

Oracle® Communications Services Gatekeeper

RESTful Application Development Guide

Release 4.1

January 2009

Copyright © 2007, 2008, 2009, Oracle and/or its affiliates. All rights reserved.

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

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

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

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

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

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

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

Contents

Document Roadmap

Document Scope and Audience	1-1
Guide to This Document	1-1
Terminology	1-2
Related Documentation	1-8

Overview of Oracle Communications Services Gatekeeper

Basic Concepts	2-1
Communication Services	2-2
Traffic Types	2-2
Application-initiated Traffic	2-2
Network-triggered Traffic	2-3
Management Structures	2-3
Functional Overview	2-4
Application Testing Workflow	2-5

Interacting with the RESTful Facade

Format	3-1
URI	3-2
Header	3-2
Service Correlation	3-3
Parameter Tunneling	3-4
Body	3-4

Attachments	3-4
Methods	3-4
HTTP GET	3-5
HTTP POST	3-5
HTTP PUT	3-6
HTTP DELETE	3-6
Authentication and Security	3-7
Notifications and Publish/Subscribe	3-7
Errors and Exceptions	3-8

Session Manager

Operations	4-1
Get Session	4-1
Request	4-2
Response	4-2
Error Response	4-2
Get Session Remaining Lifetime	4-2
Request	4-2
Response	4-2
Error Response	4-3
Destroy a Session	4-3
Request	4-3
Response	4-3
Error Response	4-3
Errors	4-3

Short Messaging

Operations	5-1
----------------------	-----

Send Sms	5-1
Request	5-1
Response	5-3
Error Response	5-3
Send Sms Ringtone.	5-3
Request	5-3
Response	5-4
Error Response	5-5
Send Sms Logo.	5-5
Request	5-5
Response	5-6
Error Response	5-6
Get Received Sms.	5-6
Request	5-6
Response	5-7
Error Response	5-7
Get Sms Delivery Status.	5-7
Request	5-7
Response	5-8
Error Response	5-9
Start Sms Notification	5-9
Request	5-9
Response	5-10
Error Response	5-10
Stop Sms Notification.	5-10
Request	5-10
Response	5-11
Error Response	5-11

Notifications and Receipts	5-11
Notify Sms Reception	5-11
Notify Sms Delivery Receipt	5-11
Errors	5-12

Multimedia Messaging

Operations	6-1
Send Message.	6-1
Request	6-1
Response	6-3
Error Response	6-3
Get Received Messages	6-3
Request	6-3
Response	6-4
Error Response	6-5
Get Message.	6-5
Request	6-5
Response	6-5
Error Response	6-5
Get Message Delivery Status	6-5
Request	6-6
Response	6-6
Error Response	6-7
Start MMS Notification	6-7
Request	6-7
Response	6-8
Error Response	6-8
Stop Message Notification	6-8

Request	6-8
Response	6-9
Error Response	6-9
Notifications and Receipts	6-9
Notify Message Reception	6-9
Notify Message Delivery Receipt	6-10
Errors	6-10

Terminal Location

Operations	7-1
Get Location	7-1
Request	7-1
Response	7-3
Error Response	7-3
Get Location For Group	7-4
Request	7-4
Response	7-5
Error Response	7-5
Get Terminal Distance	7-6
Request	7-6
Response	7-6
Error Response	7-6
Start Geographical Notifications	7-6
Request	7-7
Response	7-8
Error Response	7-8
Start Periodic Notification	7-8
Request	7-8

Response	7-9
Error Response	7-10
End Notification	7-10
Request	7-10
Response	7-10
Error Response	7-10
Notifications	7-10
Location End	7-11
Location Error	7-11
Location Notification	7-11
Errors	7-12

Third Party Call

Operations	8-1
Make Call	8-1
Request	8-1
Response	8-2
Error Response	8-2
Get Call Information	8-3
Request	8-3
Error Response	8-4
Cancel Call Request	8-4
Request	8-4
Response	8-4
Error Response	8-4
End Call	8-4
Request	8-4
Response	8-5

Error Response	8-5
Errors	8-5

Call Notification

Operations	9-1
Start Call Notification.	9-1
Request	9-1
Response	9-2
Error Response	9-3
Stop Call Notification.	9-3
Request	9-3
Response	9-3
Error Response	9-3
Start Call Direction Notification	9-3
Request	9-4
Response	9-5
Error Response	9-5
Stop Call Direction Notification.	9-5
Request	9-5
Response	9-5
Error Response	9-5
Notifications	9-6
Notify Called Number	9-6
Notify Busy	9-6
Notify No Answer	9-6
Notify Not Reachable.	9-6
Handle Called Number.	9-7
Handle Busy	9-8

