.
11. Click **New** under the KeyID Host table.
12. In the **Host** field, specify the host name of the system on which the local OCS runs. This should be the same value as you configured for the destination host for the local OCS, such as **ro.server.example.com**.
13. Click **Apply**.
14. Click the **Peers** tab.
15. Select the check box next to your newly created peer. The peer represents the local OCS.
16. Click **Update**.
17. Specify the peer settings as follows:
    - **Host**: The host name of the system on which SSU Diameter runs. This address is configured in the **General** settings located at the following location: **OCSB**, then **Signaling Tier**, then **SSU Diameter**, then the **DIAMETER** tab, then the **Diameter Configuration** subtab. See the chapter on configuring SSU Diameter in the *Oracle Communications Service Broker Signaling Server Units Configuration Guide*, for more information
    - **Address**: The IP address SSU Diameter is configured to listen on. This address is configured in the **General** settings located at the following location: **OCSB**, then **Signaling Tier**, then **SSU Diameter**, then the **DIAMETER** tab, then the **Diameter Configuration** subtab. See the chapter on configuring SSU Diameter in the *Oracle Communications Service Broker Signaling Server Units Configuration Guide*, for more information.
    - **Port**: The port number on which the SSU Diameter listens for Diameter traffic. The default value is **3588**.

Note: The host and realm you specify in this screen must match the values defined in the **Local OCF** node located in the **Degraded Mode** configuration folder. See "Configuring Local OCS Properties", for more information.

- **Protocol**: The network protocol to use for the local OCS. Select either **tcp** or **sctp**.

The **Watchdog** check box can remain disabled with a **FALSE** value.

18. Click **Apply**.

The signaling tier is now configured to route Diameter traffic to the local OCS.

Configuring IM Offloading Settings

Specify the following settings for each IM you want to enable for offloading.

To access the offloading settings in the IM:

1. In the **Processing Tier** tree, expand **Interworking Modules**.
2. Click the IM of the BRM instance for which you want to enable offloading.
3. Click the **Configuration** tab.
4. Click the **Degraded Mode** tab.
5. Configure the degraded mode settings shown in "Common IM Offloading Settings" as appropriate for the external OCS. For information on the IM configuration, see *Oracle Communications Service Broker Modules Configuration Guide*.

Configuring Local OCS Properties

The local OCS settings define properties for the Online Mediation Controller component that responds to offloaded subscriber requests. Create a new local OCS, separate from the degraded mode local OCS, to which the OE routes eligible offloaded requests to.

Creating the Offloading Local OCS

Additional local online charging systems are created and managed in the **Local OCF Configuration** tab.

Create the Offline Mode local OCS as follows:

1. In the navigation tree, expand **OCSB**.
2. Expand the **Processing Tier**.
3. Expand **Applications**.
4. Expand **Degraded Mode**.
5. Click **Local OCF**.
6. In the **Local OCF Configuration** tab, Click **New**.
7. Configure the parameters as follow:
   - **Name**: The name of the local OCS.

- **Mode**: The operating mode of the OCS. For a local OCS used with Offline Mode, select **CDR**.
- **unitCalculatorObjectName**: The implementation class that performs service unit calculations. The default is **oracle.ocsb.app.localocf.ruleengine.UnitCalculatorBasicImpl**.

8. Click **Apply**.
9. Commit the changes to the domain.

Common IM Offloading Settings

The IMs that support offloading have common configuration settings. This section provides an overview of the common settings. For more information about a particular type of IM, see *Oracle Communications Service Broker Modules Configuration Guide*.

The common offloading IM settings are:

- **On OCF Failure**: Specifies the module behavior when there is an OCF Failure. This setting is overwritten by the OCF AVP settings for **Credit Control Failure Handling** and **Realtime Required**.
  - **ALWAYS_REFUSE**: Use when offloading is disabled. No CDRs will be written to storage.
  - **USE_LOCAL_REFUSE**: Use when offloading is configured. CDRs will be written to the Subscriber Store for replay at a later time. If the Subscriber Store is unavailable, service is refused.
  - **USE_LOCAL_GRANT**: Use when offloading is configured. CDRs will be written to the Subscriber Store for replay at a later time. If the Subscriber Store is unavailable, permit service.
- **CDR Mode**: The mode in which Online Mediation Controller writes CDRs to local storage, from these options:
  - **Normal**: Online Mediation Controller writes CDRs only after it assumes the functions of the external OCS.
  - **History**: After it assumes the functions of the external OCS, Online Mediation Controller writes CDRs that reflect the history of each active session, including for requests received before assuming the active OCS role.
  - **Always**: Online Mediation Controller writes CDRs always.
- **CDR Writer Impl**: The internal class that performs CDR writing. This can be one of the following values:
  - **oracle.ocsb.app.rcc.ocfproxy.cdrwriter.CDRWriterImpl**: Writes CDRs to the Offline Mode service. This is the standard implementation to use for Offline Mode processing.
  - **com.convergin.common.diameter.ocfproxy.CDRWriterLogImpl**: Writes information to the log only. This is useful for testing.
- **CDR Writer Service** The CDR writer service that Online Mediation Controller uses, if not the default. This field should remain blank, as it is by default.
- **Degraded Mode Timer**: The period, in milliseconds, that Online Mediation Controller waits for a response from the online charging server. If the online charging server does not respond within the specified period, the user session is switched to degraded mode.

- **Local-OCF Alias**: Specifies the alias of the offloading local OCS. This alias is mapped to the destination host and destination realm of the local online charging server as defined in the configuration of Diameter SSU outbound routing rules. See the discussion of Diameter SSU in *Oracle Communications Service Broker Signaling Server Units Configuration Guide* for more information.
- **Local-OCF External Protocol**: Specifies the protocol that Online Mediation Controller uses to communicate with the local online charging server. Default value: Ro
- **Degraded Mode Error Codes**: The error response code from the external OCS that indicates a failure for which degraded mode should be activated. Specify as comma-delimited integers.

  For example:

  5012,5023

Creating and Configuring IM-OCF for the Offloading Local OCS

Online Mediation Controller requires an instance of IM-OCF to route offloaded requests to the offloading local OCS.

1. Create an IM-OCF instance for the local OCS. See the chapter on setting up IM-OCF in *Oracle Communications Service Broker Modules Configuration Guide* for more information.
2. In the **Degraded Mode** tab of the IM-OCF for the local OCS, specify the following settings:
   - **CDR Mode**: Set to **ALWAYS**.
   - **Local-OCF Alias**: Set to the alias of the local OCF.

   Optionally, configure other degraded mode settings for the IM (as described in "Common IM Offloading Settings").

Configuring Offloaded CDR Replay Behavior

The CDR replay configuration settings determine how Online Mediation Controller replays CDRs generated by offloaded subscribers.

This section describes how to configure properties associated with CDR replay. See "Replaying Charging Data Records Manually" for information about how to perform manual CDR replay.

To configure CDR replay behavior:

1. In the navigation tree, expand **OCSB**.
2. Expand **Processing Tier**.
3. Expand **Applications**.
4. Expand **Degraded Mode**.
5. Click **Replay**.
6. In the **Control** tab, set the **Enable Manual Replay** field to either of the following values:
   - **TRUE**: Disables automatic CDR replay. In this case, a Online Mediation Controller administrator must manually initiate the CDR replay process when an OCS resumes availability.

- **FALSE**: Enables automatic CDR replay. In this case, Online Mediation Controller triggers CDR replay when it detects that an OCS has resumed availability.

7. Click the **Offloaded CDR Replay** tab.
8. In the **Offloaded CDR Replay** pane, configure these settings:
   - **Enable manual replay**: If **true**, offloaded CDRs are replayed only on receipt of a manual replay start trigger. If manual replay is selected the below start and stop time settings are disabled.
   - **Start Hour**: The hour part of the time of the day to start replay of offloaded CDRs. Valid values are from 0 to 23
   - **Start Minute**: The minute part of the time of the day to stop replay of offloaded CDRs. Valid values are from 0 to 59.
   - **Stop Hour**: The hour part of the time of the day to stop replay of offloaded CDRs. Valid values are from 0 to 23.
   - **Stop Minute**: The minute part of the time of the day to stop replay of offloaded CDRs. Valid values are from 0 to 59.
   - **Enable CDR Compression**: Whether Online Mediation Controller should compress CDRs for the same subscriber before replaying to the external OCS.
   - **Enable Compression across sessions**: If **true**, CDRs are compressed across sessions. This setting is enabled only if the **Enable CDR Compression** value is set to **true**.
   - **Replay Rate**: Number of compressed CDRs to replay per second.
   - **Delete Offloaded CDRs**: If set to **TRUE**, Online Mediation Controller deletes offloaded CDRs after successful replay.
9. Click **Apply** to save your changes.

Replaying Charging Data Records Manually

By default, Online Mediation Controller replays CDRs automatically. Alternatively, you can disable automatic CDR replays, and instead invoke CDR replay manually when needed.

Configure the CDR replay behavior by performing the following steps:

1. In the navigation tree, expand **OCSB**.
2. Expand **Processing Tier**.
3. Expand **Applications**.
4. Expand **DegradedMode**.
5. Select **Replay**.
6. Select the **Offloaded CDR Replay** tab.
7. In the **Replay parameters** tab, set the **Enable manual replay** to one of the following values:
   - **TRUE**: Disables automatic CDR replay and enables manual replay.
   - **FALSE**: Enables automatic CDR replay.
8. Click **Apply**.

After enabling manual CDR replay, you can trigger CDR replay using the **startManualReplay** operation in the following runtime MBean:

oracle.ocsb.app.rcc.service.dmode.mbeanDegradedModeMBean

Configuring Offloaded CDR Compression

Online Mediation Controller can compress offloaded CDRs before replay to the external OCS. Compression of CDRs is based on one or more matching attribute value pairs (AVPs) configured in the Administration Console. Online Mediation Controller compresses individual CDRs with matching AVP values into a single cumulative usage or charge CDR when compression is enabled.

To configure an AVP for compression:

1. In the navigation tree, expand **OCSB**.
2. Expand **Processing Tier**.
3. Expand **Applications**.
4. Expand **Degraded Mode**.
5. Click **Replay**.
6. Click the **Offloaded CDR Replay** tab.
7. Click the **Avps to Match for CDR compression** tab.
8. Click **New**.
9. In the AVP pop-up window, configure this settings:
   - **AVP code**: Specify the AVP code Online Mediation Controller uses to identify compressible CDRs.
10. Click **Apply** to save your changes.
11. Commit your changes.

Configuring Service Unit Counters

Counters in the Online Mediation Controller configuration comprise the network service units that you can adapt to your Diameter application requirements.

By default, the predefined counters are based upon the requested service in the Diameter request. Specifically, they are based on the content of the REQUESTED-SERVICE-UNITS AVP, which can be:

- CC-Time is mapped to code 420
- CC-Total-Octets is mapped to code 421
- CC-Input-Octets is mapped to code 412
- CC-Output-Octets is mapped to code 414
- CC-Service-Specific-Units is mapped to code 417

To modify the default unit counter configuration:

1. In the navigation tree, expand **OCSB**.
2. Expand **Processing Tier**.
3. Expand **Applications**.
4. Expand **Degraded Mode**.

5. Select **LocalOCF**.
6. In the **General** tab, verify the implementation class that performs service unit calculations. The default is **oracle.ocsb.app.localocf.ruleengine.UnitCalculatorBasicImpl**. You should modify this value only if you have implemented a custom unit calculator.
7. Click the **Counters** tab.
8. In the Counters pane, modify the default counter configuration by selecting a counter from the list and clicking the **Update** button.
9. In the Update dialog box, enter values for these fields:
   - **type**: The AVP code number that represents the counter type.
   - **value**: The initial unit value for this counter type.
10. Click **OK** to save your configuration changes.

7

Using the Event Notification Framework

The Event Notification Framework provides a way for complementary applications to publish events to external Operational Support Systems (OSSs).

This chapter describes the configuration of the Event Notification Framework in Oracle Communications Online Mediation Controller.

About the Event Notification Framework

Complementary applications use the Event Notification Framework to publish events. The type of event and its timing are application-specific.

Events can be targeted to internal Online Mediation Controller components, or to external systems, such as your OSS.

Complementary applications fire events toward external systems, to inform them of events that occur during application execution. For example, when a subscriber's state changes, or when a subscriber's monthly usage reaches a threshold.

Complementary applications fire events toward internal Online Mediation Controller components, to request them to perform an action. For example, request sending a text message to a subscriber using the Short Message Service (SMS), or to change the subscriber's state.

Firing events toward internal Online Mediation Controller components is necessary for optimization purposes; even though complementary applications can send requests to other Online Mediation Controller components internally, applications running on the session's setup path should optimally delegate requests, through the Event Notification Framework, to other entities which will asynchronously invoke execution of the requests, thus reducing the execution time of the complementary applications running on the session setup path.

When targeting events at internal Online Mediation Controller components you need to run a built-in Event Processor. See "About the Event Processor" for more information.

When targeting events to external systems, such as your OSS, you use an asynchronous Web services Event Notification API to consume these events in your system. See "Using the Event Notification API" for more information.

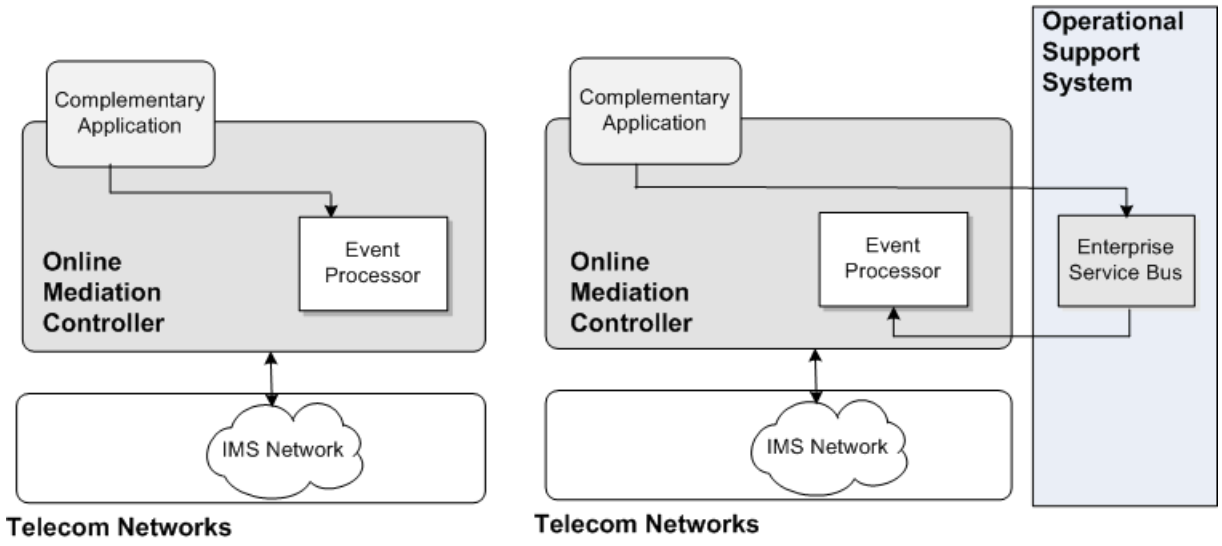
You can target events at only one destination, that is either an external OSS application, or the Event Processor.

If you target events at an external OSS application, then you must filter events published by built-in complementary applications and forward them to the Event Processor. Otherwise, you harm the functionality of the built-in complementary applications. Forwarding events to the Event Processor is necessary only if you deploy

one or more of the built-in complementary applications. The exact events that you need to filter and forward to the Event Processor depends on built-in application that you deploy. The documentation of each of the applications includes a section listing the events that the application fires.

Figure 7–1 shows how you can either target events at the internal Event Processor, or at external OSS application, in which case your OSS application should filter built-in application events and forward them to the Event Processor.

Figure 7–1 Targeting Events at Either Internal or External Entities



About the Event Notification API

The Event Notification API is SOAP-based. It gives the Online Mediation Controller the role of a SOAP client, and the application consuming the events acts as a SOAP server.

The methods and data types of the API are described using WSDL and XSD files. See "Using the Event Notification API" for more information.

About the Event Processor

The Event Notification Framework includes a built-in Event Processor that you use to manage events fired by built-in complementary applications. The Event Processor implements default handling for those events, that normally involves further activation of Online Mediation Controller components. For example, when receiving an event from the Threshold Notification application, the Event Processor triggers SMS sending to the subscriber.

The Event Processor can be extended by Oracle Consulting, to handle additional online events.

The Online Mediation Controller starts the Event Processor by default. The Event Processor itself does not require any specific configuration.

Using the Event Processor is optional. If you are not using the Event Processor, free the resources used by the Event Processor by stopping its bundles. See "Stopping the Event Processor" for more information.

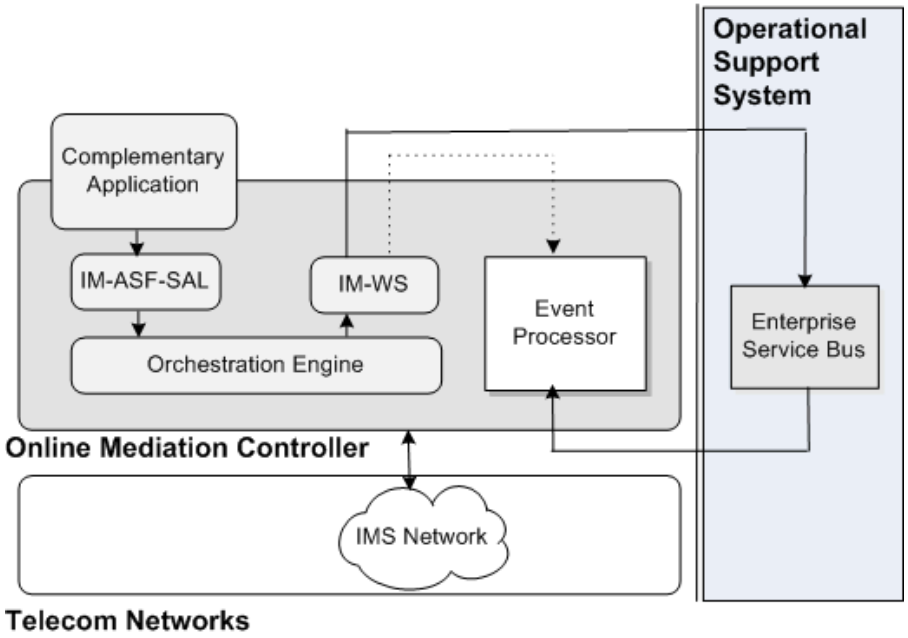
Setting Up the Event Notification Framework

Complementary applications use their SAL API to fire events. Events are thereby published through the application's IM-ASF-SAL module.

Figure 7–2 shows how events pass through the OE to IM-WS, which converts the event to a SOAP message, and send it out through the Web Service SSU. The Web Services SSU sends events at either the Event Processor or an external OSS application. In either case, you configure the internal Online Mediation Controller Event Processor to receive incoming events.

Figure 7–2 also shows the Online Mediation Controller components required to set up the Event Notification Framework.

Figure 7–2 Online Mediation Controller Components Required by the Event Notification Framework



Configuration Workflow

This section assumes that you have already deployed and configured the complementary application that publishes events and its related IM-ASF-SAL module.

To set up the Event Notification Framework:

1. In the Web Services SSU, configure Online Mediation Controller as a Web services client. See the discussion about SOAP client parameters in "Configuring SOAP Web Services Access" in *Oracle Communications Signaling Server Units Configuration Guide*.
2. In the Web Services SSU, configure the target consuming events. See "Configuring Target Event Consumers".
3. Deploy and configure an IM-WS module. See "Deploying and Configuring IM-WS".
4. Route events to IM-WS. See "Routing Events to IM-WS".

5. Enable HTTP network access by opening an HTTP listening port. See "Enabling HTTP Network Access".
6. Route incoming SOAP events to the Event Processor. See "Routing Incoming Events to the Event Processor".

Configuring Target Event Consumers

In the Web Services SSU, configure event consumers, as described in "Configuring Routing Rules for Outgoing Web Services Messages" in *Oracle Communications Service Broker Signaling Server Units Configuration Guide*.

You can define two or more consumers having a different URL that share the same alias, thereby adding a level of redundancy, treating all consumers as one logical destination, and balancing the load among consumers.

In the Administration Console:

1. In the navigation tree, expand **OCSB**.
2. Expand **Signaling Tier**.
3. Expand **SSU Web Services**.
4. Select **General**.
5. In the **SSU WS** tab, select the **Outgoing Routing Rules** tab.
6. Click **New**. The New dialog box appears.
7. In the **Name** field, enter a name for the target event consumer.
8. In the **Alias** field, enter an alias for the target event consumer.

   If you target events at the Event Processor, enter `sbeventprocess`. Otherwise, enter an alias that represents the external OSS application that you are targeting. For example: esb.
9. In the **Web Service URI**, enter the Web URI of the target event consumer:

   If you target events at the Event Processor, enter:

   `http://ip:port/soap/Events`

   Where:

   ip: the address of the signaling servers in the Signaling Domain where the Web Services SSU is running. This should be the address of the signaling server or server cluster in your deployment.

   port: the signaling servers' web services port.

   Otherwise, enter the URI of the external OSS application that you are targeting, that is a web services server, in the format:

   http://*address*:*port*/*web-service-name*.

   For example, http://telmobil.esb.org:80/eventnotification.
10. Fill in the remaining fields. See the discussion on "Configuring Routing Rules for Outgoing Web Services Messages", in *Oracle Communications Service Broker Signaling Server Units Configuration Guide* for more information.
11. Click **OK**.

Deploying and Configuring IM-WS

Deploy the IM-WS module as described in "Managing Interworking Modules" in *Oracle Communications Service Broker Modules Configuration Guide*.

In the Administration Console:

1. In the navigation tree, expand **OCSB**.
2. Expand **Processing Tier**, and then **Interworking Modules**.
3. Click **IM Management**.
4. In the **IM Management** tab, click **New**.
5. From the **Type** list, select **IMWS**.
6. In the **Name** field, enter a module instance name. For example, imws_instance.
7. Click **OK**.
8. Click **Commit**.

Configure the IM-WS module.

Specifically, in the **Web Service** tab, set the following fields as described:

- In the **Web Service Alias** field, enter the alias of the target event consumer that you configured in the Web Services SSU in step 2.
- From the **Web Service Type** list box, select **SOAP**.
- In the **Web Service Body Type**, enter **eventnotification**.

See the discussion on configuring the IM-WS in *Oracle Communications Service Broker Modules Configuration Guide* for more information.

Routing Events to IM-WS

Create an appropriate orchestration logic that routes events arriving from IM-ASF-SAL toward IM-WS. Use the Orchestration Studio to create the orchestration logic. See *Oracle Communications Service Broker Orchestration User's Guide*.

Specifically, you need to set a condition that identifies SAL messages with the **Method** set to **MESSAGE** and the **To** header containing the string **events_service**. For example:

To: sip:postEventRequest@events_service

Enabling HTTP Network Access

You enable HTTP network access to let incoming events reach the Event Processor.

To configure HTTP connectivity for the incoming events:

1. In the navigation tree, expand **OCSB**.
2. Expand **Signaling Tier**.
3. Expand **SSU Web Services**.
4. Select **General**.
5. Select the **HTTP** tab.
6. In the **Server** subtab, select the **Network Access** subtab.
7. Click **New**.
8. In the dialog box, set the properties for the HTTP listener as follows:

- **Server Address**: The local IP address or host name to which the port is bound. This should be the address of the signaling server or server cluster in your deployment.
- **Server Port**: An available listening port on Online Mediation Controller serving API client requests, such as 8989. This is the port number on which the signaling servers listens for incoming HTTP requests to the Event Processor.
- **Protocol**: The protocol used by incoming events. Choose HTTPS for secure HTTP or HTTP for unsecured HTTP. Oracle recommends using HTTPS for production deployments.
- **SSL Client Auth**: Whether SSL client certificate authentication is required for the connection. Enter false to disable SSL client certificate authentication, or true to require it. Enter true only if using HTTPS for the Protocol, in which case you will also need to set the key store and trust store identifiers.
- **Keystore Id**: The key you used when loading the keystore in the Credential Store. Use only with HTTPS. If you are using HTTP, this field can be left blank. If you have not already, load the keystore associated with the ID into the Credential Store. See *Oracle Communications Service Broker Administrator's Guide* for more information about the Credential Store.
- **Truststore Id**: The key you used when loading the trust store into the Credential Store. Use only with HTTPS. If using HTTP, this field can be left blank. See *Oracle Communications Service Broker Administrator's Guide* for more information about the Credential Store.
- **Target**: The signaling server to which this configuration applies. Leave blank to apply the configuration to all signaling servers in the deployment. Specify a signaling server name only if you want custom settings for individual signaling servers.

Routing Incoming Events to the Event Processor

To route incoming events to the Event Processor, create a new incoming routing rule in the Web Services SSU. See the discussion about incoming routing rules in "Configuring the Web Services SSU" in *Oracle Communications Service Broker Signaling Server Units Configuration Guide*.

Specifically, in the new rule, set the following fields as described:

- In the **Service Name** field, enter **oosEventService**.
- In the **Alias** field, enter **ssu:ocsb/outofsession.**

Stopping the Event Processor

To stop the Event Processor:

1. In the Administration Console, in the navigation tree, expand **OCSB**.
2. Expand **Domain Management**
3. Select **Packages**
4. In the **Filter** field, enter axia.oos. The Event Processor bundles will be listed:
   - com.convergin.wcs.axia.oos
   - com.convergin.wcs.axia.oos.ext
   - com.convergin.wcs.axia.oos.ooshash

5. Select a bundle that you want to stop and click **Stop**.

Using the Event Notification API

The event notification API is a SOAP-based API that you use to consume events, and run your specific business logic in the occurrence of these events.

The event notification API includes a single operation for receiving a notification. In general, any kind of event, triggered by any complementary application, is submitted through this API.

For most deployments, the primary consumer of the API notifications is the customer's ESB, for example, Oracle Enterprise Service Bus. The ESB filters the various notifications and distributes them to other OSS modules, based on the type of the event published. However, any other SOAP server that you implement can consume events.

To consume events, you develop a Web service with the WSDL files describing the event notification interface, and deploy the Web service on your SOAP server.

Obtaining WSDL and Schema

The following files describe the interface of the Event Notification API and its data types:

- events_service.wsdl
- events_interface.wsdl
- general.xsd
- events.xsd

You obtain the files from the signaling server. To navigate to the WSDL, go to the following address:

http://*host*:*port*/**soap**/**events**

Where *host* is the host name or IP address and *port* is the server port number you specified as the server address. See "Enabling HTTP Network Access".

If the WSDL is accessible, you can start developing your API server application. Many Integrated Development Environments (IDEs) can generate client code you can use as a starting point for your application by importing the WSDL into the IDE.

Event Notification API Reference

The rest of this chapter is a reference of the event notification API, describing the operation of the API. It lists the parameters accepted and returned by the operation. It provides examples of HTTP requests and responses for the operation.

Note: In the request and response message examples in this section, line breaks and spaces have been added to the data in the body of the message to improve readability.

Post Event

Notifies of the occurrence of an event.

The event can be of any kind. Its characteristic is defined by two fields: the event type and event category. The values of the event type and event category are not limited to a predefined set of values, and are extensible. The values depend on the complementary application initiating the event notification. Each application defines its own set of event types and categories. The events provided by each application are listed in the documentation of each application.

Request Body

Request body parameters are:

- **externalCorrelationID**: (xs:string) Optional. An event identifier, that you can use to correlate to the event.
- **senderIdentifier**: (xs:string) Optional. The identifier of the complementary application submitting the event.
- **subscriberID**: (SubscriberID) Optional. The identifier of the subscriber that owns the session for which the application sending the event was invoked. SubscriberID comprises:
  - code: (xs:int) The type of the subscriber identifier value. Possible values are:
    * 0: Mobile Subscriber ISDN Number (MSISDN) or Directory Number (DN)
    * 1: Mobile Identification Number (MIN) or International Mobile Subscriber Identity (IMSI)
    * 2: SIP URI
  - value: (xs:string) The subscriber identifier.
- **type**: (xs:string) The event type. Can be any number of characters, including any combination of letters, word characters and spaces.
- **category**: (xs:string) The event category. Helps dividing the different types of events into sets of related events. A group of related events is assigned with the same category.

  For example, events categorized as internal, are events consumed by internal Online Mediation Controller components and submitted by complementary applications to implement built in Online Mediation Controller features.

  Complementary application developers can define categories as they need.

  Can be any number of characters, including any combination of letters, word characters and spaces.
- **description**: (xs:string) Optional. A free text description of the event. Can be any number of characters, including any combination of letters, word characters and spaces.
- **additionalParam**: (NameValue) Optional. A recursive structure containing event specific parameters. Each parameter is a pair of:
  - code: (xs:string) A unique parameter identifier. Can be any number of characters, including any combination of letters, word characters and spaces.
  - A choice of:

    * value: (xs:string) The parameter value. Can be any number of characters, including any combination of letters, word characters and spaces.
    * nameValue (NameValue) more event specific parameters, related to each other, that you want to assemble in one set of parameters.

Response Body

Response body parameters are:

- **result**: (GenericOutput). The result of the operation. A data structure containing the following parameters:
  - fault: (Fault): Optional. A data structure containing the following:
    * faultcode: (xs:QName). A code identifying the fault.
    * faultstring: (xs:string). A human readable explanation of the fault.
    * faultactor: (xs:anyURI) Optional. Information about who caused the fault to happen.
    * detail (detail) Optional. Holds event specific error information. A data structure containing a sequence of xs:any and then xs:anyattribute

    Built-in complementary applications ignore this parameter.
  - externalId (string). The event identifier. This must always be equal to the externalCorrelationID parameter in the request body.
  - additionalInformation (NameValue) Optional. A recursive structure containing event specific parameters. See the description of the additionalParam parameter, in the request body. The list of parameters depends on the specific event.

Example

Example 7–1 shows a sample post event request.

Example 7–1 Request

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:even="http://xmlns.oracle.com/axia/events"
xmlns:gen="http://xmlns.oracle.com/ocsb/core/general">
<soapenv:Header/>
<soapenv:Body>
<even:postEventRequestParams>
<even:externalCorrelationID>71b42285</even:externalCorrelationID>
<even:senderID>FirstCallApplication</even:senderID>
<even:subscriberID>
<gen:code>0</gen:code>
<gen:value>+14081154970</gen:value>
</even:subscriberID>
<even:type>FirstCall</even:type>
<even:category>Charging</even:category>
even:description>account activation complete</even:description>
</even:postEventRequestParams>
</soapenv:Body>
</soapenv:Envelope>
```

Example 7–2 shows a sample post event response.

Example 7–2 Response

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
<S:Body>
<postEventResponseParams xmlns="http://xmlns.oracle.com/axia/events
xmlns:ns2="http://xmlns.oracle.com/ocsb/core/general">
<even:externalCorrelationID>71b42285</even:externalCorrelationID>
</postEventResponseParams>
</S:Body>
</S:Envelope>
```

8

Setting Up RADIUS Mediation for Accounting

This chapter describes the how to configure Oracle Communications Online Mediation Controller as a Remote Authentication Dial In Service (RADIUS) Manager. Online Mediation Controller integrates with an Oracle Communications Billing and Revenue Management (BRM) or Oracle Communications Elastic Charging Engine (ECE)RADIUS function server, to perform RADIUS accounting for offline charging sessions and events.

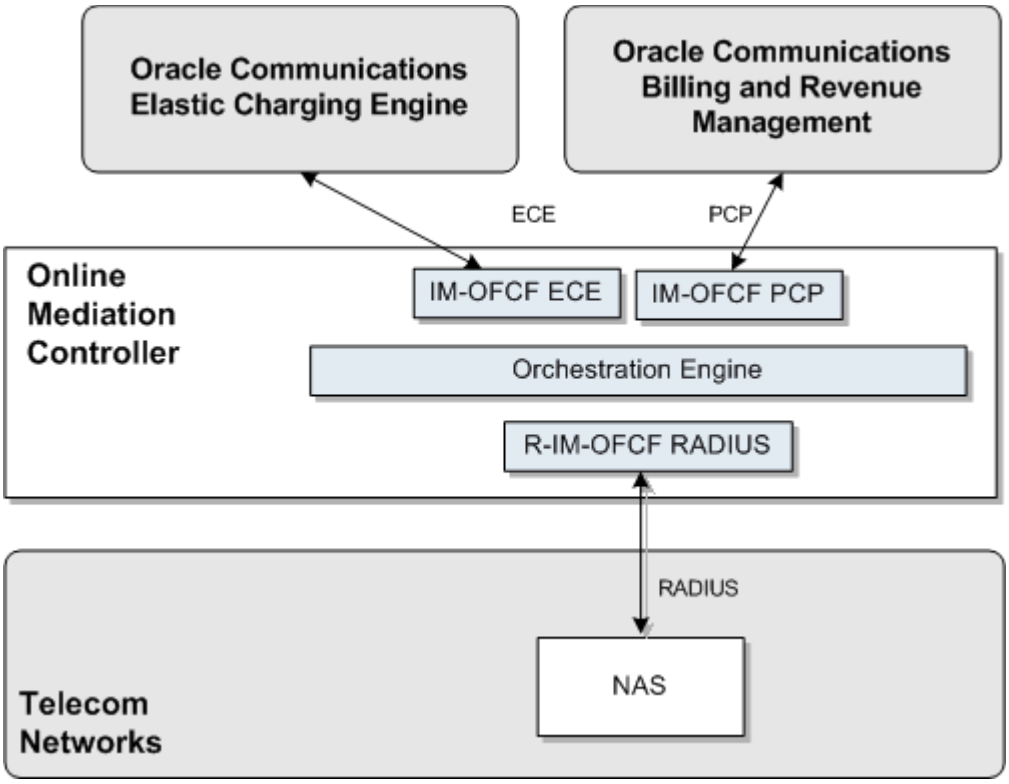
See "Setting Up RADIUS Mediation for Authentication and Authorization", for information on using Online Mediation Controller with BRM or ECE for RADIUS authentication and authorization.

About RADIUS Accounting Mediation

Online Mediation Controller translates RADIUS accounting requests to BRM or ECE client requests that BRM or ECE understands. Online Mediation Controller uses the Portal Connection Protocol (PCP) when communicating with BRM and the ECE ECE client API when used with ECE.

Figure 8–1 shows the Online Mediation Controller interworking modules that you need to configure to apply BRM or ECE offline charging services in a network supporting offline charging with RADIUS.

Figure 8–1 Online Mediation Controller Interworking Modules for Offline Charging



Configuring RADIUS Accounting Mediation

To set up Online Mediation Controller to perform RADIUS accounting mediation to BRM or ECE, you need to deploy and configure the following Online Mediation Controller components:

- IMOFCFPCP or IMOFCFECE
- RIMOFCFRADIUS

  See the chapters on configuring either the IMOFCFPCP or IMOFCFECE and configuring the RIMOFCFRADIUS in *Oracle Communications Service Broker Modules Configuration Guide* for more information.
- Orchestration Engine

  See *Oracle Communications Service Broker Orchestration User's Guide* for more information.
- SSU RADIUS
- SSU PCP or SSU ECE

  See the chapters on configuring the SSU RADIUS and configuring either the SSU PCP or SSU ECE in *Oracle Communications Service Broker Signaling Server Units Configuration Guide* for more information.

Configuration WorkFlow

To create an end-to-end configuration for RADIUS accounting:

1. Configure the SSU RADIUS. See "Configuring the SSU RADIUS", for more information.
2. Configure client profile and AVP filters. See "Configuring a Client Profile and AVP Filters", for more information.
3. Add proxy realms if required. See "Adding Proxy Realms", for more information.
4. Configure the SSU PCP or SSU ECE connection details to BRM or ECE. See "Connecting to BRM Through PCP", or "Connecting to ECE Using the ECE API", for more information.
5. Create and configure the SSU PCP or SSU ECE network entities. See "Creating and Configuring SSU PCP or SSU ECE Network Entities", for more information.
6. Create and configure an instance of RIMOFCFRADIUS. See "Creating and Configuring an RIMOFCFRADIUS Instance", for more information.
7. Create and configure the IMOFCFPCP or IMOFCFECE instance. See "Creating and Configuring an IMOFCFPCP or IMOFCFECE Instance", for more information.
8. Configure the Orchestration Engine to properly route the request to the BRM or ECE RADIUS accounting server. See "Creating Orchestration Logic for RADIUS Accounting", for more information.
9. Activate the interworking modules. See "Activating the RIMOFCFRADIUS and IMOFCFPCP or IMOFCFECE Instances" for more information.
10. Configure the RADIUS Mediation settings. See "Configuring RADIUS Mediation Settings" for more information.

Configuring the SSU RADIUS

Configure the SSU RADIUS for accounting requests as described in "Configuring the SSU RADIUS" in *Oracle Communications Service Broker Signaling Server Units Configuration Guide*. Use the following configuration data, specifically:

1. Create a new incoming routing rule.
2. Set the parameter **Name** to the rule name to use.
3. Set **Local Realm** to **any**. This is a case-sensitive field.
4. Set **Alias** to the instance name that you are going to use for the RIMOFCFRADIUS instance. This instance is created later in the configuration process. See "Creating and Configuring an RIMOFCFRADIUS Instance", for more information. We will refer to this name as *rimocfradius*. Set the type of IM instance to **RIMOCFRADIUS** and the domain id to **ocsb.com**.

   The complete string to enter in the **Alias** fields is:

   ssu:*rimofcfradius***.RIMOCFRADIUS@ocsb.com**
5. Click **Apply**.

Configuring a Client Profile and AVP Filters

To create a client profile:

1. In the **SSU RADIUS** Configuration screen, click the **RADIUS** tab.
2. Click the **Client Profile** tab.
3. Click the **ClientProfile** sub tab to define the RADIUS client profile properties.

4. Click **New**.
5. In the **New** window enter the following information:

   In the **Client Address** field, enter the address or address range for the RADIUS Network Authentication Server (NAS) client(s) to configure. You can define a single IP address or host name, or a group of RADIUS clients, if entered as a regular expression.

   In the **Client NAS Identifier** field, enter the ID of the client NAS. This can be a fully qualified domain name.

   In the **Authentication Shared Secret Key** field, enter the key in the Credential Store that maps to the secret in the Credential Store used to identify authentication requests from the NAS client.

   For more information about the Credential Store, see *Oracle Communications Service Broker Security Guide*.

   In the **accountingSharedSecretKey** field, enter the key in the Credential Store that maps to the secret in the Credential Store used to identify accounting requests from the NAS client.
6. Click **OK**.
7. Click the **Avps to copy from Request to Response** tab.
8. Choose the client profile to apply the filter to from the **Parent** drop-down list. The index of the client profile correlates to the keyId assigned to the client profile.
9. To add additional AVPs in incoming requests needed in the response:
   a. Click **New**.
   b. In the **New:** window enter:

      In the **Attribute Name** field, enter the name of an AVP included in the request and shall be included in the response.
   c. Click **Apply**.

Adding Proxy Realms

To a add a proxy realm to proxy requests to:

1. In the SSU RADIUS Configuration node, click the **RADIUS** tab.
2. Click the **Proxy Realm** tab.
3. Click **New**.
4. In the **New** window enter:

   In the **Name of the proxy realm** field, enter a name for the RADIUS server to proxy requests to.

   In the **Username Match Criteria** field, enter the username matching criteria. Use a regular expression matching the realm part of the username attribute in the request. For example, enter **isp1.net** for any user that belongs to isp1.net.

   In the **Authentication Shared Secret Key** field, enter the key in the Credential Store that maps to the secret in the Credential Store used to identify authentication requests from the NAS client. For more information about the Credential Store, see *Oracle Communications Service Broker Security Guide*.

In the **Accounting Shared Secret Key** field, enter the key in the Credential Store that maps to the secret in the Credential Store used to identify accounting requests from the NAS client.

In the **Request Timeout** field, enter the number of seconds to wait for a response before a request times out and is retried.

In the **Number of Retries** field, enter the number of times to retry a request before it is considered failed.

5. Click **Apply** to save your configuration.

Connecting to BRM Through PCP

To connect Online Mediation Controller to BRM:

1. Create BRM connection pools in the SSU PCP. See the discussion on connection pools in the chapter on configuring the PCP signaling server unit in *Oracle Communications Service Broker Signaling Server Units Configuration Guide* for more information.

   For additional information on BRM connection pools, consult the chapter on connection pools in *Oracle Communications Billing and Revenue Management System Administrator's Guide*.

2. Secure the BRM connection pools created in step 1, as described in the PCP signaling server configuration chapter in *Oracle Communications Service Broker Signaling Server Units Configuration Guide*.

3. In the Administration Console:

   a. Expand **OCSB**.

   b. Expand **Signaling Tier**.

   c. Select **SSU PCP**.

   d. Select the **PCP** tab.

   e. Select the **Credential Store** tab.

   f. In the **Password** area, enter the ID of the connection pool that you want to secure in the **Key** field. This should be the **Pool ID** you assigned to the connection pool created in step 1.

   g. In the **Password** area, enter the password of the BRM client application account used by the connection pool to access the BRM in the **Password** field. This should be the password of the account you configured in the **BRM CM Login ID** field when you initially defined the connection pool.

   h. In the **Password** area, uncheck the **One-way** check box.

   i. In the **Password** area, click the **Set** button.

   j. Repeat the Administration Console steps for each connection pool you want to secure.

4. Define destination BRM applications, as described in "Defining PCP Network Entities" in the chapter "Configuring the PCP Signaling Server Unit" in *Oracle Communications Service Broker Signaling Server Units Configuration Guide*.

5. Click **Commit** to save your configuration.

Connecting to ECE Using the ECE API

To connect Online Mediation Controller to ECE:

1. In the Administration Console:
   a. In the navigation tree, expand **OCSB**.
   b. Expand **Signaling Tier**.
   c. Select **SSU ECE**.
   d. Select the **ECE** tab.
   e. Click the **Coherence** tab.
2. Populate the ECE Protocol Adapter values used to connect to ECE using the information below. Consult your ECE administrator for specific information about the ECE implementation in your environment.

   In the **Coherence cluster name** field, enter the name of the Coherence cluster on which ECE runs. A default value of **BRM** is entered.

   In the **JMX management read-only** field, set whether Mbeans exposed by the ECE Coherence node allow operations that modify run-time attributes. The default value is set to **FALSE**.

   In the **Coherence log file name** field, provide a string used when logging is enabled. By default, the log is located in same directory as where a managed server starts.

   In the **Coherence log level** field, enter the log level for the ECE Coherence cluster. The possible numeric values range from -1 to 9. There is no default value. See the discussion on debugging in *Oracle Coherence Developer's Guide*, for more information on setting Coherence logging levels.

   In the **Use ECE well known address** field, select the boolean indicating whether a well known address (WKA) for ECE will be used. Multicast address is not supported when WKA is used.

   In the **Well know address 1 (ip:port)** field, provide the first WKA IP address and port number of the ECE Coherence cluster.

   In the **Well know address 2 (ip:port)** field, provide the second WKA IP address and port number of the ECE Coherence cluster.

   In the **Multicast address (ip:port)** field, provide the IP address and port number of the ECE Coherence cluster when using multicast.

   In the **Multicast TTL** field, enter in a value for the multicast time-to-live setting. This value determines the maximum number of hops a packet may traverse. Legal values are from 0 to 255.

   In the **Use SSL connection** field, select the boolean indicating whether to use a secure connection to ECE. The default value is **FALSE**. See the discussion on securing SSU ECE in *Oracle Communications Service Broker Signaling Server Units Configuration Guide*, for more information on setting up the SSL connection to ECE.

   See *Oracle Communications Elastic Charging Engine Administration Guide*, for additional information on configuring ECE.
3. Select the **General** tab to set the general parameters listed in Table 8–1:

Table 8–1 ECE OCS General Parameters

Name	Type	Description
Request Default Timeout	Integer	Specifies the default request timeout in milliseconds when no value is supplied by the outbound request. The default value is 2000 milliseconds.
ECE Request Batch Size	Integer	Specifies the number of ECE requests to send per request. The default value is 1.
ECE Request Batch Timeout	Integer	Specifies the ECE batch request timeout in milliseconds.
ECE Thread Pool Size	Integer	Specifies the number of ECE threads to use in the connection pool.

Creating and Configuring SSU PCP or SSU ECE Network Entities

Create network entities for SSU PCP or SSU ECE after completing the respective SSU connection configuration. See the respective chapters for SSU PCP or SSU ECE in *Oracle Communications Service Broker Signaling Server Units Configuration Guide*, for more information on creating network entities.

Creating and Configuring an RIMOFCFRADIUS Instance

Create and configure the RIMOFCFRADIUS instance for accounting requests as described in "Configuring RIMOFCF Radius" in *Oracle Communications Service Broker Modules Configuration Guide*. Use the following configuration data, specifically:

Give the IM a name that matches the **Alias** used when creating the incoming routing rule in the SSU RADIUS. See "Configuring the SSU RADIUS", for more information.

Creating and Configuring an IMOFCFPCP or IMOFCFECE Instance

Create and configure the IMOFCFPCP or IMOFCFECE instance for accounting requests as described in the respective configuration chapters for IMOFCFPCP or IMOFCFECE in *Oracle Communications Service Broker Modules Configuration Guide*. Give the IM a name that will be used by the Orchestration Engine when routing requests. We will refer to this name as *imofcfpcp* or *imofcfece*.

After creating the IMOFCFPCP or IMOFCFECE module, define the destination BRM or ECE system that the module communicates with. Specify the alias of a destination used when configuring the SSU PCP or SSU ECE network entity. See "Creating and Configuring SSU PCP or SSU ECE Network Entities", for more information.

In the Administration Console:

1. In the navigation tree, expand **OCSB**.
2. Expand **Processing Tier**.
3. Expand **Interworking Modules**.
4. Select either the IMOFCFPCP or IMOFCFECE module node.
5. Select the **Configuration** tab.
6. In the **Call Handling** tab, set the field **Destination Alias** to the string you provided for the Alias parameter when creating the SSU PCP or SSU ECE network entity.
7. Click **Apply**.

Configuring Service Type Parameters

By default, both the IMOFCFPCP and IMOFCFECE contain service type mapping values for use with basic BRM and ECE services. To view existing, or configure new service type mappings in the IM modules in the Administration Console:

1. In the navigation tree, expand **OCSB**.
2. Expand **Processing Tier**.
3. Expand **Interworking Modules**.
4. Select either the IMOFCFPCP or IMOFCFECE module.
5. Select either the **Rf PCP Mediation** or **Rf ECE Mediation** tab.
6. Select the **Service Types** tab.
7. Click **New** to create a service type mapping.

The **ServiceType** tab enables you to set up a mapping between RADIUS application IDs and BRM or ECE service types. Table 8–2 describes configuration parameters in the BRM **ServiceType** subtab. Table 8–3 describes configuration parameters in the ECE **ServiceType** subtab.

Table 8–2 Authentication Application Service Type Parameters for BRM

Name	Type	Description
BRM Service ID	Integer	The RADIUS application ID to be mapped to a BRM service type.
BRM Service Type	String	The BRM service type to use for the corresponding RADIUS application ID. For example: **service/ip**
Is Default Service Type	Boolean	Indicates whether to use this service type if none is specified. Set to: ■ **true** if to use this as a default value. ■ **false** to not use it as a default value.

Table 8–3 Authentication Application Service Type Parameters for ECE

Name	Type	Description
Service-Identifier AVP	Integer	The RADIUS application ID to be mapped to a BRM service type.
ECE product type	String	The ECE product type to use for the corresponding RADIUS application ID. For example: **VOICE**
ECE event type	String	The ECE event type to use for the corresponding RADIUS application ID. For example: **DATA_USAGE**
ECE specification version	Decimal	The ECE specification version.
Default service type	Boolean	Indicates whether to use this service type if none is specified. Set to: ■ **true** if to use this as a default value. ■ **false** to not use it as a default value.

Creating Orchestration Logic for RADIUS Accounting

Use the Orchestration Studio to route RADIUS accounting requests to the IMOFCFPCP or IMOFCFECE instance. See *Oracle Communications Service Broker Orchestration User's Guide*, for more information on configuring orchestration.

Use the following configuration data, specifically:

- Route the requests to **sip:***imofcfpcp***.IMOFCFPCP@ocsb.com** or **sip:***imofcfece***.IMOFCFECE@ocsb.com**

  Where *imofcfpcp* or *imofcfece* is the IM name you gave for the IM-OFCFPCP or IMOFCFECE instance.

Activating the RIMOFCFRADIUS and IMOFCFPCP or IMOFCFECE Instances

To activate the newly created RIMOFCFRADIUS and IMOFCFPCP or IMOFCFECE instances:

1. In the Domain Navigation pane, expand **OCSB**.
2. Expand **Processing Tier** and then **Interworking Modules**.
3. Select **IM Management**.
4. Click the RIMOFCFRADIUS instance. The instance name is the same as you gave when you created it.
5. Click **Activate**.
6. Click the IMOFCFPCP or IMOFCFECE instance. The instance name is the same as you gave when you created it.
7. Click **Activate**.

Configuring RADIUS Mediation Settings

This section describes how to configure RADUIS Mediation using the Online Mediation Controller Administration Console.

To access the RADIUS Mediation Configuration screen:

1. In the domain navigation pane, expand **OCSB**.
2. Expand **Processing Tier**.
3. Click **RADIUS Mediation**.
4. Configure the parameters in Table 8–4

Table 8–4 RADIUS Mediation General Parameters

Name	Type	Description
auth-timeout	Integer	The time to allow for an authentication requests to execute before it is considered to have timed out. Given in seconds.
BRM error codes	Integer	Comma separated BRM error codes which will trigger a request to be processed in Degraded Mode. See "Using Degraded Mode" for more information.

Table 8–4 (Cont.) RADIUS Mediation General Parameters

Name	Type	Description
degraded-mode-behavior	Enumeration, drop-down menu	Defines how authentication requests that times out are handled. Choose: ■ **accept** to treat the requests as accepted. ■ **discard** to discard the requests. ■ **reject** to reject the request.
ECE integration	Boolean	Indication whether to route charging requests to BRM or ECE. When **true**, requests will be routed to ECE.

Extending RADIUS Accounting Support

You can extend the accounting functionality by adding support for custom RADIUS AVPs. You do that by adding custom AVPs to the RADIUS dictionary in the SSU RADIUS. See the chapter on configuring the SSU RADIUS in *Oracle Communications Service Broker Signaling Server Units Configuration Guide,* for more information.

If you add custom AVPs to the RADIUS dictionary in the SSU RADIUS, you also need to implement custom mappers from RADIUS to Rf (deployed in RIMOFCFRADIUS), and from Rf to PCP or ECE (deployed in IMOFCFPCP or IMOFCFECE).

9

Setting Up RADIUS Mediation for Authentication and Authorization

This chapter describes the steps required to configure Oracle Communications Online Mediation Controller as a Remote Authentication Dial In Service (RADIUS) Manager. Online Mediation Controller supports RADIUS authentication and authorization integration with Oracle Communications Billing and Revenue Management (BRM). Network requests can also be routed to Oracle Communications Elastic Charging Engine (ECE) for authorization.

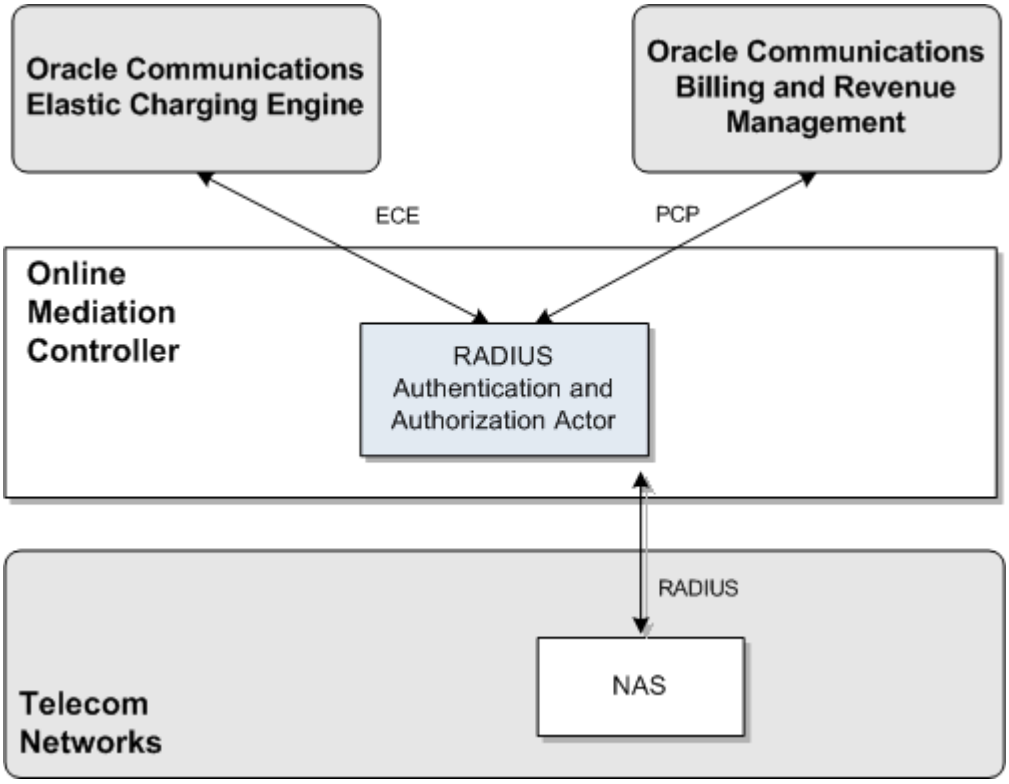
See "Setting Up RADIUS Mediation for Accounting", for information on setting up Online Mediation Controller for use with BRM or ECE RADIUS accounting.

About RADIUS Authentication and Authorization Mediation

Online Mediation Controller translates RADIUS authentication and authorization requests to requests that BRM and ECE understands. The Portal Connection Protocol (PCP) is used for communication with BRM while the ECE Authentication API is used for ECE.

Figure 9–1 shows the Online Mediation Controller components that you need to set up and configure to apply the BRM or ECE authentication and authorization services in a network supporting authentication and authorization with RADIUS.

Figure 9–1 Online Mediation Controller Components for Authentication and Authorization



Configuring RADIUS Authentication and Authorization

To set up Online Mediation Controller to perform RADIUS Authentication and Authorization mediation to BRM or ECE, you need to configure the following Online Mediation Controller components:

- SSU RADIUS
- SSU PCP or SSU ECE
- RADIUS Authentication module

Performing RADIUS Authentication and Authorization

See BRM documentation for information on how authentication and authorization is done in BRM.

See the documentation for **oracle.communication.brm.charging messages.query** in *Oracle Communications ECE Java API Reference* for information on the ECE Charging API.

Configuration Workflow

To create an end-to-end configuration for RADIUS authentication and authorization:

1. Configure the SSU RADIUS. See "Configuring the SSU RADIUS" for more information.

2. Create a set of client profiles and AVP filters for requests and responses. See "Configuring a Client Profile and AVP Filters" for more information.
3. Create a set of Proxy Realms. See "Adding Proxy Realms" for more information.
4. Configure the SSU PCP or SSU ECE to connect to BRM or ECE, respectively. See "Connecting to BRM Through PCP" or "Connecting to ECE Using the ECE API" for more information.
5. Configure RADIUS Mediation. See "Configuring RADIUS Mediation", for more information.

Configuring the SSU RADIUS

Configure the SSU RADIUS for accounting requests as described in "Configuring the SSU RADIUS" in *Oracle Communications Service Broker Signaling Server Units Configuration Guide*. Use the following configuration data, specifically:

1. Create a new incoming routing rule.
2. Set the parameter **Name** to the rule name to use.
3. Set **Local Realm** to **any**. This is a case-sensitive field.
4. Set **Alias** to the instance name that you are going to use for the RIMOFCFRADIUS instance.

   The complete string to enter in the **Alias** fields is:

   ssu:*rimofcfradius***.RIMOCFRADIUS@ocsb.com**
5. Click **Apply**.

Configuring a Client Profile and AVP Filters

To create a client profile:

1. In the **SSU RADIUS** Configuration screen, click the **RADIUS** tab.
2. Click the **Client Profile** tab.
3. Click the **ClientProfile** sub tab to define the RADIUS client profile properties.
4. Click **New**.
5. In the **New** window enter the following information:

   In the **Client Address** field, enter the address or address range for the RADIUS Network Authentication Server (NAS) client(s) to configure. You can define a single IP address or host name, or a group of RADIUS clients, if entered as a regular expression.

   In the **Client NAS Identifier** field, enter the ID of the client NAS. This can be a fully qualified domain name.

   In the **Authentication Shared Secret Key** field, enter the key in the Credential Store that maps to the secret in the Credential Store used to identify authentication requests from the NAS client.

   For more information about the Credential Store, see *Service Broker Security Guide*.

   In the **accountingSharedSecretKey** field, enter the key in the Credential Store that maps to the secret in the Credential Store used to identify accounting requests from the NAS client.
6. Click **OK**.

7. Click the **Avps to copy from Request to Response** tab.
8. Choose the client profile to apply the filter to from the **Parent** drop-down list. The index of the client profile correlates to the keyId assigned to the client profile.
9. To add additional AVPs in incoming requests needed in the response:
   a. Click **New**.
   b. In the **New:** window enter:

      In the **Attribute Name** field, enter the name of an AVP included in the request and shall be included in the response.
   c. Click **Apply**.

Adding Proxy Realms

To a add a proxy realm to proxy requests to:

1. In the SSU RADIUS Configuration node, click the **RADIUS** tab.
2. Click the **Proxy Realm** tab.
3. Click **New**.
4. In the **New** window enter:

   In the **Name of the proxy realm** field, enter a name for the RADIUS server to proxy requests to.

   In the **Username Match Criteria** field, enter the username matching criteria. Use a regular expression matching the realm part of the username attribute in the request. For example, enter **isp1.net** for any user that belongs to isp1.net.

   In the **Authentication Shared Secret Key** field, enter the key in the Credential Store that maps to the secret in the Credential Store used to identify authentication requests from the NAS client. For more information about the Credential Store, see *Oracle Communications Service Broker Security Guide*.

   In the **Accounting Shared Secret Key** field, enter the key in the Credential Store that maps to the secret in the Credential Store used to identify accounting requests from the NAS client.

   In the **Request Timeout** field, enter the number of seconds to wait for a response before a request times out and is retried.

   In the **Number of Retries** field, enter the number of times to retry a request before it is considered failed.
5. Click **Apply** to save your configuration.

Connecting to BRM Through PCP

To connect Online Mediation Controller to BRM:

1. Create BRM connection pools in the SSU PCP. See the discussion on connection pools in the chapter on configuring the PCP signaling server unit in *Oracle Communications Service Broker Signaling Server Units Configuration Guide* for more information.

   For additional information on BRM connection pools, consult the chapter on connection pools in *Oracle Communications Billing and Revenue Management System Administrator's Guide*.

2. Secure the BRM connection pools created in step 1, as described in the PCP signaling server configuration chapter in *Oracle Communications Service Broker Signaling Server Units Configuration Guide*.
3. In the Administration Console:
   a. Expand **OCSB**.
   b. Expand **Signaling Tier**.
   c. Select **SSU PCP**.
   d. Select the **PCP** tab.
   e. Select the **Credential Store** tab.
   f. In the **Password** area, enter the ID of the connection pool that you want to secure in the **Key** field. This should be the **Pool ID** you assigned to the connection pool created in step 1.
   g. In the **Password** area, enter the password of the BRM client application account used by the connection pool to access the BRM in the **Password** field. This should be the password of the account you configured in the **BRM CM Login ID** field when you initially defined the connection pool.
   h. In the **Password** area, uncheck the **One-way** check box.
   i. In the **Password** area, click **Set**.
   j. Repeat the Administration Console steps for each connection pool you want to secure.
4. Define destination BRM applications, as described in "Defining PCP Network Entities" in the chapter "Configuring the PCP Signaling Server Unit" in *Oracle Communications Service Broker Signaling Server Units Configuration Guide*.
5. Click **Commit** to save your configuration.

Connecting to ECE Using the ECE API

To connect Online Mediation Controller to ECE:

1. In the Administration Console:
   a. In the navigation tree, expand **OCSB**.
   b. Expand **Signaling Tier**.
   c. Select **SSU ECE**.
   d. Select the **ECE** tab.
   e. Click the **Coherence** tab.
2. Populate the ECE Protocol Adapter values used to connect to ECE using the information below. Consult your ECE administrator for specific information about the ECE implementation in your environment.

   In the **Coherence cluster name** field, enter the name of the Coherence cluster on which ECE runs. A default value of **BRM** is entered.

   In the **JMX management read-only** field, set whether Mbeans exposed by the ECE Coherence node allow operations that modify run-time attributes. The default value is set to **FALSE**.

In the **Coherence log file name** field, provide a string used when logging is enabled. By default, the log is located in the same directory as where the managed server is started.

In the **Coherence log level** field, enter the log level for the ECE Coherence cluster. The possible numeric values range from -1 to 9. There is no default value. See *Oracle Coherence User Guide*, for more information on setting Coherence logging levels.

In the **Use ECE well known address** field, select the boolean indicating whether a well known address (WKA) for ECE will be used. Multicast address is not supported when WKA is used.

In the **Well know address 1 (ip:port)** field, provide the first WKA IP address and port number of the ECE Coherence cluster.

In the **Well know address 2 (ip:port)** field, provide the second WKA IP address and port number of the ECE Coherence cluster.

In the **Multicast address (ip:port)** field, provide the IP address and port number of the ECE Coherence cluster when using multicast.

In the **Multicast TTL** field, enter in a value for the multicast time-to-live setting. This value determines the maximum number of hops a packet may traverse. Legal values are from 0 to 255.

In the **Use SSL connection** field, select the boolean indicating whether to use a secure connection to ECE. The default value is **FALSE**. See the discussion on securing SSU ECE in *Oracle Communications Service Broker Signaling Server Units Configuration Guide*, for more information on setting up the SSL connection to ECE.

See *Oracle Communications Elastic Charging Engine Administration Guide*, for additional information on configuring ECE.

3. Select the **General** tab to set the general parameters listed in Table 9–1:

Table 9–1 ECE OCS General Parameters

Name	Type	Description
Request Default Timeout	Integer	Specifies the default request timeout in milliseconds when no value is supplied by the outbound request. The default value is 2000 milliseconds.
ECE Request Batch Size	Integer	Specifies the number of ECE requests to send per request. The default value is 1.
ECE Request Batch Timeout	Integer	Specifies the ECE batch request timeout in milliseconds.
ECE Thread Pool Size	Integer	Specifies the number of ECE threads to use in the connection pool.

Configuring RADIUS Mediation

This section describes how to configure RADUIS Mediation using the Online Mediation Controller Administration Console.

To access the RADIUS Mediation Configuration screen:

1. In the domain navigation pane, expand **OCSB**.
2. Expand **Processing Tier**.

3. Click **RADIUS Mediation**.

The Radius Mediation configuration pane contains the **General** described in Table 9–2.

Table 9–2 RadiusAthentication Configuration Subtabs

Subtab	Description
General	Enables you to define time out value for authentication requests and how to treat accounting requests when Online Mediation Controller operates in degraded mode. See "Configuring General Parameters"

Configuring General Parameters

The **General** tab enables you to set up how the Authentication application treats authentication requests that time out. Table 9–3 describes configuration parameters in the **General** subtab.

Table 9–3 Authentication Application General Parameters

Name	Type	Description
auth-timeout	Integer	The time to allow for an authentication requests to execute before it is considered to have timed out. Given in seconds.
degraded-mode-behavior	Enumeration, drop-down menu	Defines how authentication requests that times out are handled. Choose: ■ **accept** to treat the requests as accepted. ■ **discard** to discard the requests. ■ **reject** to reject the request.

Configuring Service Type Parameters

By default, the authentication and authorization service type parameter configuration is the same as that for RADIUS accounting. See "Creating and Configuring an IMOFCFPCP or IMOFCFECE Instance", for information on viewing and customizing service type mapping.

Extending Authentication and Authorization Support

You can extend the authentication and authorization functionality by adding support for custom RADIUS AVPs. You do that by adding custom AVPs to the RADIUS dictionary in the RADIUS SSU. See “Configuring the RADIUS SSU” in *Oracle Communications Service Broker Signaling Server Units Configuration Guide* for more information.

10

Setting Up Diameter Ro Mediation

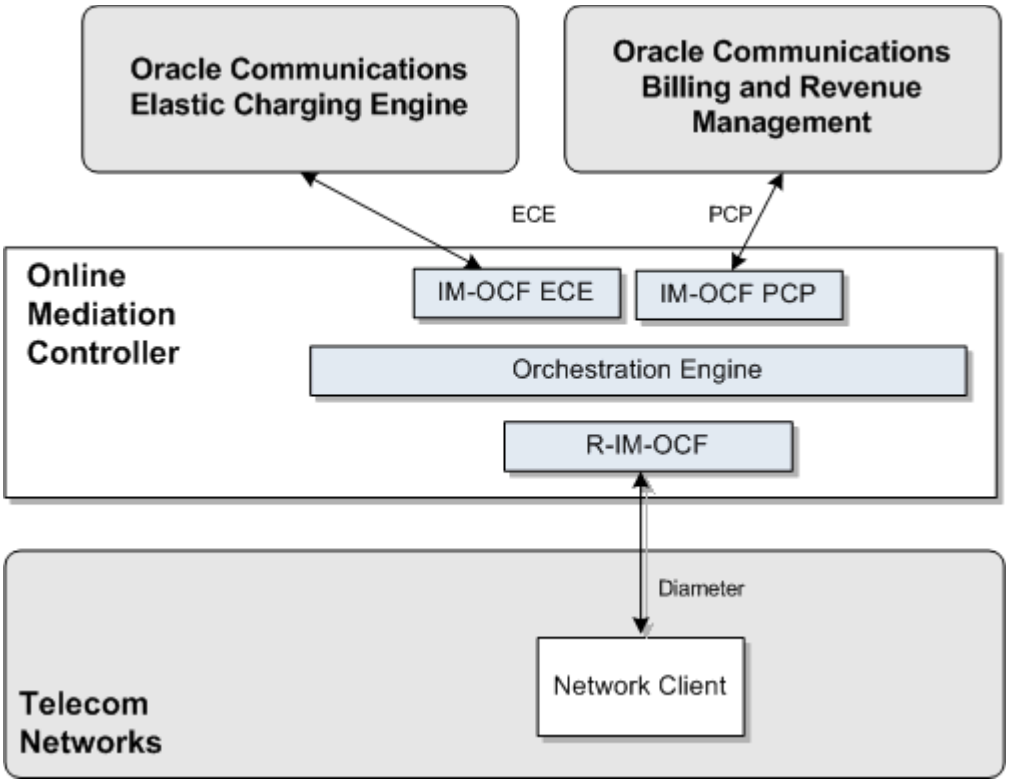
This chapter describes the how to configure Oracle Communications Online Mediation Controller to mediate Diameter Ro charging requests from network clients to an Oracle Communications Billing and Revenue Management (BRM) or Oracle Communications Elastic Charging Engine (ECE) Online Charging System (OCS).

About Diameter Ro Accounting Mediation

Online Mediation Controller translates Diameter Ro Credit Control Requests (CCRs) to BRM or ECE client requests for processing. Online Mediation Controller also translates responses from BRM and ECE into Diameter Ro Credit Control Answers (CCA) before replying to network clients. Online Mediation Controller uses the Portal Connection Protocol (PCP) when communicating with BRM and the ECE client API when used with ECE. See "Diameter Ro to BRM Opcode Mapping" for Diameter Ro operations, AVP and result codes to BRM opcode, Flist field, and results code mapping information.

Figure 10–1 shows the Online Mediation Controller interworking modules (IM) that you need to configure to apply BRM or ECE online charging services in a network supporting Diameter Ro. The Signaling Server Units (SSUs) are not shown.

Figure 10–1 Online Mediation Controller Interworking Modules for Online Charging



Configuring Diameter Ro Accounting Mediation

To set up Online Mediation Controller to perform Diameter Ro online accounting mediation to BRM or ECE, you need to configure the following Online Mediation Controller components:

1. Configure the SSU Diameter connection from Online Mediation Controller to the telecom network. See "Configuring the SSU DIAMETER", for more information.
2. Configure the SSU PCP or SSU ECE connection from Online Mediation Controller to BRM or ECE. See "Connecting to BRM Through PCP", or "Connecting to ECE Using the ECE API", for more information.
3. Create and configure the SSU PCP or SSU ECE network entities. See "Creating and Configuring SSU PCP or SSU ECE Network Entities", for more information.
4. Create and configure an instance of RIMOCF for translating Diameter Ro messages to Online Mediation Controller's internal message format. See "Creating and Configuring an RIMOCF Instance", for more information.
5. Create and configure the IMOCFPCP, or IMOCFECE, or both instance(s) for translating Online Mediation Controller messages to PCP or ECE API messages. Online Mediation Controller supports simultaneous integration with BRM and ECE. See "Creating and Configuring an IMOCFPCP and IMOCFECE Instance", for more information.

   If your implementation uses customized IM processor classes, configure the needed processor mapping. See "Mapping Custom Processor Classes", for more information.

6. Configure the Orchestration Engine to properly route the request to the BRM or ECE Diameter accounting server. See "Creating Orchestration Logic for Diameter Accounting", for more information.
7. Activate the interworking modules. See "Activating the RIMOCF and IMOCFPCP or IMOCFECE Instances" for more information.

Configuring the SSU DIAMETER

To configure SSU Diameter to accept accounting requests, complete the following steps. See the chapter on the Diameter signaling server unit in *Oracle Communications Service Broker Signaling Server Units Configuration Guide*, for more information.

1. Create an incoming routing rule.
2. Set the parameter **Name** to the rule name to use.
3. Set **Local Realm** to **any**. This is a case-sensitive field.
4. Set **Alias** to the instance name that you use for the RIMOCF instance. This instance is created later in the configuration process. See "Creating and Configuring an RIMOCF Instance", for more information. We will refer to this name as *rimocf*. Set the type of IM instance to **RIMOCF** and the domain id to **ocsb.com**.

   The complete string to enter in the **Alias** fields is:

   ssu:*rimocf***.RIMOCF@ocsb.com**
5. Click **Apply**.

Connecting to BRM Through PCP

To connect Online Mediation Controller to BRM using SSU PCP:

1. Create BRM connection pools in the SSU PCP. See the discussion on connection pools in the chapter on configuring the PCP signaling server unit in *Oracle Communications Service Broker Signaling Server Units Configuration Guide* for more information.

   For additional information on BRM connection pools, consult the chapter on connection pools in *Oracle Communications Billing and Revenue Management System Administrator's Guide*.
2. Secure the BRM connection pools created in step 1, as described in the PCP signaling server configuration chapter in *Oracle Communications Service Broker Signaling Server Units Configuration Guide*.
3. In the Administration Console:
   a. Expand **OCSB**.
   b. Expand **Signaling Tier**.
   c. Select **SSU PCP**.
   d. Select the **PCP** tab.
   e. Select the **Credential Store** tab.
   f. In the **Password** area, enter the ID of the connection pool that you want to secure in the **Key** field. This should be the **Pool ID** you assigned to the connection pool created in step 1.
   g. In the **Password** area, enter the password of the BRM client application account used by the connection pool to access the BRM in the **Password** field.

This should be the password of the account you configured in the **BRM CM Login ID** field when you initially defined the connection pool. See the discussion on SSU PCP in the *Oracle Communications Service Broker Signaling Server Units Configuration Guide*, for more information on connection pools.

h. In the **Password** area, uncheck the **One-way** check box.

i. In the **Password** area, click **Set**.

j. Repeat the Administration Console steps for each connection pool you want to secure.

4. Define destination BRM applications as PCP network entities in the **PCP Network Entities** tab. See the discussion on defining PCP network entities in the chapter "Configuring the PCP Signaling Server Unit" in *Oracle Communications Service Broker Signaling Server Units Configuration Guide*,for more information.

5. Click **Commit** to save your configuration.

Connecting to ECE Using the ECE API

To connect Online Mediation Controller to ECE:

1. In the Administration Console:

   a. In the navigation tree, expand **OCSB**.

   b. Expand **Signaling Tier**.

   c. Select **SSU ECE**.

   d. Select the **ECE** tab.

   e. Click the **Coherence** tab.

2. Populate the ECE Protocol Adapter values used to connect to ECE using the information below. Consult your ECE administrator for specific information about the ECE implementation in your environment.

   In the **Coherence cluster name** field, enter the name of the Coherence cluster on which ECE runs. A default value of **BRM** is entered.

   In the **JMX management read-only** field, set whether Mbeans exposed by the ECE Coherence node allow operations that modify run-time attributes. The default value is set to **FALSE**.

   In the **Coherence log file name** field, provide a string used when logging is enabled for debugging purposes. By default, the log is located in the same directory as where the managed server is started.

   In the **Coherence log level** field, enter the log level for the ECE Coherence cluster. The possible numeric values range from -1 to 9. There is no default value. Enable logging for debugging by setting a value. See *Oracle Coherence User Guide*, for more information on setting Coherence logging levels.

   In the **Use ECE well known address** field, select the boolean indicating whether a well known address (WKA) for ECE will be used. Multicast address is not supported when WKA is used.

   In the **Well know address 1 (ip:port)** field, provide the first WKA IP address and port number of the ECE Coherence cluster.

   In the **Well know address 2 (ip:port)** field, provide the second WKA IP address and port number of the ECE Coherence cluster.

In the **Multicast address (ip:port)** field, provide the IP address and port number of the ECE Coherence cluster when using multicast.

In the **Multicast TTL** field, enter in a value for the multicast time-to-live setting. This value determines the maximum number of hops a packet may traverse. Legal values are from 0 to 255.

In the **Use SSL connection** field, select the boolean indicating whether to use a secure connection to ECE. The default value is **FALSE**. See the discussion on securing SSU ECE in *Oracle Communications Service Broker Signaling Server Units Configuration Guide*, for more information on setting up the SSL connection to ECE.

See *Oracle Communications Elastic Charging Engine Administration Guide*, for additional information on configuring ECE.

3. Select the **General** tab to set the general parameters listed below:

   In the **Request Default Timeout** field, specify the default request timeout in milliseconds when no value is supplied by the outbound request. The default value is 2000 milliseconds.

   In the **ECE Request Batch Size** field, specify the number of ECE requests to send per request. The default value is **1**.

   In the **ECE Request Batch Timeout** field, specify the ECE batch request timeout in milliseconds.

   In the **ECE Thread Pool Size** field, specify the number of ECE threads to use in the connection pool.

Creating and Configuring SSU PCP or SSU ECE Network Entities

Create network entities for SSU PCP or SSU ECE after completing the respective SSU connection configuration. See the respective chapters for SSU PCP or SSU ECE in *Oracle Communications Service Broker Signaling Server Units Configuration Guide*, for more information on creating network entities.

Creating and Configuring an RIMOCF Instance

Create and configure the RIMOCF instance for accounting requests as described in "Configuring RIMOCF" in *Oracle Communications Service Broker Modules Configuration Guide*. Use the following configuration data, specifically:

Give the IM a name that matches the **Alias** used when creating the incoming routing rule in the SSU Diameter. See "Configuring the SSU DIAMETER", for more information.

Creating and Configuring an IMOCFPCP and IMOCFECE Instance

Create and configure the IMOCFPCP or IMOCFECE instance for accounting requests as described in the respective configuration chapters for IMOCFPCP or IMOCFECE in *Oracle Communications Service Broker Modules Configuration Guide*. Give the IM a name that will be used by the Orchestration Engine when routing requests. We will refer to this name as *imocfpcp* or *imocfece*.

After creating the IMOCFPCP, or IMOCFECE, or both module(s), define the destination BRM or ECE system that the module communicates with. Specify the alias of a destination used when configuring the SSU PCP or SSU ECE network entity. See "Creating and Configuring SSU PCP or SSU ECE Network Entities", for more information.

In the Administration Console:

1. In the navigation tree, expand **OCSB**.
2. Expand **Processing Tier**.
3. Expand **Interworking Modules**.
4. Select either the IMOFCFPCP or IMOFCFECE module node.
5. Select the **Configuration** tab.
6. In the **Call Handling** tab, set the field **Destination Alias** to the string you provided for the Alias parameter when creating the SSU PCP or SSU ECE network entity.
7. Click **Apply**.

Configuring Service Type Parameters

By default, both the IMOCFPCP and IMOCFECE contain service type mapping values for use with basic BRM and ECE services. To view existing, or configure new service type mappings in the IM modules in the Administration Console:

1. In the navigation tree, expand **OCSB**.
2. Expand **Processing Tier**.
3. Expand **Interworking Modules**.
4. Select either the IMOCFPCP or IMOCFECE module node.
5. Select either the **Ro PCP Mediation** or **Ro ECE Mediation** tab.
6. Select the **Service Types** tab.
7. Click **New** to create a service type mapping.

The **ServiceType** tab enables you to set up a mapping between Diameter application IDs and BRM or ECE service types. Table 10–1 describes configuration parameters in the BRM **ServiceType** subtab. Table 10–2 describes configuration parameters in the ECE **ServiceType** subtab.

Table 10–1 Authentication Application Service Type Parameters for BRM

Name	Type	Description
Service-Identifier AVP	Integer	The Diameter Service-Identifier AVP value to be mapped to a BRM service type.
BRM Service Type	String	The BRM service type to use for the corresponding Diameter Service-Identifier AVP value. For example: **service/ip**
Is Default Service Type	Boolean	Indicates whether to use this service type if none is specified. Set to: ■ **true** if to use this as a default value. ■ **false** to not use it as a default value.

Table 10–2 Authentication Application Service Type Parameters for ECE

Name	Type	Description
Service-Identifier AVP	Integer	The Diameter Service-Identifier AVP value to be mapped to a BRM service type.

Table 10–2 (Cont.) Authentication Application Service Type Parameters for ECE

Name	Type	Description
Service-Context-Id	Integer	The Service Context Id AVP value
Rating-Group-AVP	Integer	The Rating Group AVP value.
ECE product type	String	The ECE product type to use for the corresponding Diameter Service-Identifier AVP value. For example: **VOICE**
ECE event type	String	The ECE event type to use for the corresponding Diameter Service-Identifier AVP value. For example: **DATA_USAGE**
ECE specification version	Decimal	The ECE specification version.
Default service type	Boolean	Indicates whether to use this service type if none is specified. Set to: ■ **true** if to use this as a default value. ■ **false** to not use it as a default value.

Mapping Custom Processor Classes

If your implementation uses customized BRM, or ECE, or both processor classes, edit the entries in the **Processor Mapping** subtabs located in the respective mediation tabs of the IMOFCFPCP and or IMOFCFECE Administration console configuration nodes.

For information on how to edit processor class mappings in IMOFCFPCP, or IMOFCFECE, see the chapters on configuring these IMs in *Oracle Communications Service Broker Modules Configuration Guide*.

Creating Orchestration Logic for Diameter Accounting

Use the Orchestration Studio to route Diameter accounting requests to the IMOCFPCP or IMOCFECE instance. See *Oracle Communications Service Broker Orchestration User's Guide*, for more information on configuring orchestration.

Use the following configuration data, specifically:

- Route the requests to **sip:***imocfpcp***.IMOCFPCP@ocsb.com** or **sip:***imocfece***.IMOCFECE@ocsb.com**

  Where *imocfpcp* or *imocfece* is the IM name you gave for the IMOCFPCP or IMOCFECE instance.

Activating the RIMOCF and IMOCFPCP or IMOCFECE Instances

To activate the newly created RIMOCF and IMOCFPCP or IMOCFECE instances:

1. In the Domain Navigation pane, expand **OCSB**.
2. Expand **Processing Tier** and then **Interworking Modules**.
3. Select **IM Management**.
4. Click the RIMOCF instance. The instance name is the same as you gave when you created it.
5. Click **Activate**.

6. Click the IMOCFPCP or IMOCFECE instance. The instance name is the same as you gave when you created it.
7. Click **Activate**.

Extending RADIUS Accounting Support

You can extend the accounting functionality by adding support for custom RADIUS AVPs. You do that by adding custom AVPs to the RADIUS dictionary in the SSU RADIUS. See the chapter on configuring the SSU RADIUS in *Oracle Communications Service Broker Signaling Server Units Configuration Guide,* for more information.

If you add custom AVPs to the RADIUS dictionary in the SSU RADIUS, you also need to implement custom mappers from RADIUS to Rf (deployed in RIMOFCFRADIUS), and from Rf to PCP or ECE (deployed in IMOFCFPCP or IMOFCFECE). See the discussion on configuring service type parameters in the respective IM chapters of *Oracle Communications Service Broker Modules Configuration Guide*, for more information.

11

Monitoring Online Mediation Controller

This chapter describes how to monitor Oracle Communications Online Mediation Controller.

About Monitoring Online Mediation Controller

You can monitor how Online Mediation Controller operates by receiving the following information:

- Statistics on messages and sessions that Interworking Modules (IMs) and the Orchestration Engine (OE) handle. See "Monitoring the Processing Domain" for more information.
- Status of the network entities with which Signaling Server Units (SSUs) communicate. See "Monitoring the Signaling Domain" for more information.
- Availability of the Diameter and RADIUS ports. See "Checking Availability of the Diameter and RADIUS Ports" for more information.

Monitoring the Processing Domain

A deployment of Online Mediation Controller might involve the following components:

- OE
- In online charging solutions:
  - IM-OCF, which communicates with an online charging system
  - R-IM-OCF, which communicates with the network
- In offline charging solutions:
  - IM-OFCF PCP, which communicates with an offline charging system
  - R-IM-OFCF, which communicates with the network

Using runtime MBeans, you can gather statistics on sessions and messages that these IMs send and receive through their interfaces:

- Base Diameter. See "Monitoring the Base Diameter Interface" for more information.
- Diameter Ro (online charging). See "Monitoring the Interfaces in Online Charging Solutions" for more information.
- Diameter Rf (offline charging). See "Monitoring the Interfaces in Offline Charging Solutions" for more information.

Monitoring the Base Diameter Interface

Both online charging and offline charging IMs provide the base Diameter interface. Using **DiameterRuntimeMBean**, you can get statistics on messages that the IM sent and received through this interface.

Online Mediation Controller creates a separate instance of **DiameterRuntimeMBean** for each instance of IM-OCF, R-IM-OCF, IM-OFCF, and R-IM-OFCF.

The object name of this MBean is **com.convergin:Type=DiameterRuntime,Version=***MBean_Version*,**Location=**<*server-name*>,**Name=***IM_Instance_Name*.**Diameter**.

Table 11–1 describes the counters that **DiameterRuntimeMBean** provides.

Table 11–1 DiameterRuntimeMBean Counters

To Get Total Number of...	Use...
Sent and received answers	getAnsCount()
Sent and received requests	getRequestCount()

For more information on how to access runtime MBeans, see the discussion on monitoring Service Broker using Runtime MBeans in *Oracle Communications Service Broker System Administrator's Guide*.

Monitoring the Interfaces in Online Charging Solutions

In an online charging solution, IM-OCF and R-IM-OCF provide the Diameter Ro interface. You can get the following statistics on messages that the IM sent and received through this interface:

- Counters of Diameter Ro charging requests and charging answers. See "Getting Statistics on Charging Requests and Charging Answers" for more information.
- Gauge for Diameter Ro sessions. See "Getting Statistics on Currently Active Sessions" for more information.

Getting Statistics on Charging Requests and Charging Answers

Using **RoRuntimeMBean**, you can get counters of charging requests and charging answers that IM-OCF or R-IM-OCF sent and received.

Online Mediation Controller creates a separate instance of **RoRuntimeMBean** for each instance of IM-OCF and R-IM-OCF.

The object name of this MBean is **com.convergin:Type=RoRuntime,Version=***MBean_Version*,**Location=**<*server-name*>,**Name=***IM_Instance_Name*.**Ro**.

Table 11–2 describes the counters that **RoRuntimeMBean** provides.

Table 11–2 RoRuntimeMBean Counters

To Get Total Number of...	Use...
Session Charging with Unit Reservation (SCUR) and Event Charging with Unit Reservation (ECUR) sessions that the IM handled	getChargingWithUnitReservationSessionCount()
Immediate Event Charging (IEC) sessions that the IM handled	getIecSessionCount()
Sent and received Credit-Control-Requests (CCRs)	getCcrCount()

Table 11–2 (Cont.) RoRuntimeMBean Counters

To Get Total Number of...	Use...
CCRs sent and received with the **Media-Initiator-Flag** AVP set to **CalledParty**	getCcrCalledInitiatorCount()
CCRs sent and received with the **Media-Initiator-Flag** AVP set to CallingParty	getCcrCallingInitiatorCount()
CCRs sent and received with the **Media-Initiator-Flag** AVP set to **Unknown**	getCcrUnknownInitiatorCount()
Sent and received Credit Control Answers (CCAs)	getCcaCount()
Sent and received CCAs with a successful result	getSuccessCcaCount()
Sent and received CCAs with an error result	getErrorCcaCount()
Sent and received Initial requests	getInitialRequestCount()
Sent and received Update CCRs	getUpdateRequestCount()
Sent and received Terminate CCRs	getTerminateRequestCount()
Sent and received Event CCRs	getImmediateRequestCount()
Successfully completed sessions	getCompleteSessionCount()
Error sessions	getErrorSessionCount()
Blocked sessions, that is the number of sessions terminated with the result code 4012	getBlockedSessionCount()
Redirected sessions, that is the number of CCAs with the **Final-Unit-Action** AVP set to **Redirect**.	getRedirectedSessionCount()

For more information on how to access runtime MBeans, see the discussion on monitoring Service Broker using Runtime MBeans in *Oracle Communications Service Broker System Administrator's Guide*.

Getting Statistics on Currently Active Sessions

Using **RoGaugeRuntimeMBean**, you can get the gauge of sessions that IM-OCF or IM-OCF is currently handling.

Online Mediation Controller creates a separate instance of **RoGaugeRuntimeMBean** for each instance of IM-OCF and R-IM-OCF.

The object name of this MBean is **com.convergin:Type=RoGaugeRuntime,Version=***MBean_Version*,**Location=**<*server-name*>,**Name=***IM_Instance_Name*.**Ro**.

Table 11–3 describes the gauge that **RoGaugeRuntimeMBean** provides.

Table 11–3 RoGaugeRuntimeMBean Gauge

To Get Total Number of...	Use...
Currently active sessions	getActiveSessionsGauge()

For more information on how to access runtime MBeans, see the discussion on monitoring Service Broker using Runtime MBeans in *Oracle Communications Service Broker System Administrator's Guide*.

Monitoring the Interfaces in Offline Charging Solutions

In an offline charging solution, IM-OFCF PCP and R-IM-OFCF RADIUS provide the Rf interface.

You can get the following statistics on messages that the IM sent and received through this interface:

- Counters of Diameter Rf accounting requests and accounting answers. See "Getting Statistics on Accounting Requests and Accounting Answers" for more information.
- Gauge for Diameter Rf sessions. See "Getting Statistics on Currently Active Sessions" for more information.

Getting Statistics on Accounting Requests and Accounting Answers

Using **RfRuntimeMBean**, you can get statistics on accounting requests and accounting answers that IM-OFCF PCP or R-IM-OFCF RADIUS sent and received.

Online Mediation Controller creates a separate instance of **RfRuntimeMBean** for each instance of IM-OFCF PCP and R-IM-OFCF PCP.

The object name of this MBean is **com.convergin:Type=RfRuntime,Version=***MBean_Version*,**Location=**<*server-name*>,**Name=***IM_Instance_Name*.**Rf**.

Table 11–4 describes the counters that **RfRuntimeMBean** provides.

Table 11–4 RfRuntimeMBean Counters

To Get Total Number of...	Use...
Sessions handled by the IM	getSessionCount()
Sent and received Accounting Requests (ACR)	getAcrCount()
ACRs sent and received with the Accounting-Record-Type AVP set to START-RECORD	getStartAcrCount()
ACRs sent and received with the Accounting-Record-Type AVP set to INTERIM_RECORD	getInterimAcrCount()
ACRs sent and received with the Accounting-Record-Type AVP set to STOP_RECORD	getStopAcrCount()
ACRs sent and received with the Accounting-Record-Type AVP set to EVENT_RECORD	getEventAcrCount()
Sent and received Accounting Answers (ACAs)	getAcaCount()
ACAs sent and received with the Result-Code AVP < 3000 (success)	getAcaSuccessCount()
ACAs sent and received with the Result-Code AVP >= 3000 (failure)	getAcaErrorCount()
Successfully completed sessions	getCompleteSessionCount()

Table 11–4 (Cont.) RfRuntimeMBean Counters

To Get Total Number of...	Use...
Error sessions	getErrorSessionCount()

For more information on how to access runtime MBeans, see the discussion on monitoring Service Broker using Runtime MBeans in *Oracle Communications Service Broker System Administrator's Guide*.

Getting Statistics on Currently Active Sessions

Using **RfGaugeRuntimeMBean**, you can get the gauge of sessions that IM-OFCF or R-IM-OFCF is currently handling.

Online Mediation Controller creates a separate instance of **RfGaugeRuntimeMBean** for each instance of IM-OFCF PCP and R-IM-OFCF PCP.

The object name of this MBean is **com.convergin:Type=RfgaugeRuntime,Version=***MBean_Version*,**Location=**<*server-name*>,**Name=***IM_Instance_Name*.**Rf**.

Table 11–5 describes the gauge that **RfGaugeRuntimeMBean** provides.

Table 11–5 RfGaugeRuntimeMBean

To Get Total Number of...	Use...
Currently active sessions	getActiveSessionsGauge()

For more information on how to access runtime MBeans, see the discussion on monitoring Service Broker using Runtime MBeans in *Oracle Communications Service Broker System Administrator's Guide*.

Getting Statistics on the Orchestration Engine

Using **OeRuntimeMBean**, you can get the counters of sessions and application triggers.

The object name of this MBean is **com.convergin:Type=OeRuntime,Version=***MBean_Version*,**Location=***Server_Name*,**Name=***IM_instance_name*.**Oe**

Table 11–6 describes the counters that **OeRuntimeMBean** provides.

Table 11–6 OeRuntimeMBean Counters

To Get Total Number of...	Use...
Sessions that the OE handled	getSessionCount()
Successful application triggering that the OE performed	getSuccessfulApplicationTriggeringCount()
Unsuccessful application triggering that the OE attempted to perform	getUnsuccessfulApplicationTriggeringCount()
2xx and 3xx responses that the OE received	getSuccessfulUAServerTriggeringCounter()

Using **OlpRuntimeMBean**, you can get the counter of triggering a specific Orchestration Login Processor (OLP).

The object name of this MBean is **com.convergin:Type=OlpRuntime,Version=***MBean_Version*,**Location=***Server_Name*,**Name=***IM_Instance_Name*.**Olp.***Olp_Name*

Table 11–7 describes the counter that **OlpRuntimeMBean** provides.

Table 11–7 OlpRuntimeMBean Counter

To Get Total Number of...	Use...
Times that the OE triggered the specific OLP	getExecutionCount()

Using **OprRuntimeMBean**, you can get the counter of queries that the Orchestration Logic Processor (OLP) executed.

The object name of this MBean is **com.convergin:Type=OprRuntime,Version=***MBean_Version***,Location=***Server_Name***,Name=***IM_Instance_Name***.Opr.***Opr_Name

Table 11–8 describes the counters that **OprRuntimeMBean** provides.

Table 11–8 OprRuntimeMBean Counters

To Get Total Number of...	Use...
Successful queries that an OPR executed	getSuccessfulQueryCount()
Unsuccessful queries that an OPR attempted to execute	getUnsuccessfulQueryCount()

For more information on how to access runtime MBeans, see the discussion on monitoring Service Broker using Runtime MBeans in *Oracle Communications Service Broker System Administrator's Guide*.

Monitoring the Signaling Domain

A deployment of Online Mediation Controller might involve the following SSUs:

- Diameter SSU. See "Checking the Status of Diameter Network Entities" for more information.
- PCP SSU. See "Checking the Status of PCP Network Entities" for more information.
- WS SSU. See "Checking the Status of Web Service Network Entities" for more information.

Using runtime MBeans, you can check whether the network entity with which the SSU communicates is active.

Checking the Status of Diameter Network Entities

Using **NetworkEntityRuntimeMBean**, you can get the status of the Diameter network entity. Table 11–9 describes the attributes that **NetworkEntityRuntimeMBean** provides.

Table 11–9 NetworkEntityRuntimeMBean Attributes

Attribute	Description
getValue()	The attribute contains the address of the network entity in the URI format where with the colon character (:) is replaced with the underscore.

Table 11–9 (Cont.) NetworkEntityRuntimeMBean Attributes

Attribute	Description
getStatus()	Specifies a network entity status: ■ 0 - Network entity is unavailable ■ 1 - Network entity is available ■ 2 - Status of the network entity is unknown

For more information on how to access runtime MBeans, see the discussion on monitoring Service Broker using Runtime MBeans in *Oracle Communications Service Broker System Administrator's Guide*.

Checking the Status of PCP Network Entities

Using **PcpPaStatisticsMBean**, you can get the status of the BRM application. Table 11–10 describes the attribute that **PcpPaStatisticsMBean** provides.

Table 11–10 PCP SSU Monitoring Attribute

Attribute	Description
getPcpPaStatus()	Specifies the status of the BRM applicatikon. Possible values: ■ 0 - Inactive ■ 1 - Active

For more information on how to access runtime MBeans, see the discussion on monitoring Service Broker using Runtime MBeans in *Oracle Communications Service Broker System Administrator's Guide*.

Checking the Status of Web Service Network Entities

Using **NetworkEntityRuntimeMBean**, you can get the status of the Web Services entity. Table 11–11 describes the attributes that **NetworkEntityRuntimeMBean** provides.

Table 11–11 NetworkEntityRuntimeMBean

Attribute	Description
getValue()	The attribute contains the address of the network entity in the URI format where with the colon character (:) is replaced with the underscore.
getStatus()	Specifies a network entity status: ■ 0 - Network entity is unavailable ■ 1 - Network entity is available ■ 2 - Status of the network entity is unknown

For more information on how to access runtime MBeans, see the discussion on monitoring Service Broker using Runtime MBeans in *Oracle Communications Service Broker System Administrator's Guide*.

Checking Availability of the Diameter and RADIUS Ports

To ensure a stable communication between entities in a Diameter or RADIUS network and Online Mediation Controller, you can check whether the Diameter and RADIUS ports of Online Mediation Controller are up. This check is known as healthcheck.

To perform a healthcheck for a Diameter or RADIUS port:

- Configure the Load Balancer to send to Online Mediation Controller a TCP open request.

  If you check availability of the Diameter port, then after the connection is established, you can receive more detailed information, such as a **Result-Code** AVP on the status of the port, by sending a **Capabilities-Exchange-Request** (CER) and analyzing the received **Capabilities-Exchange-Answer** (CEA).

12

Using the Balance Manager

The Oracle Communications Online Mediation Controller Balance Manager provides a way to enable prepaid subscribers to check their account balances and replenish ("top up") their accounts either from stored payment account information or by redeeming a voucher.

This chapter describes the Balance Manager feature and the Balance Manager API.

About the Balance Manager Feature

The Balance Manager feature is implemented in Online Mediation Controller by a Balance Manager Service.

The Balance Manager Service integrates with online charging systems (OCS) like Oracle Billing and Revenue Management (BRM) and Oracle Communications Elastic Charging Engine (ECE) to access the subscriber billing accounts. Communication between the Balance Manager Service and BRM is over Portal Communications Protocol (PCP) while communication between the Balance Manager Service and ECE uses the ECE API.

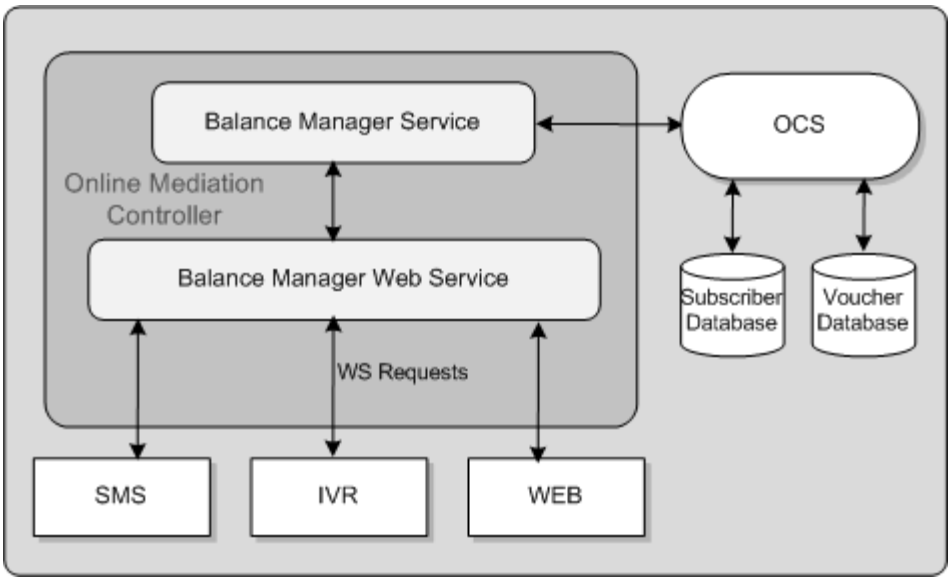
The OCS authenticates the subscribers and maintains the subscriber billing accounts. All subscriber payment information, such as credit card and debit account numbers, is stored by the OCS.

To check their balances, top up their account balances from their credit or debit accounts, and redeem vouchers, subscribers can interact with the Balance Manager applications using Interactive Voice Response (IVR), Short Message Service (SMS), or a Web interface. This chapter describes the procedures for implementing IVR and SMS support. It does not describe implementation of a Web interface, but such an interface would use the same Balance Manager Web Service APIs as the IVR and SMS interfaces.

The Balance Manager Web Service mediates between the IVR, SMS, or Web front ends and the Balance Manager Service. The Balance Manager Service translates the WS messages to PCP or ECE and communicates with BRM or ECE to perform the requested action in the prepaid accounts on the OCS.

Figure 12–1 shows a high level overview of the Balance Manager in context with components that are external to Online Mediation Controller.

Figure 12–1 Balance Manager Overview



Graphic depicts a big box representing Online Mediation Controller. Inside the Online Mediation Controller box from top to bottom are Balance Manager Service and Balance Manager Web Service connected by two-way arrows. The Balance Manager Web Service is connected by two-way arrows labeled "WS Requests" to SMS, IVR and WEB boxes outside the Online Mediation Controller box; these represent the external interfaces.

The Balance Manager Service is connected by two-way arrows outside the Online Mediation Controller box to an oval representing the OCS (BRM or ECE), which itself is connected to two databases: Subscriber DB and Voucher DB.

****

The Balance Manager applications perform four activities, encapsulated in the four methods in the Balance Manager API:

- authenticate: Authenticates subscribers who want to check their balances or add funds to their accounts using payment resources stored on BRM.
- check balance: Checks the balance in a subscriber account.
- top up: Adds funds to ("tops up") an account using the subscriber's payment resources stored on BRM.
- voucher top up: Adds funds to an account by redeeming a voucher.

See "Using the Balance Manager Web Services API" for more information about these methods.

Mid-call Warning

If a subscriber approaches the credit limit configured for his account in the OCS while on a call, the Balance Manager can request that the IVR service play a warning announcement or tone. If the subscriber's payment information is registered in the OCS, the subscriber can top up his account immediately.

Voucher Redemption

The Balance Manager feature supports using vouchers to add funds to a subscriber account. The subscriber redeems the voucher by sending the voucher information (for example, a scratch card number and PIN) to a short-code address supplied by the operator. The vouchers must have been previously registered with the OCS.

Generic Set Up for the Balance Manager

Use of the Balance Manager assumes that the subscriber accounts in the OCS are configured to use one or more payment methods (credit card, direct debit). Payment methods are setup in the OCS.

Use of the voucher top-up feature also assumes that, if vouchers are to be accepted as payment, the OCS has been configured for voucher payments and that the vouchers have been loaded into the OCS database. See the discussion of the Voucher Manager and the Voucher Administration Center in the *Service Integration Components* guide in the BRM documentation set for information about how to set up vouchers in BRM.

In addition to setting up BRM with subscriber payment information and vouchers, you must configure BRM for subscriber authentication.

Also, there are configuration tasks to perform in Online Mediation Controller to enable communication among the various components.

These generic configuration tasks are required for all interfaces to the Balance Manager (IVR, SMS, and Web) and for all the Balance Manager SMS applications: (Balance Manager SMS, Top up App. and Voucher Top-Up).

The generic configuration tasks are as follows:

1. In BRM, create an authentication service. See "Creating an Authentication Service" for details. This step is optional, because you can use an existing authentication service, if one exists, instead of creating a new one.
2. In BRM, create an authentication plan that incorporates the authentication service from step 1. This step is required. See "Creating an Authentication Plan" for details.
3. In BRM, for every subscriber, add the authentication plan to the subscribers' accounts. This step is required. See "Adding the Authentication Plan to Subscriber Accounts" for details.
4. In the Online Mediation Controller Administration Console, configure the authentication service for the application. This step is required. See "Configuring the Authentication Service for the Balance Manager Applications" for details.
5. In the Online Mediation Controller Administration Console, configure the Portal Communications Protocol Signaling Server Unit (PCP SSU) to enable communication between Online Mediation Controller and BRM. This step is required. See "Configuring the PCP SSU".
6. In the Online Mediation Controller Administration Console, configure the Web Services SSU HTTP incoming rule for the Balance Manager. This configuration directs Online Mediation Controller how to route Web Service messages to the Balance Manager. This step is required. See "Configuring the HTTP Incoming Rule for the Web Services SSU" for details.
7. In the Online Mediation Controller Administration Console, configure the credentials to enable clients of the Balance Manager SOAP Web Services to access Balance Manager API. See "Configuring Web Service Client Credentials" for

details.

8. In the Online Mediation Controller Administration Console, configure the Balance Manager Web Service. See "Configuring the Balance Manager Web Service" for details.

In addition, there are several required configuration tasks required only for the Balance Manager application, which provides the SMS interface to the Balance Manager. See "SMS Configuration Tasks" for information about these configuration tasks.

Creating an Authentication Service

BRM must be configured with an authentication service for the Balance Manager to use to authenticate subscribers. You can use an existing BRM authentication service, or you can create a new one.

To create a new authentication service in BRM:

1. Connect to the BRM Developer Center.
2. In the class browser, navigate to **service/authentication**.
3. Create a new storable class with no new fields under **service/authentication**.

   It is not necessary to create any new fields because all the information that the Balance Manager needs are the LOGIN and PASSWORD fields, which already exist in the parent **service** class. The **phone number** parameter passed by the Balance Manager request maps to the LOGIN field in the BRM service and the **PIN** maps to the PASSWD field.
4. Click **OK**.

For more information about creating a new storable class, see the Online Help in the Storable Class Editor in the BRM Developer Center and the discussion of creating custom storable classes in the BRM documentation.

Creating an Authentication Plan

In BRM, create an authentication plan that incorporates the authentication service.

To create an authentication plan:

1. Connect to the BRM Pricing Center.
2. In the Pricing Center application, create a new plan.
3. In the **Plan Attributes** tab, click **Actions** - **New Service**.

   The Service Deal dialog box appears.
4. In the Service Deal dialog box, select **/service/authentication** as the service type and **<No deal>** for the deal.
5. Click **OK**.
6. Click **OK** in the **Plan Attributes** tab to save the authentication plan.

For more information, see the discussion of creating plans in the BRM documentation about setting up rating and pricing.

Adding the Authentication Plan to Subscriber Accounts

Each subscriber account must be associated with the authentication plan.

To associate a subscriber with the authentication plan:

1. Connect to the BRM Customer Center.
2. Select the subscriber in the left panel.
3. In the Purchase Options list, select **Add Plan**.
4. Select the authentication plan to use for the Balance Manager from the list of plans.
5. Click through the **Next** buttons to **Finish**.

For more information about adding a new plan, see the discussion of adding a product with a new service in the BRM Customer Center Help.

Configuring the Authentication Service for the Balance Manager Applications

To configure the authentication service for the Balance Manager feature:

1. In the Online Mediation Controller Administration Console, expand **OCSB**.
2. Expand **Processing Tier**.
3. Expand **Applications**.
4. Click the **Lock and Edit** icon.
5. Expand **Balance Manager**.
6. Click **BRM Authentication**.
7. In the **Authentication Service** field, enter the name of the BRM authentication service that the Balance Manager applications use to authenticate subscribers.

   This is either a pre-existing authentication service in BRM or the one that you created for the Balance Manager. See "Creating an Authentication Service".
8. Click **Apply**.

Configuring the PCP SSU

The PCP SSU must be configured to enable communication between Online Mediation Controller and BRM.

To configure the PCP SSU, follow the instructions in the discussion of configuring PCP signaling server units in the *Oracle Communications Service Broker Signaling Server Units Configuration Guide*.

Configuring the HTTP Incoming Rule for the Web Services SSU

To set the HTTP incoming rule for the WS SSU:

1. In the Online Mediation Controller Administration Console, expand **OCSB**.
2. Expand **Signaling Tier**.
3. Click the **Lock and Edit** icon.
4. Expand **SSU Web Services**.
5. Click **General**.
6. Click the **SSU WS** tab.
7. Click the **Incoming Routing Rules** subtab.
8. Click **New**.

9. In the dialog box, set the incoming routing rules as follows:
   - Set the Name field to an incoming rule name of your choice.
   - Set the **Service Name** field to **BalanceManagerService**.
   - Set the Alias field to **ssu:topup@ocsb/topup**.
10. Click **OK**.

For more information, see the discussion of configuring routing rules for incoming Web Service messages in *Oracle Communications Service Broker Signaling Server Units Configuration Guide*.

Configuring Web Service Client Credentials

You can set up credentials to authenticate external Web Services clients for access to the Balance Manager Web Service.

To configure credentials for the Web Service:

1. In the Online Mediation Controller Administration Console, expand **OCSB**.
2. Expand **Signaling Tier**.
3. Click the **Lock and Edit** icon.
4. Expand **SSU Web Services**.
5. Click **General**.
6. Click the **SOAP** tab.
7. Click the **Credential Store** subtab.
8. Enter credentials for the Web Service client. See the discussion of authenticating SOAP requests with WSSE UsernameToken credentials in *Oracle Communications Service Broker Signaling Server Units Configuration Guide* for detailed instructions on configuring credentials.

Credential checking is optional and disabled by default. To enable it, set the **Authentication Method** field in the Balance Manager endpoint to USERNAME_TOKEN, as described in the "Configuring the Balance Manager Web Service" section.

Configuring the Balance Manager Web Service

To configure the Balance Manager SOAP Web Service:

1. In the Online Mediation Controller Administration Console, expand **OCSB**.
2. Expand **Signaling Tier**.
3. Click the **Lock and Edit** icon.
4. Expand **SSU Web Services**.
5. Click **Balance Manager**.
6. Click the **Balance Manager** tab.
7. Click the **End Point** subtab.
8. Click **New**.
9. In the dialog box, set the Balance Manager endpoint as follows:
   - Set the **URI** field to **/BalanceManagerService**.

- Set the **Authentication Method** field to either **NONE** or **USERNAME_TOKEN**.

  The USER_NAME token is for WSSE Username Token security. As an alternative, you can authenticate credentials using HTTP Basic Auth. See the discussion of HTTP server security contexts in *Oracle Communications Service Broker Signaling Server Units Configuration Guide* for information about using Basic Auth.

- If the **Authentication Method** field is **USERNAME_TOKEN**, set the **Username** field to the user name of the service. This value is ignored if the **Authentication Method** is **NONE**.

- If the **Authentication Method** field is **USERNAME_TOKEN**, set the **Credential Key** field to the key that you configured for the service in the Credential Store. This value is ignored if the **Authentication Method** is **NONE.**

10. Click **OK**.

Using the Balance Manager IVR Interface

Subscribers can check their balances and top up their accounts from their phone keypads using dual-tone multifrequency (DTMF) technology. They can add value using the credit or debit information associated with their accounts, or they can redeem a voucher to add value to their own or any other prepaid account.

The IVR implementation uses voice xml (vxml) scripts to play prompts and to gather DTMF input from the subscriber. Examples of these prompts are "Please enter the 10 digit telephone number.", "Please enter the amount you want add.", "You have entered an invalid telephone number pin combination.", and so on.

IVR Components

The following components, external to Online Mediation Controller, are required to interact with the Balance Manager Web Service to support IVR interaction:

- IVR Web Server

  The IVR Web Server acts as a document server for the vxml scripts and as a Web Service client to the Balance Manager Web Service.

  The vxml scripts are hosted in a servlet or JSP container to enable communication with the Top Up Web service.

- Media Server

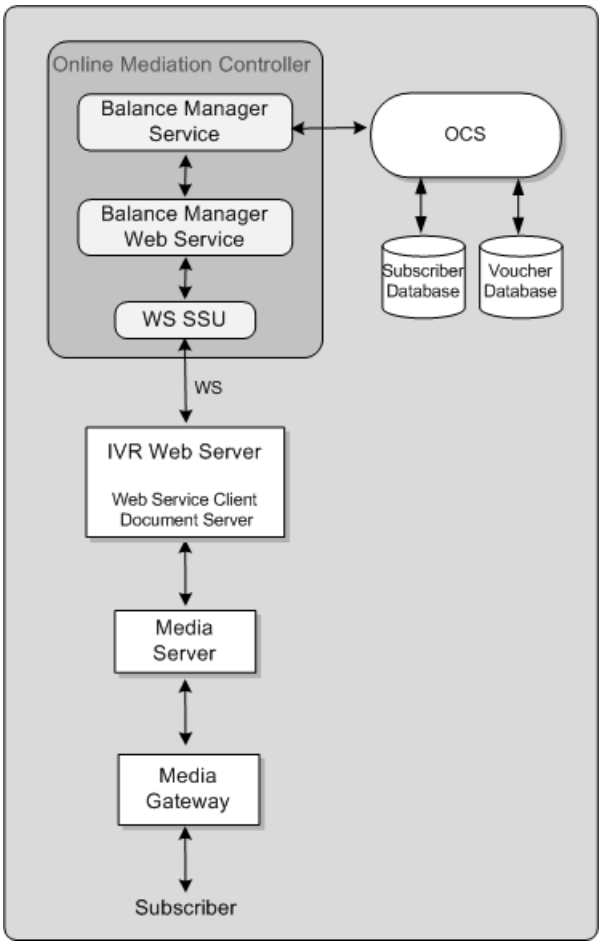
  The media server plays the vxml scripts. Operators can use the media server of their choice.

- Media Gateway

  The Media Gateway processes the incoming IVR communication from the caller and generates the SIP call with the initial vxml script.

The WS SSU in Online Mediation Controller receives the Web Service requests and routes them to the Balance Manager Web Service for processing. Figure 12–2 displays the described components in the Online Mediation Controller IVR workflow.

Figure 12–2 IVR Workflow



Graphic depicts IVR workflow as follows:

At the top, a box representing Online Mediation Controller, which contains (from top to bottom) Balance Manager Service, Balance Manager Web Service and WS SSU all connected by two-way arrows. To the right and outside of the Online Mediation Controller box from the Balance Manager Service, two way arrows connect to a OCS such as BRM or ECE, and its subscriber and voucher databases.

Two way arrows point down and out of the Online Mediation Controller box from the WS SSU to the IVR Web Server, then Media Server, then Media Gateway to the Subscriber at the bottom.

****

IVR Web Service Client

The IVR Web Service client initiates IVR interaction by sending a SIP INVITE to the media server. This INVITE contains the URI of the first vxml script to invoke.

The Web Service client implements the logic to control the flow of vxml scripts to the media server. Based on interactions with the subscriber and responses from the Balance Manager Web Service, the Web Service client sends the next logical vxml script for the media server to play.

You implement the IVR Web Service client to work with the media server of your choice. Configure the media server to invoke the initial vmxl script for an incoming Balance Manager request; for example:

http://*webserver_ host*:*port*/BalanceManager/welcome.vxml

Online Mediation Controller includes a sample IVR Web Service client implementation as source code that you can compile and run. The sample is built on WebLogic server using the Voxeo Prophecy media server to play the scripts.

The sample source code and supporting files are found in the following file:

Oracle_Home/**samples/topup.ivr.sample.source.zip**

To use the sample, unzip the sample archive and follow the instructions in the **README** file included in the archive.

Using the Balance Manager SMS Interface

Subscribers can check their balances, top up their accounts, and redeem vouchers by sending SMS messages to and receiving them from the following Balance Manager SMS applications:

- Balance Manager SMS
- Top up App
- Voucher Top-Up

These applications reside within Online Mediation Controller and interact with various Online Mediation Controller components.

SMS Workflow and Components

The workflow for sending an SMS request to the Balance Manager SMS applications is as follows:

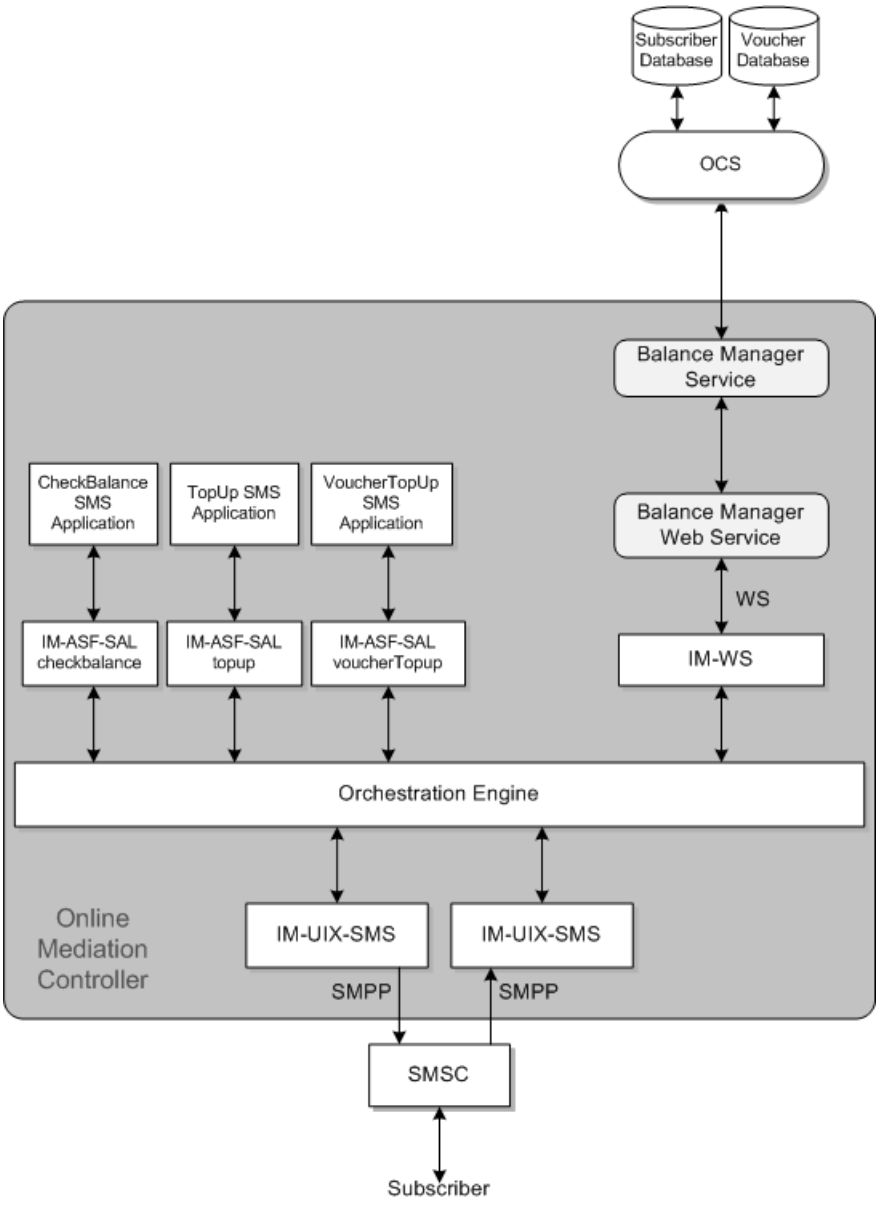
1. The subscriber sends an SMS message requesting one of the Balance Manager services: check balance, top up, or redeem voucher.
2. The SMSC receives the SMS message from the subscriber and sends it to the IM-UIX-SMS module inside Online Mediation Controller.
3. The IM-UIX-SMS receives the request from the SMSC and forwards it to the Orchestration Engine.
4. The Orchestration Engine routes the request to the appropriate IM-ASF-SAL module.
5. The IM-ASF-SAL module sends the request to the appropriate Balance Manager SMS application.
6. The application processes the request to create the appropriate calls to the Balance Manager Web Service API. It sends the request back up through the IM-ASF_SAL through the Orchestration Engine, where the OE routes the request to the IM-WS.
7. The IM-WS sends the request to the Balance Manager Web Service.
8. The Balance Manager Web Service sends the request to the Balance Manager Service, which handles the fulfillment of the request on BRM.
9. After the request is processed, the Balance Manager Web Service sends the result to the appropriate IM-ASF-SAL.

In the case of check balance, the result is the current balance. For topup and voucher top up, it is the success or failure to increment the subscriber's credit balance.

10. The IM-ASF-SAL sends the results by SMS message to the SMSC through the Orchestration Engine and the IM-UIX-SMS module.

Figure 12–3 shows the SMS workflow and components.

Figure 12–3 SMS Workflow



A large box in the middle of the graphic represents Online Mediation Controller. Inside the upper right area in this box are smaller boxes vertically stacked and connected by bi-directional arrows representing from top to bottom: Balance Manager Service, Balance Manager Web Service, IM-WS. The Balance Manager Service at the top is connected by bi-directional arrows outside the Online Mediation Controller box to an online charging system, such as BRM, which is depicted as attached to two databases: Subscriber DB and Voucher DB. The IM-WS at the bottom of the stack is

connected by bi-directional arrows to a long horizontal box that runs the entire width of the Online Mediation Controller box, which represents the Orchestration Engine.

In the upper left area of the Online Mediation Controller box, bi-directional arrows connect the Orchestration Engine to three boxes horizontally distributed above it. These boxes represent the three IM-ASF_SALs: checkbalance, topup, and voucher topup. These three IM-ASF_SAL boxes are connected by bi-directional arrows each to a second set of horizontally distributed boxes representing the corresponding applications: CheckBalance SMS Application, TopUp SMS Application, Voucher TopUp SMS Application. All of these are inside the Online Mediation Controller box.Below the Orchestration Engine bi-directional arrows point to two boxes, one representing the IM-UIX-SMS for incoming SMPP traffic and the other the IM-UIX-SMS for outgoing SMPP traffic. The incoming IM-UIX-SMS is connected by a uni-directonal arrow pointing down outside the Online Mediation Controller box to the SMSC. The outgoing IM-UIX-SMS is connected by a uni-directional arrow pointing up from the SMSC. The SMSC is shown connected to the Subscriber by a bi-directional arrow.

****

SMS Configuration Tasks

The SMS interface to the Balance Manager requires additional configuration of several Online Mediation Controller components in addition to the tasks described in the "Generic Set Up for the Balance Manager" section.

The following configuration tasks are required if you are using the Balance Manager SMS application. These tasks are all performed in the Online Mediation Controller Administration Console.

1. Add and configure two IM-UIX-SMS SMPP instances to process requests to and from the SMSC. See "Setting Up the IM-UIX-SMS Module" for details.
2. Configure the SMPP SSUs to set up routing of submit_sm and deliver_sm requests to the addresses of the three SMS applications. See "Configuring the SMPP Signaling Server Unit (SMPP SSU)" for details.
3. Create three IM-ASF-SAL instances, one for each application, to enable the applications to interact with the other Online Mediation Controller modules. See "Creating and Configuring the IM-ASF-SALs" for details.
4. Create an IM-WS instance for the Balance Manager. This instance enables the Balance Manager to receive messages from and send messages to Web Services using SOAP and REST interfaces. See "Creating and Configuring an IM-WS" for details.
5. Create an outgoing rule to define how the Web Services SSU routes outgoing Balance Manager messages. "Creating an Outgoing Routing Rule for the Web Services SSU" for details.
6. Set up the orchestration logic, which tells the Orchestration Engine the order in which to invoke the applications. See "Setting Up the Orchestration Logic for Balance Manager" for details.
7. Configure the text of outgoing messages in the Balance Manager SMS application. See "Configuring the Balance Manager SMS Application Messages" for details.

Setting Up the IM-UIX-SMS Module

The SMS interface to the Balance Manager SMS applications requires two IM-UIXSMS-SMPP instances: one to receive requests from the SMSC and another to send responses to the SMSC. If these modules do not exist, you must create them.

To create an IM-UIX-SMSSMPP instance to handle SMS requests:

1. In the Administration Console, expand **OCSB**.
2. Click the **Lock and Edit** icon.
3. Expand **Processing Tier**.
4. Expand **Interworking Modules**.
5. Click **IM Management**.
6. Click **New**.
7. From the Type menu, select **IMUIXSMSSMPP34**.
8. From the Version menu, select **2.0.0.0**.
9. In the Name field, enter a name for the instance; for example: imsmpp_in.
10. Click **OK**.
11. Click **Commit** to commit your changes.

Repeat this procedure to create another instance to handle responses; for example imsmpp_out.

Then configure both IM-UIX-SMSSMPP instances.

To configure an IM-UIX-SMSSMPP instance:

1. In the Administration Console, expand **OCSB**.
2. Click the **Lock and Edit** icon.
3. Expand **Processing Tier**.
4. Expand **Interworking Modules**.
5. Select the instance that you want to configure.
6. See the discussion of configuring IM-UIX-SMS in *Oracle Communications Service Broker Modules Configuration Guide* for detailed instructions for configuring the IM-UIX-SMSSMPP instances.
7. Click **Apply**.

Configuring the SMPP Signaling Server Unit (SMPP SSU)

The SMPP SSU provides Online Mediation Controller with access to the SMSC over SMPP.

In this module, configure:

- three incoming routing rules, one for each application
- an SMPP network entity
- a connection to the SMSC

The incoming routing rule defines the IM-UIX-SMS instance to which the SMPP SSU routes deliver_sm messages from the SMSC.

To create the incoming routing rules:

1. In the Administration Console, expand **OCSB**.
2. Click the **Lock and Edit** icon.
3. Expand **Signaling Tier**.
4. Click **SSU SMPP**.
5. Click the **SSU SMPP** tab.
6. Click the **Incoming Routing Rules** subtab.
7. For each of the three Balance Manager SMS applications, do the following:
   a. Click **New**.
   b. In the dialog box, provide a name for the incoming rule in the Name field; for example: CheckBalance, TopUp, VoucherTopUp.
   c. In the SMPP Destination Address field, enter the IP address of the SMSC that you configured in the IFC as the incoming SMSC for the Balance Manager. This is the destination address to be used in deliver_sm messages to the application. The Balance Manager SMS applications have different IP addresses.
   d. In the Service Type field, enter **SMS**.
   e. In the Alias field, enter the value that you configured for the Default SMSC Alias field of the incoming IM-UIXSMSSMPP instance.
   f. Click **OK**.

See the discussion of configuring incoming routing rules in the configuring SMPP sign in *Oracle Communications Service Broker Signaling Server Units Configuration Guide* for more information.

The SMPP network entity defines the SMSC to which the SMPP SSU routes submit_sm messages generated by the IM-UIX-SMS.

To create the network entity:

1. In the Administration Console, expand **OCSB**.
2. Click the **Lock and Edit** icon.
3. Expand **Signaling Tier**.
4. Click **SSU SMPP**.
5. Click the **SSU SMPP** tab.
6. Click the **SMPP Network Entities** subtab.
7. Click **New**.
8. See the discussion of configuring SMPP network entities in *Oracle Communications Service Broker Signaling Server Units Configuration Guide* for instructions on configuring the network entity.
9. Click **OK**.

The connection to the SMSC specifies the IP address and port to connect to, and other connection parameters.

To configure a connection to the SMSC:

1. In the Administration Console, expand **OCSB**.
2. Click the **Lock and Edit** icon.

3. Expand **Signaling Tier**.
4. Click **SSU SMPP**.
5. Click the **SMPP** tab.
6. See the discussion of setting up SMSC connections in *Oracle Communications Service Broker Signaling Server Units Configuration Guide* for detailed instructions on setting up SMSC connections.

Creating and Configuring the IM-ASF-SALs

An IM-ASF-SAL enables the Balance Manager to interface with the other Online Mediation Controller components. Create three IM-ASF-SALs, one for each Balance Manager SMS application.

To create the IM-ASF-SALs:

1. In the Administration Console, expand **OCSB**.
2. Click the **Lock and Edit** icon.
3. Expand **Processing Tier**.
4. Expand **Interworking Modules**.
5. Click **IM Management**.
6. For each of the three Balance Manager SMS applications, do the following:
   a. Click **New**.
   b. From the **Type** menu, select **IMASFSAL**.
   c. From the **Version** menu, select **2.0.0.0**.
   d. In the **Name** field, enter a name for the instance; for example: imCheckBalance, imTopUp, imVoucherTopup.
   e. Click **OK**.

To configure the Balance Manager IM-ASF-SALs:

1. In the Administration Console, expand **OCSB**.
2. Expand **Processing Tier**.
3. Expand **Interworking Modules**.
4. For each of the three Balance Manager SMS applications, do the following:
   a. In the left panel of the console, select one of the IMASFSAL instances created for the Balance Manager SMS applications.

      The configuration form for that IM-ASF-SAL appears in the right panel.
   b. Click the **Application Server** tab.
   c. In the **SAL Application Address** field, enter the appropriate SIP URI of the Online Mediation Controller application to which to connect this IM-ASF -SAL. The choices are: **sip:checkbalance@oracle.com**, **sip:topup@oracle.com**, or **sip:voucherTopup@oracle.com**.
   d. From the **SAL Mode** field, select **INLINE**.
   e. Configure the fields in the **Session Keep Alive** and **SAL** tabs as appropriate for your installation.

For information on the fields in the **Session Keep Alive** and **SAL** tabs, see the discussion of configuring IM-ASF-SAL in *Oracle Communications Service Broker Modules Configuration Guide*

f. Click **Apply**.

5. After you have configured all three Balance Manager SMS applications, commit your changes to save the configuration and deploy it to the managed servers.

Creating and Configuring an IM-WS

The IM-WS enables the Balance Manager to receive messages from and send messages to Web Services using Simple Object Access Protocol (SOAP).

To create the IM-WS:

1. In the Administration Console, expand **OCSB**.
2. Click the **Lock and Edit** icon.
3. Expand **Processing Tier**.
4. Expand **Interworking Modules**.
5. Click **IM Management**.
6. Click **New**.
7. In the dialog box, select **IMWS** from the Type menu.
8. Select **2.0.0.0** for the version.
9. Assign a name to the instance; for example: imwsbalmgr.
10. Click **OK**.
11. Click the **Commit** icon to commit your changes.

To configure the IM-WS:

1. In the Administration Console, expand **OCSB**.
2. Expand **Processing Tier**.
3. Expand **Interworking Modules**.
4. In the left panel, select the IMWS instance created for the Balance Manager.

   The configuration form for that IMWS appears in the right panel.
5. Click the **WebService** tab.
6. In the Web Service Alias field, enter **topup**.
7. From the Web Service type menu, select **SOAP**.
8. In the Web Service Body Type field, enter **topuprequest**.
9. Click **Apply**.

Creating an Outgoing Routing Rule for the Web Services SSU

The outgoing routing rule for the Web Services SSU defines how messages are routed from the Balance Manager Service.

This task assumes that the network access point on which the Web Services SSU listens for HTTP traffic has been previously configured. See the discussion of configuring HTTP server network access settings in *Oracle Communications Service Broker Signaling Server Units Configuration Guide* for more information.

To create an outgoing routing rule for the Web Services SSU:

1. In the Administration Console, expand **OCSB**.
2. Click the **Lock and Edit** icon.
3. Expand **Signaling Tier**.
4. Expand **SSU Web Services**.
5. Click **General**.
6. Click the **SSU WS** tab.
7. Click the **Outgoing Routing Rules** subtab.
8. Click **New**.
9. In the **Name** field, assign a name to the rule.
10. In the **Alias** field, enter **topup**.
11. In the **Web Service URI** field, enter the URI of your Balance Manager Web Service in the form:

    http://*BalanceManagerWebServiceIPAddress:port***/soap/BalanceManagerService**
12. See the discussion of configuring routing rules for outgoing Web Services in *Oracle Communications Service Broker Signaling Server Units Configuration Guide* for information on configuring the remaining fields.
13. Click **OK**.

Setting Up the Orchestration Logic for Balance Manager

You must inform the Orchestration Engine of the order in which to invoke the Balance Manager components. The order is:

1. The message is sent from the IM-UIX-SMSSMPP module that receives incoming SMS messages from the SMSC to the appropriate IM-ASF-SAL module.
2. Normally, the message is sent from the IM-ASF-SAL module to the Balance Manager Web Service. However, if there is a problem with the request, no message is sent to the Balance Manager Web service.
3. The message is sent from the IM-ASF-SAL module to the IM-UIX-SMSSMPP module that sends results MSM messages to the SMSC.

First, edit the orchestration logic in the Orchestration Engine and then update the subscribers' orchestration logic with the new flow.

You can add the orchestration logic for the Balance Manager using Orchestration Studio user interface or you can define the Initial Filter Criteria directly in an **ifc.xml** file that you edit directly with a text editor. See *Oracle Communications Service Broker Orchestration User's Guide* for information on setting up orchestration logic using either method.

The criteria in the i**fc.xml** file are based on the TO header and the MESSAGE method. To access the Balance Manager Web Service, use one or more of the following: getBalance, voucherTopup, topup.

The orchestration logic associated with a subscriber is stored as part of the subscriber profile. After you have updated **ifc.xml** for the Balance Manager components, incorporate it into the `<ifcProfileData>` element in the subscriber profile using the **updateSubscriber** operation. See the Subscriber Store API Reference in *Oracle*

Communications Service Broker Subscriber Store User's Guide for information about this operation.

The IP addresses for the three applications in the **ifc.xml** file should be set to the same values as the IP addresses that you configured for the incoming routing rules in the SMPP SSU.

Configuring the Balance Manager SMS Application Messages

This step configures the outgoing messages that will be displayed to the end user in response to requests to check a balance, top up an account from a credit card or bank account, or redeem a voucher.

To configure the Top up App application messages:

1. In the Administration Console, expand **OCSB**.
2. Click the **Lock and Edit** icon.
3. Expand **Processing Tier**.
4. Expand **Applications**.
5. Click **Top up App**.

   The Top up App configuration form appears in the right panel.
6. Click the **Top up configuration** tab.
7. In the Success Message field, enter the text for the message that the user will receive if the request succeeds.

   The only information available to the application is whether the operation succeeded or failed.
8. In the Error Message field, enter the text for the message that the user will receive if the request fails.
9. In the Content parse field enter the regular expression to use to parse the incoming message.

   The regular expression describes how the content is divided between the amount and the PIN. For top up, it should describe two groups of digits, one for the amount and one for the PIN. For example, **(\d+\.?\d):(\d+)** describes two groups, separated by a colon. The first group, representing the amount, contains any number of digits followed by an optional dot followed by any number of digits. The second group, representing the PIN describes any number of digits
10. Click **Apply**.

To configure the Voucher Top-up application messages:

1. In the Administration Console, expand **OCSB**.
2. Click the **Lock and Edit** icon.
3. Expand **Processing Tier**.
4. Expand **Applications**.
5. Click **Voucher Top-Up**.

   The Voucher Top-Up configuration form appears in the right panel.
6. Click the **Voucher Top up configuration** tab.
7. In the Success Message field, enter the text for the message that the user will receive if the request succeeds.

The only information available to the application is whether the operation succeeded or failed.

8. In the Error Message field, enter the text for the message that the user will receive if the request fails.
9. In the Content parse field enter the regular expression to use to parse the incoming message.

   The regular expression describes how the content is divided between the voucher number and the voucher PIN. For voucher redemption, it should describe two groups of digits, one for the voucher and one for the voucher PIN. For example, **(\d+):(\d+)** describes two groups of any number of consecutive digits separated by a colon.
10. Click **Apply**.

To configure the Balance Manager SMS application messages:

1. In the Administration Console, expand **OCSB**.
2. Click the **Lock and Edit** icon.
3. Expand **Processing Tier**.
4. Expand **Applications**.
5. Expand **Balance Manager**.
6. Expand **SMS**.
7. Click **Check Balance**.
8. In the **Success Message** field, enter the text for the message that the user will receive if the request succeeds. You can use **%s** as the placeholder for the amount, which the Balance Manager will supply; for example, "Your new balance is %s".
9. In the **Error Message** field, enter the text for the message that the user will receive if the request fails.
10. In the **Request SMS Format** field, enter the regular expression to use to parse the incoming message.

    The regular expression describes the part of the content that is the PIN. For example, **\d+** describes any number of consecutive digits.
11. Click **Apply**.

Using the Balance Manager Web Services API

The following Balance Manager Web Service APIs support the basic Balance Manager functionality through SOAP Web Services:

- authenticate
- getBalance
- topUp
- voucherTopup

The Balance Manager SOAP Web Service Definition Language (WSDL) file is by default located in:

http://*host***:***port***/soap/BalanceManagerService?wsdl**

authenticate

Verifies that the subscriber phone number and PIN are valid.

If the phone number and pin combination is not valid, throws an AuthenticationException. To ensure security, the AuthenticationException has no attributes that explain why the authentication failed.

Request Parameters

The request body parameters are:

- **phonenumber**: (String) Required. Phone number of the subscriber account to authenticate. Corresponds to the LOGIN value in the subscriber's account on BRM.
- **PIN**: (String) Required. PIN to authenticate. Corresponds to the PASSWORD value in the subscriber's account on BRM.

Response Parameters

None

Error Response

The error responses are:

- AuthenticationFault: Authentication failed. For security no information about why authentication failed is provided.

```
<element name="AuthenticationFault" type="tns:AuthenticationFault" />
<complexType name="AuthenticationFault">
    <sequence />
</complexType>
```

- ServiceException: Failed for reasons other than invalid phonenumber and pin combination.

```
<element name="ServiceException" type="tns:ServiceException" />
<complexType name="ServiceException">
    <sequence>
        <xsd:element name="messageId" type="xsd:string" />
        <xsd:element name="text" type="xsd:string" />
        <xsd:element name="variables" type="xsd:string"
            minOccurs="0" maxOccurs="unbounded" />
    </sequence>
 </complexType>
```

Example

Example 12–1 authenticate request

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:rcc="http://oracle/ocsb/app/rcc">
   <soapenv:Header/>
   <soapenv:Body>
      <rcc:authenticate>
         <rcc:phonenumber>16505067000</rcc:phonenumber>
```

```
            <rcc:PIN>O3r5a7c</rcc:PIN>
         </rcc:authenticate>
      </soapenv:Body>
   </soapenv:Envelope>
```

getBalance

Gets the current balance for the specified account.

Performs authentication using the authenticate operation before accessing the account information.

Request Parameters

The request body parameters are:

- **phonenumber**: (String) Required. Phone number associated with the account. Corresponds to the LOGIN value in the subscriber's account on BRM.
- **PIN**: (String) Required. PIN associated with the account. Corresponds to the PASSWORD value in the subscriber's account on BRM.
- **showCurrencyInfo**: (Boolean) Required. If **true**, currency details associated with the account are included in the returned BalanceType object. If **false**, currency details are not included.

Response Parameters

The response body parameter is:

balanceType: (balanceType) Object showing the balance in the subscriber's account.

The balanceType object encapsulates credit balance information associated with a subscriber account:

```
<complexType name="balanceType">
    <sequence>
        <!-value of the account balance-->
        <element name="amount" type="xsd:double" />
        <!-BRM resourceId corresponding to this balanceType-->
        <element name="type" type="xsd:int" />
        <!-object describing the currency of the balance-->
        <element name="details" type="tns:currencyDetails" minOccurs="0" />
    </sequence>
</complexType>
```

The **currencyDetails** object describes the currency associated with the account.

```
<complexType name="currencyDetails">
    <sequence>
        <!-name of the currency-->
        <element name="name" type="xsd:string" />
        <!-ISO 3166 currency code-->
        <element name="code" type="xsd:string" />
        <!-symbol representing the currency such as $ or €-->
        <element name="symbol" type="xsd:string" />
    </sequence>
</complexType>
```

Error Response

The error responses are:

- AuthenticationFault: Authentication failed. For security no information about why authentication failed is provided.

```
<element name="AuthenticationFault" type="tns:AuthenticationFault" />
<complexType name="AuthenticationFault">
    <sequence />
</complexType>
```

- ServiceException: Failed for reasons other than invalid phonenumber and pin combination.

```
<element name="ServiceException" type="tns:ServiceException" />
<complexType name="ServiceException">
    <sequence>
        <xsd:element name="messageId" type="xsd:string" />
        <xsd:element name="text" type="xsd:string" />
        <xsd:element name="variables" type="xsd:string"
            minOccurs="0" maxOccurs="unbounded" />
    </sequence>
</complexType>
```

Examples

Example 12–2 getBalance request

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:rcc="http://oracle/ocsb/app/rcc">
   <soapenv:Header/>
   <soapenv:Body>
      <rcc:getBalance>
         <rcc:phonenumber>16505067000</rcc:phonenumber>
         <rcc:PIN>O3r5a7c</rcc:PIN>
         <rcc:showCurrencyInfo>true</rcc:showCurrencyInfo>
      </rcc:getBalance>
   </soapenv:Body>
</soapenv:Envelope>
```

Example 12–3 getBalanceResponse

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:rcc="http://oracle/ocsb/app/rcc">
   <soapenv:Header/>
   <soapenv:Body>
      <rcc:getBalanceResponse>
         <rcc:balance>
            <rcc:amount>346.20</rcc:amount>
            <rcc:type>100001</rcc:type>
            <rcc:details>
               <rcc:name>United States dollar</rcc:name>
               <rcc:code>USD</rcc:code>
               <rcc:symbol>$</rcc:symbol>
            </rcc:details>
         </rcc:balance>
      </rcc:getBalanceResponse>
   </soapenv:Body>
</soapenv:Envelope>
```

topUp

Adds the specified amount to the specified subscriber account.

The funds are supplied by the subscriber's predefined payment arrangement in BRM. No credit card or debit account information is transmitted by this method.

Performs authentication using the authenticate operation before adding funds to the account.

Request Parameters

The request body parameters are:

- **phonenumber**: (String) Required. Phone number associated with the account. Corresponds to the LOGIN value in the subscriber's account on BRM.
- **PIN**: (String) Required. PIN associated with the account. Corresponds to the PASSWORD value in the subscriber's account on BRM.
- **amount**: (double) Required. Amount to add to the account.

Response Parameters

None.

Error Response

The error responses are:

- AuthenticationFault: Authentication failed. For security no information about why authentication failed is provided.

```
<element name="AuthenticationFault" type="tns:AuthenticationFault" />
<complexType name="AuthenticationFault">
    <sequence />
</complexType>
```

- ServiceException: Failed for reasons other than invalid phone number and pin combination.

```
<element name="ServiceException" type="tns:ServiceException" />
<complexType name="ServiceException">
    <sequence>
        <xsd:element name="messageId" type="xsd:string" />
        <xsd:element name="text" type="xsd:string" />
        <xsd:element name="variables" type="xsd:string"
            minOccurs="0" maxOccurs="unbounded" />
    </sequence>
</complexType>
```

Example

Example 12–4 topUp Request

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:rcc="http://oracle/ocsb/app/rcc">
   <soapenv:Header/>
   <soapenv:Body>
      <rcc:topup>
```

```
            <rcc:phonenumber>16505067000</rcc:phonenumber>
            <rcc:PIN>O3r5a7c</rcc:PIN>
            <rcc:amount>100</rcc:amount>
         </rcc:topup>
      </soapenv:Body>
   </soapenv:Envelope>
```

voucherTopup

Redeems a voucher to add the voucher value to the specified subscriber account.

This operation does not perform authentication. Users can redeem a voucher to add funds to accounts that are not their own.

Request Parameters

The request body parameters are:

- **phonenumber:** (String) Required. Phone number associated with the account to which the voucher amount is added. Corresponds to the LOGIN value in the subscriber's account on BRM.
- **voucherNumber**: (String) Required. Number of the voucher.
- **voucherPin**: (double) Required. PIN for the voucher.

Response Parameters

None.

Error Response

The error response is:

- ServiceException: Generic exception

```
<element name="ServiceException" type="tns:ServiceException" />
<complexType name="ServiceException">
    <sequence>
        <xsd:element name="messageId" type="xsd:string" />
        <xsd:element name="text" type="xsd:string" />
        <xsd:element name="variables" type="xsd:string"
            minOccurs="0" maxOccurs="unbounded" />
    </sequence>
</complexType>
```

Example

Example 12–5 voucherTopup request

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:rcc="http://oracle/ocsb/app/rcc">
   <soapenv:Header/>
   <soapenv:Body>
      <rcc:voucherTopup>
         <rcc:phonenumber>16505067000</rcc:phonenumber>
         <rcc:vouchernumber>0020031003</rcc:vouchernumber>
         <rcc:voucherpin>6T3#e</rcc:voucherpin>
      </rcc:voucherTopup>
   </soapenv:Body>
</soapenv:Envelope>
```

13

Configuring the Announcement Player Application

This chapter describes how you configure the Announcement Player application in Oracle Communications Online Mediation Controller.

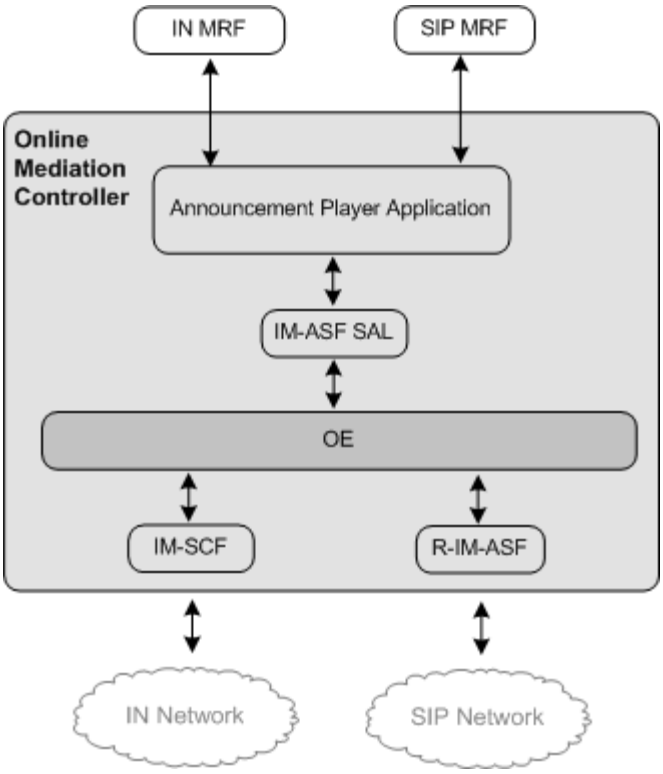
About the Announcement Player

The Announcement Player is a Online Mediation Controller application that plays an announcement at the beginning of a call to a calling party. For example, you can use the Announcement Player to play a welcome announcement to a new subscriber when this subscriber is making the first call.

The Announcement Player communicates with the Orchestration Engine (OE) through IM-ASF SAL interworking module. To play announcements, the Announcement Player uses the Media Resource Function (MRF) that you defined when configuring the SIP SSU.

Figure 13–1 shows the place of the Announcement Player in the overall architecture of Online Mediation Controller. Depending on the subscriber's network, you must set up an instance of either IM-SCF (for IN networks) or R-IM-ASF (for SIP networks). The Announcement Player can communicate with both IN MRF and SIP MRF.

Figure 13–1 Announcement Player Application



Note: The Announcement Player can be used with Online Mediation Controller Policy Controller only. You must have Policy Controller installed to use the Announcement Player.

Like any other application, the Announcement Player is a part of the orchestration flow. You need to configure the orchestration flow in a way that triggers the Announcement Player when the OE receives an INVITE message. After the Announcement Player has finished playing an announcement, the OE routes the session to the next application in the orchestration flow.

To allow the OE to trigger the Announcement Player, you need to perform the following steps:

1. Set up the instance of IM-ASF SAL and MRF. See "Setting Up the Announcement Player" for more information.
2. Specify the MRF alias and the code of the announcement to be played. See "Configuring the Announcement Player" for more information.

Setting Up the Announcement Player

To set up the Announcement Player:

1. Create an instance of IM-ASF SAL. See the "Managing Interworking Modules" chapter in *Oracle Communications Service Broker Modules Configuration Guide* for more information.
2. In the **Application Server** tab, set the parameters as follows:

- In the **SAL Application Address** field, type **sip:announcementplayer@oracle.com**.
- From the **SAL Mode** list, select **INLINE**.

3. Specify configuration parameters on the **Session Keep Alive** and **SAL** tabs as required. See the "Configuring IM-ASF SAL" chapter in *Oracle Communications Service Broker Modules Configuration Guide* for more information.
4. Using the Orchestration Studio, add the newly created instance of IM-ASF SAL to the orchestration flow. See the "Specifying IMs" section of the "Building an Orchestration Logic Flow" chapter in the *Oracle Communications Service Broker Orchestration User's Guide* for more information.
5. To trigger the Announcement Player when it receives an INVITE message, in the Orchestration Studio, add the condition for routing the session to the IM-ASF SAL and check whether the SIP Method is INVITE. See the "Adding Conditions" section of the "Building an Orchestration Logic Flow" chapter in the *Oracle Communications Service Broker Orchestration User's Guide* for more information.
6. Using the **SIP Network Entities** tab in the SSU SIP configuration screen, specify the alias and the physical address of the MRF that the Announcement Player uses. See the "Configuring SIP Network Entities" section of the "Configuring SIP Signaling Server Units " chapter in the *Oracle Communications Service Broker Signaling Server Units Configuration Guide* for more information.

Configuring the Announcement Player

To configure the Announcement Player:

1. In the navigation tree, expand **OCSB**.
2. Expand **Processing Tier**.
3. Expand **Applications**.
4. Select **Announcement Player**.
5. In the **mrf-alias** field, enter the alias of the MRF that the Announcement Player should trigger to play the announcement.

   This alias must correspond to the alias that you defined when configuring SIP network entities in the SIP SSU. See the "Configuring SIP Network Entities" section of the "Configuring SIP Signaling Server Units " chapter in the *Oracle Communications Service Broker Signaling Server Units Configuration Guide* for more information.
6. In the **announcement-code** field, enter the code of the announcement that the MRF should play.
7. Click **Apply**.

14

Redirecting Sessions

This chapter describes how you configure the Redirection application in Oracle Communications Online Mediation Controller.

About the Redirection Application

Redirection is an Online Mediation Controller application that can redirect Diameter Credit Control Answers (CCAs) from online charging systems (OCS) to another Online Mediation Controller application. For example, when the OCS replies with a CCA containing a low credit notification, the Redirection application can send the session to the Announcement Player Application to notify the subscriber about a low balance.

The Redirection application supports redirection of CCA responses from Oracle Communications Billing and Revenue Management (BRM), Oracle Communications Elastic Charging Engine (ECE) and compatible third-party Diameter Ro based charging systems when configured within the respective OCS Interworking Module (IM).

When configured, the OCS IM performs session redirection when it receives a CCA response containing one of the following criteria values from the OCS indicating a low or empty account balance for the subscriber:

- FUI (Final-Unit-Indicator)
- NO_MONEY_NOTIFICATION
- LOW_CREDIT_NOTIFCATION

The Redirection application supports both the immediate and non-immediate redirection of sessions following timeout of a validity duration. You use immediate redirection in cases where a subscriber receives a zero balance CCA. Configure non-immediate redirection for scenarios where the CCA includes usable remaining resources that allow a session to continue for a specified time before redirection.

You can configure redirection at both the IM and subscriber levels.

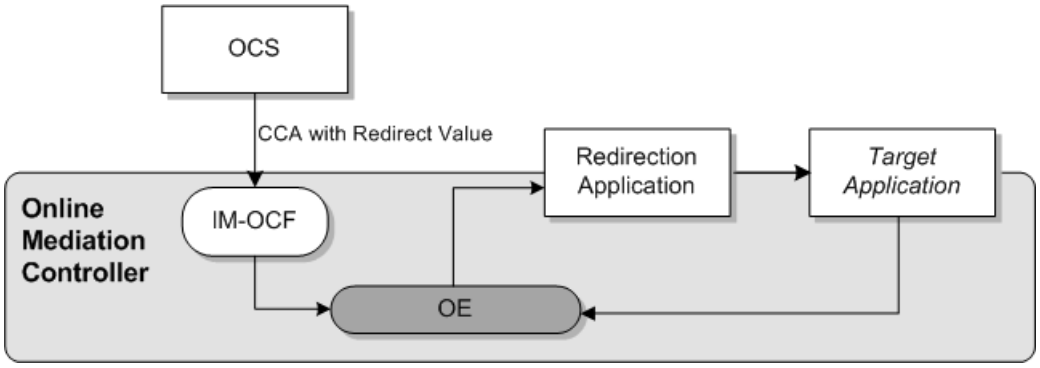
- Redirection at the IM level
- Redirection at the subscriber level

The OCS IM **Redirection** tab contains the configuration for default redirection behavior of all CCAs passing through the IM. See "Configuring the Redirection Application in an IM", for more information about IM level redirection. Configure subscriber-level redirection behavior by creating redirection profile data in the Subscriber Store. The presence of a redirection profile in the Subscriber Store for a subscriber overrides the IM-level redirection configuration. See "Configuring Redirection Using Subscriber Profiles", for information about using the redirection

profile data.

Figure 14–1 shows a sample redirection flow for a CCA containing redirection criteria. IM-OCF identifies the need to redirect the session. The Orchestration Engine (OE) passes the session to the Redirection application which sends the session to a target application. The target application returns the session to the OE for further orchestration.

Figure 14–1 Redirection Application Flow



To allow the OE to trigger the Redirection Application, you need to set up the instance of IM-OCF *OCS* for the OCS used in your implementation with redirection criteria. See "Configuring the Redirection Application in an IM", for more information.

Configuring the Redirection Application in an IM

To set up the Redirection application:

1. Create an instance of IM-OCF *OCS* for the OCS implemented in your environment:
   - Use **IM-OCF PCP** for BRM.
   - Use **IM-OCF ECE** for ECE.
   - Use **IM-OCF** for third-party Diameter Ro OCS.

   See the IM-specific chapters in *Oracle Communications Service Broker Modules Configuration Guide* for more information.
2. In the **Administration Console**, navigate to the IM configuration by clicking **Processing Tier**, then **Interworking Modules**, then selecting the IM-OCF you want to configure.
3. Select the **Redirection** tab.
4. In the **RedirectionCriteria** tab, create the required redirection behaviors for each criteria using the following procedure:
   1. Click **New**.
   2. Configure the redirection behavior attributes listed in Table 14–1:

Table 14–1 Redirection Behavior Attributes

Attribute	Type	Description
Criteria	Enum	Indicates the type of message triggering redirection supplied by the online charging system. Supported values are **FUI**, **NO_MONEY_NOTIFICATION**, and **LOW_CREDIT_ NOTIFICATION**. The default value is **FUI**.
Address	String	The address of the redirection server.
Address Type	Enum	Defines the address type used by the redirection server. Supported values are **IPV4**, **IPV6**, **URL**, and **SIP_URI**. The default value is **IPV4**.
Validity Time	Integer	The allowed time, in seconds, remaining for a redirected subscriber to access network resources before redirection occurs.
Immediately	Boolean	Indicates whether redirection should happen **IMMEDIATELY** or **NON_IMMEDIATELY** (after the value specified in **Validity Time**). When the type is set to NO_ MONEY, this value must be set to IMMEDIATELY. The default value is **IMMEDIATELY**.

Configuring Redirection Using Subscriber Profiles

When configured, the IM Redirection application handles session redirection for subscribers without redirection profile data setup in the Subscriber Store. However, you can configure subscriber-level redirection behaviors by creating redirection profile data in the Subscriber Store.

If you have configured subscriber redirection profile data, the Redirection application gives precedence to it over the configuration at the IM level.

See "Configuring the Subscriber Store", for more information on redirection profile data when using BRM with the Subscriber Store. See *Oracle Communications Service Broker Subscriber Store User's Guide*, for more information on redirection profile data in the local Subscriber Store.

15

Configuring the Home Zones Application

This chapter describes how you configure the Home Zones application in Oracle Communications Online Mediation Controller.

About Network Zoning

You can apply different rates or service access conditions depending on the current location of a mobile subscriber. For example, you can charge a minimum rate when a subscriber is located at home and apply a higher rate when the subscriber is calling from abroad.

Online Mediation Controller recognizes several areas that represent various locations of a subscriber. These areas are called network zones:

- Home zone

  This is an area of a network in which a subscriber is registered. For example, a home zone might be an area closest to a subscriber's residence. You define a subscriber's home zone as a part of the subscriber's profile. See "Defining a Home Zone" for more information.

- Home network

  This is an entire network in which a subscriber is registered. You define a home network using the Home Zones application. See "Defining the Home Network" for more information.

- Roaming

  If a subscriber is not located in either home network or home zone, this subscriber is considered located in a roaming zone.

About Configuration of Network Zones

You define a home zone and home network for each subscriber. The process of defining these zones requires specifying the following criteria:

- Access protocols (such as IEEE-802.11a, 3GPP-GERAN, or 3GPP-UTRAN-FDD). These are the protocols that the subscriber uses to access the network. You can define multiple protocols to handle situations when a subscriber transfers different types of data (for example, voice, short messages, and multimedia) from the same cell.
- Cell identity parameter for each protocol that you defined. A cell identity parameter contains IDs of cells that use the specified access protocol. This parameter varies depending on the access protocol. For example, if you specified

that the access protocol is **IEEE-802.11a**, the call identity parameter can be **cgi-3gpp**. Similarly, if the access protocol is **3GPP-GERAN**, the call identity parameter can be **gsm-location-number**. See *3GPP TS 24.229* for more information.

- IDs of home network cells. You set the cell identity parameter to IDs of cells in which the specified network access protocol is used.

About the Home Zones Application

The information about the network zone in which the subscriber is currently located is transferred to Online Mediation Controller in the **P-Access-Network-Info** header of a message that the subscriber sends.

The Home Zones application is a Online Mediation Controller application that checks this header and identifies whether a mobile subscriber is in the home zone, home network, or roaming.

Depending on the zone, the application adds to the session a custom header, **x-wcs-location**, and sets it accordingly. Other applications can use the value of this header and implement various scenarios. For example, if a subscriber is in the home zone, a charging application can apply a special rate.

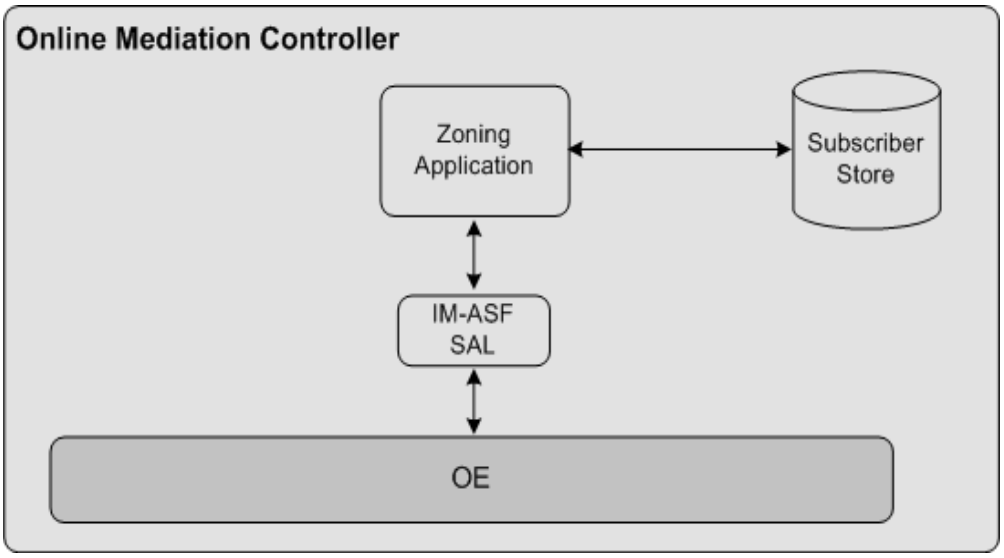
After the Home Zones application receives a message, the application does the following:

1. The Home Zones application retrieves the subscriber's profile from the Subscriber Store to check whether the information set in the **P-Access-Network-Info** header of the message matches the settings that define the home zone of the subscriber.
2. If the **P-Access-Network-Info** header matches these settings, the Home Zones application adds the **x-wcs-location** header to the message and sets this header to "**homezone**".
3. Otherwise, the Home Zones application checks whether the **P-Access-Network-Info** header of the message matches the settings that define the home network.
4. If the **P-Access-Network-Info** header matches these settings, the Home Zones application adds the **x-wcs-location** header to the message and sets this header to "**home network**".
5. Otherwise, the Home Zones application adds the **x-wcs-location** header to the message and sets this header to "**roaming**".

The Home Zones application communicates with the OE through IM-ASF SAL interworking module.

Figure 15–1 shows the place of the Home Zones application in the overall architecture of Online Mediation Controller.

Figure 15–1 Zoning Application



To allow the OE to trigger the Home Zones application, you need to perform the following steps:

1. Define a subscriber's home zone. See "Defining a Home Zone" for more information.
2. Define a subscriber's home network. See "Defining the Home Network" for more information.
3. Set up an instance of IM-ASF SAL and add it to the orchestration flow. See "Setting Up an Instance of IM-ASF SAL" for more information.

Defining a Home Zone

A home zone is an area of the network in which a subscriber is registered. For example, a home zone might be an area closest to a subscriber's residence. You define a home zone in the subscriber's profile under the `<profileDataExtensions>` element as follows:

```
<profileDataExtensions>
   <extensionId>homezone</extensionId>
   <profileDataExtensions>
      <name>access_protocol;cell_identity_parameter</name>
      <value>colon_separated_IDs_of_home_zone_cells</value>
   </profileDataExtensions>
</profileDataExtensions>
```

For example, you might want to specify that a home zone is the area of the network that includes the cells of the following types having the IDs as described in Table 15–1.

Table 15–1 Home Zone Example

Access Protocol	Cell Identity Parameter	Cell IDs
IEEE-802.11a	utran-cell-id-3gpp	1200FF00 1210FF01
3GPP-GERAN	gsm-location-number	1234ABCD 5678EBCD

To reflect this configuration in a subscriber's profile, you need to set up the <profileDataExtensions> element as follows:

```
<profileDataExtensions>
   <extensionId>homenetwork</extensionId>
   <profileDataExtensions>
      <name>IEEE-802.11a;utran-cell-id-3gpp</name>
      <value>1200FF00;1210FF01</value>
   </profileDataExtensions>
   <profileDataExtensions>
      <name>3GPP-GERAN;gsm-location-number</name>
      <value>1234ABCD;5678EBCD</value>
   </profileDataExtensions>
</profileDataExtensions>
```

You create and update subscriber profiles using the Subscriber Provisioning API. See the "Subscriber Provisioning API Reference" chapter in *Service Broker Subscriber Store User's Guide*.

Defining the Home Network

You can define the home network using the Administration Console. The configuration includes specifying access protocols and cell identity parameters for the home network as well as IDs of cells that the home network includes.

Specifying Access Protocols and Cell Identity Parameters

To specify access protocols and cell identity parameters:

1. In the Administration Console, expand **OCSB**.
2. Expand **Processing Tier**.
3. Expand **Applications**.
4. Select **Home Zones**.
5. Click the **Home Network Cell Types** tab.
6. Click **New**.

   The New dialog box appears.
7. In the **Access Type** field, enter the name of the network access protocol used in the cell. For example, you can set **Access Type** to **IEEE-802.11a**.

   See *3GPP TS 24.229* for the list of allowed values.
8. In the **Access Information** field, enter the name of the cell identity parameter according to the network access protocol. For example, you can set Access Information to **utran-cell-id-3gpp**.

   See *3GPP TS 24.229* for the list of allowed values.

   You use this parameter to specify IDs of network cells in which the specified network access protocol is used. See "Specifying IDs of Home Network Cells" for more information.
9. Click **OK**.

Specifying IDs of Home Network Cells

To specify IDs of home network cells:

1. In the Administration Console, expand **OCSB**.
2. Expand **Processing Tier**.
3. Expand **Applications**.
4. Select **Home Zones**.
5. Click the **Home Network Cell Values** tab.
6. In the **Parent** list, select the cell whose ID you want to define.
7. Click **New**.

   The New dialog box appears.
8. In the **Value Data** field, enter an ID of the cell.
9. Click **OK**.

Setting Up an Instance of IM-ASF SAL

To set up an instance of IM-ASF SAL:

1. Create an instance of IM-ASF SAL. See the "Managing Interworking Modules" chapter in *Oracle Communications Service Broker Modules Configuration Guide* for more information.
2. In the **Application Server** tab, set the parameters as follows:
   - In the **SAL Application Address** field, type **sip:homezone@oracle.com**.
   - From the **SAL Mode** list, select **INLINE**.
3. Specify configuration parameters on the **Session Keep Alive** and **SAL** tabs as required. See the "Configuring IM-ASF SAL" chapter in *Oracle Communications Service Broker Modules Configuration Guide* for more information.
4. Specify configuration parameters of the created instance of IM-ASF SAL to make it communicate with the Home Zones application. See the "Configuring IM-ASF SAL" chapter in *Oracle Communications Service Broker Modules Configuration Guide* for more information.
5. Using the Orchestration Studio, do the following:
   a. Add the instance of IM-ASF SAL that communicates with the Home Zones application to the orchestration flow. See the "Specifying IMs" section of the "Building an Orchestration Logic Flow" chapter in *Oracle Communications Service Broker Orchestration User's Guide* for more information.
   b. If necessary, specify conditions that the session must meet to be routed to the instance of IM-ASF SAL. See the "Adding Conditions" section of the "Building an Orchestration Logic Flow" chapter in *Oracle Communications Service Broker Orchestration Studio Guide* for more information.
   c. To trigger other applications based on the value of the **x-wcs-location** header, add conditions for routing the session to different IMs and check whether the **x-wcs-location** is "**homezone**", "**home network**", or "**roaming**". See the "Adding Conditions" section of the "Building an Orchestration Logic Flow" chapter in *Oracle Communications Service Broker Orchestration User's Guide* for more information.

16

Configuring the Threshold Notification Application

This chapter describes how you configure the Threshold Notification application in Oracle Communications Online Mediation Controller.

Note: Online Mediation Controller does not use the Threshold Notification application when the Diameter-based orchestration mode is enabled.

About Threshold Rules

A threshold is a value of a Diameter Attribute-Value Pair (AVP) that triggers Online Mediation Controller to generate an event. Other applications can subscribe to this event and perform various actions when the event occurs.

For example, you can define that if a subscriber reaches a requested time quota, Online Mediation Controller generates an event that triggers an application to send to the subscriber an SMS notification.

You define thresholds for each subscriber by creating threshold rules. A threshold rule is a set of conditions that specify the following:

- An AVP to be checked. Currently, Threshold Notification checks the CC-TIME AVP only
- Threshold value of the CC-TIME AVP
- Whether Online Mediation Controller applies the rule every time the threshold is reached
- Whether Online Mediation Controller should apply the rule only when the subscriber is in roaming

You define threshold rules in a subscriber's profile.

About Threshold Notification

Threshold Notification is a Online Mediation Controller application that compares threshold rules of a subscriber with the units that the subscriber actually used. If Threshold Notification finds that conditions of a threshold rule are met, Threshold Notification generates an event.

Threshold Notification communicates with the OE through IM-ASF SAL interworking module.

Threshold Notification works as follows:

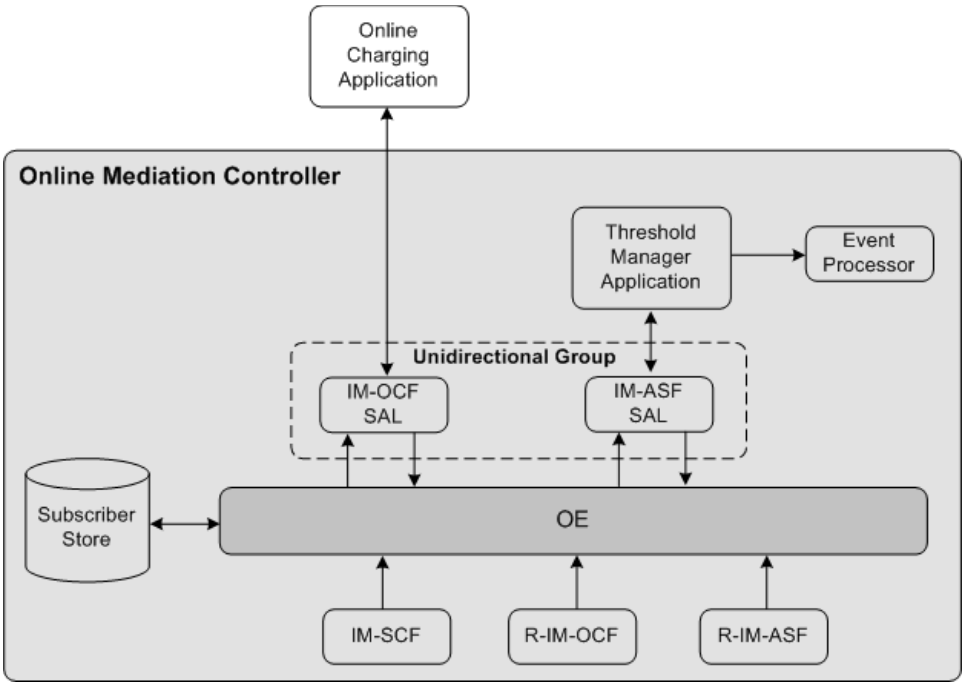
1. Network-facing interworking modules (IMs), such as IM-SCF, R-IM-OCF, or R-IM-ASF, add information about used units and units to be requested to the **ChargingInfo** header of the message.
2. The Orchestration Engine (OE) communicates with the Subscriber Store to retrieve threshold rules from the subscriber profile. The OE adds these rules to the body of the message.
3. The OE forwards the session to IM-OCF. Based on the **ChargingInfo** header, IM-OCF generates a CCR and sends it to the online charging application.
4. The online charging application responds to IM-OCF with a Credit Control Answer (CCA). In this CCA, the online charging application specifies how many units are granted.
5. Based on the CCA, IM-OCF updates the **ChargingInfo** header by adding the amount of granted units.
6. The OE forwards the session to Threshold Notification.

   To allow Threshold Notification to receive updated charging information, the OE must always route a session to Threshold Notification after IM-OCF. To achieve this configuration, you need to form IM-OCF and Threshold Notification into a unidirectional group (see the "Defining the Orchestration Order of Messages Sent by a Called Party" section of the "Invoking Applications Based on the Previous Session Route" chapter in *Oracle Communications Service Broker Orchestration User's Guide* for more information about unidirectional groups).
7. Threshold Notification compares the information stored in the **ChargingInfo** header added by the network-facing modules with the threshold rules added by the OE. When Threshold Notification application finds a match between threshold rules and the information in the **ChargingInfo** header, Threshold Notification generates an event and submits it to Event Processor.

   See "Using the Event Notification API" for more information about how applications can consume events and run their own business logic in the occurrence of these events.

Figure 16–1 shows the place of the Threshold Notification application in the overall architecture of Online Mediation Controller.

Figure 16–1 Threshold Notification Application



To allow the OE to trigger the Threshold Notification application, you need to perform the following steps:

1. Define threshold rules. See "Defining Threshold Rules" for more information.
2. Set up an instance of IM-OCF. See "Setting Up an Instance of IM-OCF".
3. Set up an instance of IM-ASF SAL. See "Setting Up an Instance of IM-ASF SAL" for more information.

About Forcing Network-Facing Modules to Send Charging Requests

In order to timely identify a threshold that the subscriber reaches, Threshold Notification needs to constantly receive the most recent information about used and granted units. Therefore, Threshold Notification needs to check when a network-facing IM sends the next charging request. If the IM schedules to send such a request after the threshold is already reached, Threshold Notification should enforce the IM to send the charging request earlier.

For example, the threshold rule states that a subscriber needs to receive a notification 50 minutes after the call began. If the IM is configured to send an updated charging request every 30 minutes, the online charging application sends CCAs as follows:

- The first CCA is sent in the beginning of the call.
- The second CCA is sent 30 minutes after the call began.
- The third CCA is sent 60 minutes after the call began. In this case, Threshold Notification receives the CCA too late because Threshold Notification should generate an event 50 minutes after the call began.

To receive CCAs in time, Threshold Notification automatically forces network-facing IMs to send an updated charging request earlier than the IM is originally scheduled. In this case, Threshold Notification receives a CCA in time. Threshold Notification does so by adding the custom **x_wcs_threshold** header to the CCA. In the header,

Threshold Notification specifies how much time in advance, in minutes, the IM should send the next request before the threshold is reached. In the example described above, Threshold Notification should add the **x_wcs_threshold** header to the second CCA and set this header to 10 minutes. This enforces the IM to send the IM the next request 50 minutes after the call began.

Defining Threshold Rules

You define threshold rules in the subscriber's profile under the **<profileDataExtensions>** element as follows:

```
<extensions>
   <extension>
      <id>session-threshold</id>
      <field key="threshold_rule_name" value="AVP;Threshold_
Value;IsRecurring;IsRoamingOnly"/>
   </extension>
</extensions>
```

For example, you might want to define two rules that trigger the Threshold Notification application to generate an event when the following occurs:

- Rule 1:
  - The subscriber used 25 minutes
  - Online Mediation Controller applies the rule every time the threshold is reached
  - Whether or not the subscriber is in roaming, Online Mediation Controller always applies this rule
- Rule 2:
  - The subscriber used 10 minutes
  - Online Mediation Controller applies the rule only once
  - Online Mediation Controller applies this rule only when the subscriber is in roaming

To reflect this configuration in a subscriber's profile, you need to set up the **<profileDataExtensions>** element as follows:

```
<extensions>
   <extension>
      <id>session-threshold</id>
      <field key="threshold1" value="CC_TIME;25;true;false"/>
      <field key="threshold2" value="CC_TIME;10;true;true"/>
   </extension>
</extensions>
```

You create and update subscriber profiles using the Subscriber Provisioning API. See the "Subscriber Provisioning API Reference" chapter in *Oracle Communications Service Broker Subscriber Store User's Guide*.

Setting Up an Instance of IM-OCF

To set up an instance of IM-OCF:

1. Create an instance of IM-OCF. See the "Managing Interworking Modules" chapter in *Oracle Communications Service Broker Modules Configuration Guide* for more information.
2. Configure the instance of IM-OCF as required. See the "Configuring IM-OCF" chapter in *Oracle Communications Service Broker Modules Configuration Guide* for more information.
3. Using the Orchestration Studio, do the following:
   a. Add the instance of IM-OCF to the orchestration flow. See the "Specifying IMs" section of the "Building an Orchestration Logic Flow" chapter in *Oracle Communications Service Broker Orchestration Guide* for more information.
   b. If necessary, specify conditions that the session must meet to be routed to the instance of IM-OCF. See the "Adding Conditions" section of the "Building an Orchestration Logic Flow" chapter in *Oracle Communications Service Broker Orchestration Guide* for more information.

Setting Up an Instance of IM-ASF SAL

To set up an instance of IM-ASF SAL:

1. Create an instance of IM-ASF SAL. See the "Managing Interworking Modules" chapter in *Oracle Communications Service Broker Modules Configuration Guide* for more information.
2. In the **Application Server** tab, set the parameters as follows:
   - In the **SAL Application Address** field, type **sip:threshold@oracle.com**.
   - From the **SAL Mode** list, select **INLINE**.
3. Specify configuration parameters on the **Session Keep Alive** and **SAL** tabs as required. See the "Configuring IM-ASF SAL" chapter in *Oracle Communications Service Broker Modules Configuration Guide* for more information.
4. Using the Orchestration Studio, do the following:
   a. Add the instance of IM-ASF SAL that communicates with the Threshold application to the orchestration flow. See the "Specifying IMs" section of the "Building an Orchestration Logic Flow" chapter in *Oracle Communications Service Broker Orchestration User's Guide* for more information.
   b. Create a flow in which the OE routes a session first to IM-OCF and then to IM-ASF SAL. See the "Building an Orchestration Logic Flow" chapter in *Oracle Communications Service Broker Orchestration User's Guide* for more information.
   c. Add the instance of IM-OCF and IM-ASF SAL into a unidirectional group. See the "Defining the Orchestration Order of Messages Sent by a Called Party" section of the "Invoking Applications Based on the Previous Session Route" chapter in *Oracle Communications Service Broker Orchestration User's Guide* for more information.
   d. If necessary, specify conditions that the session must meet to be routed to the instance of IM-ASF SAL. See the "Adding Conditions" section of the "Building an Orchestration Logic Flow" chapter in *Oracle Communications Service Broker Orchestration User's Guide* for more information.

17

Implementing Overload Protection

This chapter describes how to implement overload protection for Oracle Communications Online Mediation Controller.

About Overload Protection

In some cases, such as unanticipated traffic peaks or failure of a network hardware or software component, the load on Service Broker modules can increase significantly. This can cause a situation known as system overload in which Online Mediation Controller modules have insufficient resources to handle new sessions. If overload is not handled correctly, the system can malfunction and critical data can be lost.

To handle increased amounts of traffic without damaging operations of the entire system, Service Broker provides an overload protection mechanism. This mechanism operates in Processing Domains where you can define criteria for overload detection.

By default, too many number of 1) active sessions or 2) initial requests triggers an overload condition. The system rejects initial requests while the overload lasts. In addition to rejecting initial requests, Online Mediation Controller provides the capability to customize how the system behaves if overload occurs.

Using Gauges and Counters as Key Overload Indicators

When you configure gauges and counters as key overload indicators, Online Mediation Controller triggers overload protection if threshold values are crossed as measured by those indicators. You can select any of the counters and gauges provided by Online Mediation Controller to serve as key overload indicators. See "Monitoring Online Mediation Controller" for information about Online Mediation Controller counters.

> **Note:** Consult with Oracle Technical Support if you have questions about which runtime MBeans are best suited to implement overload protection in your network environment.

When you configure overload protection, your settings are applied uniformly across all managed servers in the domain. Usually, server load balancing allocates traffic "fairly" across servers. However, it is possible for a single managed server in a domain to enter an overload condition while the other servers are functioning normally.

Understanding System and Module Levels of Overload Protection

You configure Online Mediation Controller overload protection at the system and the module level:

- System counters and gauges: These two indicators can detect and trigger an overload condition that might occur across any number of clustered managed servers or when deploying any combination of Service Broker products.
  - **`SystemCountersRuntimeMBean.SessionGauge`**
  - **`SystemCountersRuntimeMBean.InitialRequestCount`**

  The SessionGauge gauge represents the total number of active sessions on a single managed server (JVM). Sessions are application specific, for example there are separate Online Mediation Controller, Service Controller, and Policy Controller sessions.

  The InitialRequestCount counter represents the session creation rate. For example, the number of new sessions created per second on a single managed server.

  You configure the SessionGauge and InitialRequestCount indicators for the managed servers in the Administration Console. All managed servers share those configuration settings.

  There is no global overload protection status. However, if any managed server goes into an overload state, as defined by the shared configuration, the system stops accepting new sessions until the overload condition ceases.
- Module-level counters and gauges: These indicators are for specific modules. Using the Administration Console, you can configure these indicators for overload protection under the monitoring tabs here: Expand **Platform**, then **OCSB**, then **Processing Tier**, and then **Interworking and Supplementary Modules**.

  See "Counters and Gauges For Online Mediation Controller", for descriptions of the Online Mediation Controller counters and gauges.

Counters and Gauges For Online Mediation Controller

"Monitoring Online Mediation Controller" describes a broad range of counters and gauges that can be used to monitor Online Mediation Controller. You can use any of the counters or gauges to implement overload protection.

Table 17–1 lists several counters that are particularly useful for implementing overload protection for the following SOAP Web services:

- Subscriber Provisioning API including StoreSubscriber, UpdateSubscriber, GetSubscriber, and DeleteSubscriber.
- TopUp services including including Authenticate, TopUp, GetBalance,and VoucherTopup.

To configure counters for Subscriber Provisioning and TopUp use a JMX-client such as JConsole to access these MBeans:

- Subscriber Provisioning API MBeans are in this bundle:
  - **`oracle.ocsb.app.rcc.service.subscriber_store.provisioning.soap_ws`**
- TopUp MBeans are in this bundle:
  - **`oracle.ocsb.app.rcc.feature.topup.ws.ws_app`**

> **Note:** In a multi-domain topology, the counters for Subscriber Provisioning are not valid. A workaround is to deploy a unified domain.

Table 17–1 describes the counters and gauges for Subscriber Provisioning overload protection and Table 17–2 describes the counters and gauges for TopUp services overload protection.

There are no A) threshold crossed and B) threshold ceased default values for Subscriber Provisioning of for TopUp services counters.

The most important counters for implementing overload protection are:

- **`TotalSubscriberCount`** for Subscriber provisioning services
- **`TopupTotalRequestCount`** for TopUp services

Table 17–1 Counters and Gauges for Subscriber Provisioning Services

Attribute	Description
StoreSubscriberCount	Number of StoreSubscriber requests during the counter interval.
UpdateSubscriberCount	Number of UpdateSubscriber requests during the counter interval.
GetSubscriberCount	Number of GetSubscriber requests during the counter interval.
DeleteSubscriberCount	Number of DeleteSubscriber requests during the counter interval.
TotalSubscriberCount	The total number of the above four counters for Subscriber provisioning related requests during the counter interval.

Table 17–2 Counters and Gauges for TopUp Services

Attribute	Description
AuthenticateCount	Number of Authenticate requests during the counter interval.
TopupCount	Number of Topup requests during the counter interval.
GetBalanceCount	Number of GetBalance request.s during the counter interval.
VoucherTopupCount	Number of VoucherTopup requests during the counter interval.
TopupTotalRequestCount	The total number of the above four counters for Topup related requests during the counter interval.

Understanding the Essential Steps for Configuring Overload Protection

These steps must be followed to configure overload protection.

1. By default, the following gauge and counter are defined as your key overload indicators:
   - **`SystemCountersRuntimeMBean.SessionGauge`**: A gauge that measures the number of active sessions handled by a single managed server. The number of active sessions includes Online Mediation Controller sessions, and if installed, sessions from other installed products: Service Controller, and Policy Controller.
   - **`SystemCountersRuntimeMBean.InitialRequestCount`**: A counter that measures the rate at which a managed server receives new sessions.

     The SessionGauge and InitialRequestCount measurements are per managed server but all of the managed servers share a common configuration. If any managed server goes into an overload condition, the system stops accepting new sessions until the overload condition ceases.

2. Identify Module level counters and gauges you want to use as key overload indicators.
3. Configure Threshold Crossed Notifications details for your module-level counters and gauges.

   For each counter and gauge you want to use as a key overload indicator, define an upper threshold and ceased value threshold. For example, if the upper threshold value is 100 and the ceased value is 90 then if 100 is crossed the system remains overloaded until the value goes below 90.

   Specify a threshold name for each module-level counter or gauge. If you use either sessionGauge or initialRequestCount as the threshold name value you do not have to add these indicators to the Key Overload Indicators pane.
4. Configure Key Overload Indicators.

   After you have configured the module-level counters and gauges you want to use for overload protection you need to specify that they are to be used by Online Mediation Controller as Key Overload Indicators.

   You do this by using the Administration Console. Expand Tier Management, and then use the Key Overload pane to define your key overload indicators.

   Important: Service Broker activates overload protection when any of your key overload indicator crosses its upper threshold.
5. Customize overload protection behavior.

   The behavior of Online Mediation Controller is that if an overload condition occurs, the system continues to handle all active sessions but rejects initial requests until the overload condition ceases.

   In addition to the default protection behavior, you can customize how Service Broker responds to SIP and Diameter network entities that attempt to establish sessions during a system overload.

   Example: You can define the type of error and value of the SIP Retry-After header field that Service Broker uses to respond to newly established SIP sessions.

   You can customize overload protection behavior by using the Administration Console. Expand Tier Management, then Overload and Tracing, and then the Overload Protection Methods pane.

Configuring Key Overload Indicators

The following sections describe in detail how to configure Key Overload Indicators.

Configuring Threshold Crossed Notifications Rules

This section describes how to create Threshold Crossed Notifications rules for overload protection. The components of these rules specify MBean type, threshold name, crossed and ceased threshold values, and other fields.

The following steps are applicable to both systemwide and module-level counters and gauges. There are only two systemwide overload indicators: sessionGauge and InitialRequestCount. The default settings for these indicators should usually not be changed.

To transform any of the counters or gauges you configure in this section to be a key overload indicator you must match the threshold name value you set under the Monitoring tab with the threshold name value in the Key Overload indicators pane.

For example, the default key overload indicator threshold name **`sessionGauge`** matches the threshold name value in the default threshold crossed notifications rule also sessionGauge.

To configure Threshold Crossed Notification Rules do the following:

1. In the navigation tree, expand the **OCSB** node.
2. Expand **Processing Tier**.
3. Do any of the following:
   - To configure System-level Counters and Gauges: Expand Tier Management, then Monitoring, and then Monitoring. **Note**: You cannot add more counters and gauges in addition to the default sessionGauge and InitialRequestCount indicators. However, if required you can modify details such as crossed and ceased threshold values.
   - To configure Module-level Counters and Gauges for OMC:

     Expand Interworking or Supplementary Modules, and then expand the module for which you want to configure a counter or gauge as an overload indicator.
4. In the Monitoring tab, select **Threshold Crossed Notifications**.
5. Be sure you have selected **Lock & Edit** and then click **New**.
6. In the **Threshold Name** field, enter a string that names the threshold. This value is referenced by the key overload indicators.
7. For the **Enable threshold** field, select **True** or **False**. Only enabled thresholds are considered for overload protection.
8. In the **MBean Type** field, enter the type of MBean.
9. For the **Counting Type** field, select the Counting method. For gauges use CurrentGeneralValue and for counters use CurrentIntervalDeltaValue.
10. In the **MBean Attribute** field, enter an MBean attribute. For SystemGaugeRuntime use SessionGauge and for SystemCountRuntime use InitialRequestCount.
11. In the **Threshold class** field, enter **High**. Crossing a low threshold does not cause an overload state.
12. In the **Threshold Value** field, enter an integer value which when crossed triggers an overload state.
13. In the **Threshold ceased value** field, enter an integer value which when crossed the triggered threshold ceases. This value is applicable only to gauges.
14. In the **Threshold crossed message** field, enter a message included in the threshold notification.
15. In the **Threshold ceased message** field, enter a message included in the threshold ceased notification.
16. In the **Server filter** field, leave it empty or use a regular expression to filter on a managed server. For example "managed_1" or "server."
17. In the **Resource filter** field, enter a unique name for the indicator.
18. Click **Apply**.

Specifying Your Key Overload Indicators

Identify the module-level counters and gauges you want to use for overload protection. Use the Administration Console to list the names and threshold names for these indicators.

The Overload Protection pane is pre-populated with these two systemwide indicators:

- **`SystemCountersRuntimeMBean.SessionGauge`**
- **`SystemCountersRuntimeMBean.InitialRequestCount`**

To specify module-level Key Overload Indicators do the following:

1. In the navigation tree, expand the **OCSB** node.
2. Expand **Processing Tier**.
3. Expand **Tier Management**.
4. Select **Overload Protection**. The **Key Overload Indicators** pane appears.
5. Be sure you have selected **Lock & Edit** and then click **New**.
6. In the **Name** field, enter a unique name for the indicator.
7. In the **Threshold Name** field, enter a unique string that references the threshold.

   Multiple indicators at the system or module-level can use the same Threshold Name. In this situation, all matching crossed thresholds will be considered to indicate an overload state.

   Example: If any module-level counter or gauge use either sessionGauge or initialRequestCount as the threshold name value, you do not have to add these module-level indicators to the Key Overload Indicators pane. However, the module-level settings (e.g. crossed threshold value) will override the platform level settings for that individual module only.
8. Click **Apply**.

Configuring General Monitoring Parameters

This section describes how to configure general attributes for overload protection notifications.

Configure these attributes at both the system and module levels. Expand Tier Management, then Monitoring, then select the General tab. To configure IMs expand Interworking Modules, then the IM you want to configure, then Monitoring, and then select the General tab.

At the Tier level these parameters affect only SessionGauge and InitialRequestCount. At the module level the parameters affect all runtime MBeans in the module. Table 17–3 describes the configuration parameters in the Monitoring General tab.

Table 17–3 General Overload Configuration Parameters

Name	Description
Enable runtime MBeans	Disables runtime MBeans so you can neither poll them for values or get notifications.
Enable Notifications	Disables only notifications, so you can still poll values from the MBean.

Table 17–3 (Cont.) General Overload Configuration Parameters

Name	Description
Counter Interval (sec)	This parameter specifies the length of the interval in seconds. Note: This parameter is not configurable at the module level.
Notification trigger interval (sec)	Sampling interval in seconds for checking notifications. For example, if the Counter Interval is set to 10 seconds and the Notification trigger interval is set to 2 seconds for each counter interval the system will determine 5 times whether the threshold value has been crossed.

Configuring the Overload Protection Mechanism

When system overload occurs, Service Broker rejects new sessions and sends response messages to the network entities that attempted to establish the new sessions.

In the Overload Protection Methods pane you can configure how Service Broker responds to attempts by SIP and Diameter network entities to establish new sessions.

Table 17–4 describes configuration parameters you can access by doing the following: Expand Platform, then Processing Tier, then Overload and Tracing, and then the Overload Protection Methods subtab.

Table 17–4 Overload Protection Methods

Name	Type	Description
Enabled	BOOL	Specifies whether the overload protection methods specified in this table are enabled. Possible values: ■ TRUE ■ FALSE
SIP Response Status Code	STRING	Specifies a SIP error that Service Broker returns to a SIP network entity when Service Broker declines an attempt to establish a session. Default value: 503
SIP Retry-After	STRING	Specifies the value that Service Broker sets in the Retry-After header of the error response sent to the network entity. This value defines how long the network entity waits before it retries to establish a session. Default value: 300
Diameter Response Result Code	STRING	Specifies a response Result Code AVP that Service Broker returns to a Diameter network entity when Service Broker declines the attempt to establish a session. Default value: 5012
Web Service Response Status Code	INT	Specifies an error code that Online Mediation Controller returns to a Web service network entity when Service Broker declines the attempt to establish a session. Default value: 503

Table 17–4 (Cont.) Overload Protection Methods

Name	Type	Description
SAL Response Status Code	INT	Specifies an error code that Online Mediation Controller returns to a SAL application when Service Broker declines the attempt to establish a session. Default value: 503

A

Diameter Ro to BRM Opcode Mapping

This appendix contains reference information detailing the Diameter Ro to Oracle Communications Billing and Revenue Management (BRM) opcode mapping. When mediating charging related messages (CCRs and CCAs) between a Diameter node and Billing and Revenue Management, Oracle Communications Online Mediation Controller maps the referenced Diameter operations and their attribute-value pairs (AVPs) to the listed Billing and Revenue Management opcodes and Flists. PCP to Diameter Ro results codes are also provided.

CCR Operation to Billing and Revenue Management Opcode Mapping

Table A–1 contains the supported CCR operation and their respective Billing and Revenue Management opcode mapping. Each operation also includes the incoming AVP criteria used to map the opcode.

Table A–1 CCR Operations to BRM Opcode Mapping

CCR Operation	BRM Opcode	Incoming AVP Criteria
Balance Check	PCM_OP_TCF_AAA_QUERY_BALANCE	CC-Request-Type=**EVENT_REQUEST** Requested-Action=**CHECK_BALANCE**
Service Price Enquiry	PCM_OP_TCF_AAA_SERVICE_PRICE_ENQUIRY	CC-Request-Type=**EVENT_REQUEST** Requested-Action=**PRICE_ENQUIRY**
Direct Debit	PCM_OP_TCF_AAA_STOP_ACCOUNTING	CC-Request-Type=**EVENT_REQUEST** Requested-Action=**DIRECT_DEBIT**
Refund Amount	PCM_OP_TCF_AAA_REFUND	CC-Request-Type=**EVENT_REQUEST** Requested-Action=**REFUND_ACCOUNT**
Session Initiate	PCM_OP_TCF_AAA_AUTHORIZE	CC-Request-Type=**INITIAL**
Session Update	PCM_OP_TCF_AAA_UPDATE_AND_REAUTHORIZE	CC-Request-Type=**UPDATE**
Session Terminate	PCM_OP_TCF_AAA_STOP_ACCOUNTING	CC-Request-Type=**TERMINATION**
Session Terminate Request (STR)	PCM_OP_TCF_AAA_CANCEL_AUTHORIZATION	N/A

CCR Session Initial Request AVP to Opcode Flist Mapping

Table A–2 contains the CCR session initial request AVP to opcode input Flist mapping.

Table A–2 CCR Session Initial Request AVP to Opcode Flist Mapping

CCR AVP	Input Flist Field	Notes
Session-Id	PIN_FLD_AUTHORIZATION_ID	N/A
Service-Identifier	PIN_FLD_POID	Used for service type POID only
Subscription-Id-Data	PIN_FLD_MSID	N/A
Subscription-Id-Data	TELCO_INFO.PIN_FLD_IMSI	Used if the Subscription-Id-Type is **END_USER_IMSI**
Subscription-Id-Data	TELCO_INFO.PIN_FLD_MSISDN	Used if Subscription-Id-Type is **END_USER_E164**
Origin-Realm	PIN_FLD_ORIGIN_NETWORK	N/A
Destination-Realm	PIN_FLD_DESTINATION_NETWORK	N/A
N/A	PIN_FLD_OBJTYPE	Always mapped to **gsm**
Requested-Service-Unit.CC-Time	PIN_FLD_QUANTITY	Used for duration-based rating
Requested-Service-Unit.CC-Money.Currency-Code	PIN_FLD_UNIT	N/A
Requested-Service-Unit.CC-Money.Unit-Value.Value-Digits/Exponent	PIN_FLD_AMOUNT	N/A
Requested-Service-Unit.CC-Input-Octets	PIN_FLD_REQ_BYTES_DOWNLINK	Used for volume-based rating
Requested-Service-Unit.CC-Output-Octets	PIN_FLD_REQ_BYTES_UPLINK	Used for volume-based rating
Requested-Service-Unit.CC-Total-Octets	PIN_FLD_REQ_BYTES_DOWNLINK	Used for volume-based rating
Event-Timestamp	PIN_FLD_START_T	N/A
N/A	PIN_FLD_EXPIRATION_T	Used for the **ConfiguredreservationExpirationTime** value
Service-Information, MMTel-Information, Subscriber-Role	EXTENDED_INFO.GSM_INFO.PIN_FLD_DIRECTION	N/A
Service-Information, PS-Information, 3GPP-SGSN-MCC-MNC	EXTENDED_INFO.GSM_INFO.PIN_FLD_ORIGIN_SID or EXTENDED_INFO.GSM_INFO.PIN_FLD_DESTINATION_SID	Flist field depends on direction of communication
Service-Information, PS-Information, 3GPP-User-Location-Info	EXTENDED_INFO.GSM_INFO.PIN_FLD_LOC_AREA and PIN_FLD_CELL_ID	N/A
Service-Information, PS-Information, 3GPP2-Bsid	EXTENDED_INFO.GSM_INFO.PIN_FLD_CELL_ID	N/A
Calling-Station-Id	PIN_FLD_CALLING_NUMBER	N/A
Service-Information, IMS-Information, Calling-Party-Address	PIN_FLD_CALLING_NUMBER	N/A
Called-Station -Id	PIN_FLD_CALLED_NUMBER	N/A
Service-Information, IMS-Information, Called-Party-Address	PIN_FLD_CALLED_NUMBER	N/A

CCR Session Update Request AVP to Opcode Flist Mapping

Table A–3 contains the CCR session update request AVP to opcode input Flist mapping.

Table A–3 CCR Session Update Request AVP to Opcode Flist Mapping

CCR AVP	Input Flist Field	Notes
Session-Id	PIN_FLD_AUTHORIZATION_ID	N/A
Service-Identifier	PIN_FLD_POID	Used for service type POID only
Subscription-Id-Data	PIN_FLD_MSID	N/A
Subscription-Id-Data	TELCO_INFO.PIN_FLD_IMSI	Used if the Subscription-Id-Type is **END_USER_IMSI**
Subscription-Id-Data	TELCO_INFO.PIN_FLD_MSISDN	Used if Subscription-Id-Type is **END_USER_E164**
Origin-Realm	PIN_FLD_ORIGIN_NETWORK	N/A
Destination-Realm	PIN_FLD_DESTINATION_NETWORK	N/A
N/A	PIN_FLD_OBJTYPE	Always mapped to **gsm**
Used-Service-Unit.CC-Time	PIN_FLD_QUANTITY	Used for duration-based rating
Event-Timestamp	PIN_FLD_END_T	Used when Used-Service-Unit.CC-Time is not present
Used-Service-Unit.CC-Money.Currency-Code	PIN_FLD_UNIT	N/A
Used-Service-Unit.CC-Money.Unit-Value.Value-Digits/Exponent	PIN_FLD_AMOUNT	N/A
Used-Service-Unit.CC-Input-Octets	PIN_FLD_BYTES_DOWNLINK	Used for volume-based rating
Used-Service-Unit.CC-Output-Octets	PIN_FLD_BYTES_UPLINK	Used for volume-based rating
Used-Service-Unit.CC-Total-Octets	PIN_FLD_BYTES_DOWNLINK	Used for volume-based rating
Requested-Service-Unit.CC-Time	PIN_FLD_REQ_QUANTITY	Used for duration-based rating
Requested-Service-Unit.CC-Money.Currency-Code	PIN_FLD_UNIT	N/A
Requested-Service-Unit.CC-Money.Unit-Value.Value-Digits/Exponent	PIN_FLD_REQ_AMOUNT	N/A
Requested-Service-Unit.CC-Input-Octets	PIN_FLD_REQ_BYTES_DOWNLINK	Used for volume-based rating
Requested-Service-Unit.CC-Output-Octets	PIN_FLD_REQ_BYTES_UPLINK	Used for volume-based rating
Requested-Service-Unit.CC-Total-Octets	PIN_FLD_REQ_BYTES_DOWNLINK	Used for volume-based rating

Table A–3 (Cont.) CCR Session Update Request AVP to Opcode Flist Mapping

CCR AVP	Input Flist Field	Notes
N/A	PIN_FLD_REQ_MODE	Possible values: ■ Duration: **2** ■ Duration and Volume: **6** ■ Volume: **4** ■ Amount: **1**
N/A	PIN_FLD_EXPIRATION_T	Used for the **ConfiguredreservationExpiration Time** value
Service-Information, MMTel-Information, Subscriber-Role	EXTENDED_INFO.GSM_INFO.PIN_FLD_DIRECTION	N/A
Service-Information, PS-Information, 3GPP-SGSN-MCC-MNC	EXTENDED_INFO.GSM_INFO.PIN_FLD_ORIGIN_SID or EXTENDED_INFO.GSM_INFO.PIN_FLD_DESTINATION_SID	Flist field depends on direction of communication
Service-Information, PS-Information, 3GPP-User-Location-Info	EXTENDED_INFO.GSM_INFO.PIN_FLD_LOC_AREA and PIN_FLD_CELL_ID	N/A
Service-Information, PS-Information, 3GPP2-Bsid	EXTENDED_INFO.GSM_INFO.PIN_FLD_CELL_ID	N/A
Calling-Station-Id	PIN_FLD_CALLING_NUMBER	N/A
Service-Information, IMS-Information, Calling-Party-Address	PIN_FLD_CALLING_NUMBER	N/A
Called-Station-Id	PIN_FLD_CALLED_NUMBER	N/A
Service-Information, IMS-Information, Called-Party-Address	PIN_FLD_CALLED_NUMBER	N/A

CCR Session Termination Request AVP to Opcode Flist Mapping

Table A–4 contains the CCR session termination request AVP to opcode input Flist mapping.

Table A–4 CCR Session Termination Request AVP to Opcode Flist Mapping

CCR AVP	Input Flist Field	Notes
Session-Id	PIN_FLD_AUTHORIZATION_ID	N/A
Service-Identifier	PIN_FLD_POID	Used for service type POID only
Subscription-Id-Data	PIN_FLD_MSID	N/A
Subscription-Id-Data	TELCO_INFO.PIN_FLD_IMSI	Used if the Subscription-Id-Type is **END_USER_IMSI**
Subscription-Id-Data	TELCO_INFO.PIN_FLD_MSISDN	Used if Subscription-Id-Type is **END_USER_E164**
Origin-Realm	PIN_FLD_ORIGIN_NETWORK	N/A
Destination-Realm	PIN_FLD_DESTINATION_NETWORK	N/A
N/A	PIN_FLD_OBJTYPE	Always mapped to **gsm**

Table A–4 (Cont.) CCR Session Termination Request AVP to Opcode Flist Mapping

CCR AVP	Input Flist Field	Notes
Event-Timestamp	PIN_FLD_END_T	N/A
Used-Service-Unit.CC-Time	PIN_FLD_QUANTITY	Used for duration-based rating
Used-Service-Unit.CC-Money.Currency-Code	PIN_FLD_UNIT	N/A
Used-Service-Unit.CC-Money.Unit-Value.Value-Digits/Exponent	PIN_FLD_AMOUNT	N/A
Used-Service-Unit.CC-Input-Octets	PIN_FLD_BYTES_DOWNLINK	Used for volume-based rating
Used-Service-Unit.CC-Output-Octets	PIN_FLD_BYTES_UPLINK	Used for volume-based rating
Used-Service-Unit.CC-Total-Octets	PIN_FLD_BYTES_DOWNLINK	Used for volume-based rating
N/A	PIN_FLD_EXPIRATION_T	Used for the **ConfiguredreservationExpirationTime** value
Service-Information, MMTel-Information, Subscriber-Role	EXTENDED_INFO.GSM_INFO.PIN_FLD_DIRECTION	N/A
Service-Information, PS-Information, 3GPP-SGSN-MCC-MNC	EXTENDED_INFO.GSM_INFO.PIN_FLD_ORIGIN_SID or EXTENDED_INFO.GSM_INFO.PIN_FLD_DESTINATION_SID	Flist field depends on direction of communication
Service-Information, PS-Information, 3GPP-User-Location-Info	EXTENDED_INFO.GSM_INFO.PIN_FLD_LOC_AREA and PIN_FLD_CELL_ID	N/A
Service-Information, PS-Information, 3GPP2-Bsid	EXTENDED_INFO.GSM_INFO.PIN_FLD_CELL_ID	N/A
Calling-Station-Id	PIN_FLD_CALLING_NUMBER	N/A
Service-Information, IMS-Information, Calling-Party-Address	PIN_FLD_CALLING_NUMBER	N/A
Called-Station-Id	PIN_FLD_CALLED_NUMBER	N/A
Service-Information, IMS-Information, Called-Party-Address	PIN_FLD_CALLED_NUMBER	N/A

CCR Event DirectDebit Request AVP to Opcode Flist Mapping

Table A–5 contains the CCR event DirectDebit request AVP to opcode input Flist mapping.

Table A–5 CCR Event DirectDebit Request AVP to Opcode Flist Mapping

CCR AVP	Input Flist Field	Notes
Session-Id	PIN_FLD_AUTHORIZATION_ID	N/A
Service-Identifier	PIN_FLD_POID	Used for service type POID only
Subscription-Id-Data	PIN_FLD_MSID	N/A

Table A–5 (Cont.) CCR Event DirectDebit Request AVP to Opcode Flist Mapping

CCR AVP	Input Flist Field	Notes
Subscription-Id-Data	TELCO_INFO.PIN_FLD_IMSI	Used if the Subscription-Id-Type is **END_USER_IMSI**
Subscription-Id-Data	TELCO_INFO.PIN_FLD_MSISDN	Used if Subscription-Id-Type is **END_USER_E164**
Origin-Realm	PIN_FLD_ORIGIN_NETWORK	N/A
Destination-Realm	PIN_FLD_DESTINATION_NETWORK	N/A
N/A	PIN_FLD_OBJTYPE	Always mapped to **gsm**
Event-Timestamp	PIN_FLD_END_T	N/A
Requested-Service-Unit.CC-Time	PIN_FLD_QUANTITY	Used for duration-based rating
Requested-Service-Unit.CC-Money.Currency-Code	PIN_FLD_UNIT	N/A
Requested-Service-Unit.CC-Money.Unit-Value.Value-Digits/Exponent	PIN_FLD_AMOUNT	N/A
Requested-Service-Unit.CC-Input-Octets	PIN_FLD_REQ_BYTES_DOWNLINK	Used for volume-based rating
Requested-Service-Unit.CC-Output-Octets	PIN_FLD_REQ_BYTES_UPLINK	Used for volume-based rating
Requested-Service-Unit.CC-Total-Octets	PIN_FLD_REQ_BYTES_DOWNLINK	Used for volume-based rating
Requested-Service-Unit.CC-Service-Specific-Units	PIN_FLD_NUMBER_OF_UNITS	Used for occurrence-based rating
N/A	PIN_FLD_EXPIRATION_T	Used for the **ConfiguredreservationExpirationTime** value
N/A	EXTENDED_INFO.GSM_INFO.PIN_FLD_DIRECTION	Set as **1** for GSM
Termination-Cause	PIN_FLD_TERMINATE	N/A
N/A	PIN_FLD_AGGREGATE_MODE	Set as **1** to aggregate

CCR Event RefundAccount Request AVP to Opcode Flist Mapping

Table A–6 contains the CCR event RefundAccount request AVP to opcode input Flist mapping.

Table A–6 CCR Event RefundAccount Request AVP to Opcode Flist Mapping

CCR AVP	Input Flist Field	Notes
Session-Id	PIN_FLD_AUTHORIZATION_ID	N/A
Service-Identifier	PIN_FLD_POID	Used for service type POID only
Subscription-Id-Data	PIN_FLD_MSID	N/A
Subscription-Id-Data	TELCO_INFO.PIN_FLD_IMSI	Used if the Subscription-Id-Type is **END_USER_IMSI**
Subscription-Id-Data	TELCO_INFO.PIN_FLD_MSISDN	Used if Subscription-Id-Type is **END_USER_E164**
Origin-Realm	PIN_FLD_ORIGIN_NETWORK	N/A

Table A–6 (Cont.) CCR Event RefundAccount Request AVP to Opcode Flist Mapping

CCR AVP	Input Flist Field	Notes
Destination-Realm	PIN_FLD_DESTINATION_NETWORK	N/A
N/A	PIN_FLD_OBJTYPE	Always mapped to **gsm**
Requested-Service-Unit.CC-Time	PIN_FLD_QUANTITY	Used for duration-based rating
Requested-Service-Unit.CC-Money.Currency-Code	PIN_FLD_UNIT	N/A
Requested-Service-Unit.CC-Money.Unit-Value.Value-Digits/Exponent	PIN_FLD_AMOUNT	N/A
Requested-Service-Unit.CC-Input-Octets	PIN_FLD_REQ_BYTES_DOWNLINK	Used for volume-based rating
Requested-Service-Unit.CC-Output-Octets	PIN_FLD_REQ_BYTES_UPLINK	Used for volume-based rating
Requested-Service-Unit.CC-Total-Octets	PIN_FLD_REQ_BYTES_DOWNLINK	Used for volume-based rating
Requested-Service-Unit.CC-Service-Specific-Units	PIN_FLD_NUMBER_OF_UNITS	Used for occurrence-based rating
N/A	EXTENDED_INFO.GSM_INFO.PIN_FLD_DIRECTION	Set as **1** for GSM

CCR Event CheckBalance Request AVP to Opcode Flist Mapping

Table A–7 contains the CCR event CheckBalance request AVP to opcode input Flist mapping.

Table A–7 CCR Event CheckBalance Request AVP to Opcode Flist Mapping

CCR AVP	Input Flist Field	Notes
Session-Id	PIN_FLD_AUTHORIZATION_ID	N/A
Service-Identifier	PIN_FLD_POID	Used for service type POID only
Subscription-Id-Data	PIN_FLD_MSID	N/A
Subscription-Id-Data	TELCO_INFO.PIN_FLD_IMSI	Used if the Subscription-Id-Type is **END_USER_IMSI**
Subscription-Id-Data	TELCO_INFO.PIN_FLD_MSISDN	Used if Subscription-Id-Type is **END_USER_E164**

CCR Event PriceEnquiry Request AVP to Opcode Flist Mapping

Table A–8 contains the CCR event PriceEnquiry request AVP to opcode input Flist mapping.

Table A–8 CCR Event PriceEnquiry Request AVP to Opcode Flist Mapping

CCR AVP	Input Flist Field	Notes
Session-Id	PIN_FLD_AUTHORIZATION_ID	N/A
Service-Identifier	PIN_FLD_POID	Used for service type POID only
Subscription-Id-Data	PIN_FLD_MSID	N/A

Table A–8 (Cont.) CCR Event PriceEnquiry Request AVP to Opcode Flist Mapping

CCR AVP	Input Flist Field	Notes
Subscription-Id-Data	TELCO_INFO.PIN_FLD_IMSI	Used if the Subscription-Id-Type is **END_USER_IMSI**
Subscription-Id-Data	TELCO_INFO.PIN_FLD_MSISDN	Used if Subscription-Id-Type is **END_USER_E164**
Origin-Realm	PIN_FLD_ORIGIN_NETWORK	N/A
Destination-Realm	PIN_FLD_DESTINATION_NETWORK	N/A
Event-Timestamp	PIN_FLD_START_T	N/A
N/A	PIN_FLD_END_T	Current time is used
Requested-Service-Unit.CC-Input-Octets	PIN_FLD_REQ_BYTES_DOWNLINK	Used for volume-based rating
Requested-Service-Unit.CC-Output-Octets	PIN_FLD_REQ_BYTES_UPLINK	Used for volume-based rating
Requested-Service-Unit.CC-Total-Octets	PIN_FLD_REQ_BYTES_DOWNLINK	Used for volume-based rating
Requested-Service-Unit.CC-Service-Specific-Units	PIN_FLD_NUMBER_OF_UNITS	Used for occurrence-based rating
N/A	EXTENDED_INFO.GSM_INFO.PIN_FLD_DIRECTION	Set as **1** for GSM

Opcode Output Flist to CCA Session Response Mapping

Table A–9 contains the BRM output opcode input Flist to CCA session response mapping.

Table A–9 BRM Output Opcode Flist to CCA Session Response Mapping

Output Flist Field	CCA AVP	Notes
PIN_FLD_RATING_STATUS PIN_FLD_RESULT PIN_FLD_REASON	Result-Code	N/A
PIN_FLD_AVAILABLE_RESOURCE_LIMIT	Final-Unit-Indication	If available resource == **0**
PIN_FLD_RUM_MAP	Granted-Service-Unit	N/A
PIN_FLD_EXPIRATION_T	Validity-Time	N/A

Diameter Ro to Billing and Revenue Management Result Codes Mapping

Table A–10 contains the Diameter Ro to Billing and Revenue Management result codes mapping.

Table A–10 Diameter Ro to BRM Result Codes Mapping

Diameter Result Code	Description	BRM PIN_FLD_RESULT, PIN_FLD_REASON, PIN_FLD_RATING_STATUS Values (respectively)	Error Code
3002	DIAMETER_UNABLE_TO_DELIVER	N/A	code=ERR_TIMEOUT (108)
5004	DIAMETER_INVALID_AVP_VALUE	N/A	code=ERR_BAD_ARG (4)
3004	DIAMETER_TOO_BUSY	N/A	code=ERR_PERF_LIMIT_REACHED (118)
5030	DIAMETER_USER_UNKNOWN	N/A	code=ERR_BAD_ARG (4), errField=EBUF_PIN_FLD_MSID (*MSID*)
2001	DIAMETER_SUCCESS	1,-, -	N/A
5003	DIAMETER_AUTHORIZATION_REJECTED	0, NULL, -	N/A
5012	DIAMETER_UNABLE_TO_COMPLY	0, 0, 0 or 0, 4, 0	N/A
5012	DIAMETER_UNABLE_TO_COMPLY	0, 0, 1 or 0, 4, 1	N/A
5012	DIAMETER_UNABLE_TO_COMPLY	0, 0, 13 or 0, 4, 13	N/A
5031	DIAMETER_RATING_FAILED	0, 0, 10 or 0, 4, 10	N/A
5031	DIAMETER_RATING_FAILED	0, 0, 11 or 0, 4, 11	N/A
4010	DIAMETER_END_USER_SERVICE_DENIED	0, 0, 12 or 0, 4, 12	N/A
5031	DIAMETER_RATING_FAILED	0, 0, 14 or 0, 4, 14	N/A
5031	DIAMETER_RATING_FAILED	0, 0, 15 or 0, 4, 15	N/A
5031	DIAMETER_RATING_FAILED	0, 0, 16 or 0, 4, 16	N/A
5031	DIAMETER_RATING_FAILED	0, 0, 18 or 0, 4, 18	N/A
5031	DIAMETER_RATING_FAILED	0, 0, 19 or 0, 4, 19	N/A
5031	DIAMETER_RATING_FAILED	0, 0, 20 or 0, 4, 20	N/A
5031	DIAMETER_RATING_FAILED	0, 0, 21 or 0, 4, 21	N/A
5031	DIAMETER_RATING_FAILED	0, 0, 22 or 0, 4, 22	N/A
5031	DIAMETER_RATING_FAILED	0, 0, 23 or 0, 4, 23	N/A
5031	DIAMETER_RATING_FAILED	0, 0, 24 or 0, 4, 24	N/A
5031	DIAMETER_RATING_FAILED	0, 0, 25 or 0, 4, 25	N/A
4012	DIAMETER_CREDIT_LIMIT_REACHED	0, 0, 26 or 0, 4, 26 or NULL, NULL, 26	N/A
4010	DIAMETER_END_USER_SERVICE_DENIED	0, 0, 17 or 0, 4, 17	N/A
4010	DIAMETER_END_USER_SERVICE_DENIED	0, 1, -	N/A
5012	DIAMETER_UNABLE_TO_COMPLY	0, 2, -	N/A
5012	DIAMETER_UNABLE_TO_COMPLY	0, 3, -	N/A
4012	DIAMETER_CREDIT_LIMIT_REACHED	0, 5, -	N/A
5012	DIAMETER_UNABLE_TO_COMPLY	0, 6, -	N/A