Handle No Answer	9-9
Handle Not Reachable	9-10
Errors	9-10

Presence

Operations	10-1
Get Open Subscriptions	10-1
Request	10-2
Response	10-2
Error Response	10-3
Get My Watchers	10-3
Request	10-3
Response	10-3
Error Response	10-3
Update Subscription Authorization	10-3
Request	10-4
Response	10-4
Error Response	10-4
Subscribe Presence	10-4
Request	10-5
Response	10-5
Error Response	10-6
Block Subscription	10-6
Request	10-6
Response	10-6
Error Response	10-6
Start Presence Notification	10-6
Request	10-6

Response	10-8
Error Response	10-8
End Presence Notification	10-8
Request	10-8
Response	10-9
Error Response	10-9
Get User Presence.	10-9
Request	10-9
Response	10-10
Error Response	10-11
Publish	10-11
Request	10-11
Response	10-13
Error Response	10-13
Notifications	10-13
Status Changed	10-13
Notify Subscription	10-15
Status End.	10-15
Subscription Ended.	10-15
Errors	10-15

Payment

Operations	11-1
Charge Amount	11-1
Request	11-1
Response	11-2
Error Response	11-2
Refund Amount	11-2

Request	11-2
Response	11-3
Error Response	11-3
Charge Split Amount	11-3
Request	11-4
Response	11-4
Error Response	11-4
Reserve Amount.	11-5
Request	11-5
Response	11-5
Error Response	11-6
Reserve Additional Amount.	11-6
Request	11-6
Response	11-7
Error Response	11-7
Charge Reservation	11-7
Request	11-7
Response	11-8
Error Response	11-8
Release Reservation	11-8
Request	11-8
Response	11-8
Error Response	11-9
Errors	11-9

Subscriber Profile

Operations	12-1
Get	12-1

Request	12-1
Response	12-2
Error Response	12-2
Get Profile	12-2
Request	12-2
Response	12-3
Error Response	12-3
Errors	12-3

WAP Push

Operation	13-1
Send Push Message	13-2
Request	13-2
Response	13-3
Error Response	13-5
Notifications	13-5
Result Notification Message	13-6
Errors	13-6

Document Roadmap

This chapter describes the audience for and the organization of this document: It includes:

- [Document Scope and Audience](#)
- [Guide to This Document](#)
- [Terminology](#)
- [Related Documentation](#)

Document Scope and Audience

This document provides information for those developers who wish to integrate functionality provided by telecom networks into their programs by using the RESTful Web Services offered by Oracle Communications Services Gatekeeper. It includes a high-level overview of the process, including the login and security procedures, and a description of the interfaces and operations that are available for use.

Guide to This Document

The document contains the following chapters:

[Chapter 1, “Document Roadmap”](#): This chapter

[Chapter 2, “Overview of Oracle Communications Services Gatekeeper”](#): A general introduction to the concepts involved in using Oracle Communications Services Gatekeeper

[Chapter 3, “Interacting with the RESTful Facade”](#): An overview of how Oracle Communications Services Gatekeeper uses RESTful interfaces

[Chapter 4, “Session Manager”](#): A detailed description of the RESTful Session Manager Web Service

[Chapter 5, “Short Messaging”](#): A detailed description of the RESTful Short Messaging Web Service

[Chapter 6, “Multimedia Messaging”](#): A detailed description of the RESTful Multimedia Messaging Web Service

[Chapter 7, “Terminal Location”](#): A detailed description of the RESTful Terminal Location Web Service

[Chapter 8, “Third Party Call”](#): A detailed description of the RESTful Third Party Call Web Service

[Chapter 9, “Call Notification”](#): A detailed description of the RESTful Call Notification Web Service

[Chapter 10, “Presence”](#): A detailed description of the RESTful Presence Web Service

[Chapter 11, “Payment”](#): A detailed description of the RESTful Payment Web Service

[Chapter 13, “WAP Push”](#): A detailed description of the RESTful WAP Push Web Service

Terminology

The following terms and acronyms are used in this document:

- 3GPP—3rd Generation Partnership Project, a collaborative group of telecom standards bodies
- Account—A registered application or service provider. An account belongs to an account group, which is tied to a common SLA.
- Account group—Multiple registered service providers or applications that share a common SLA
- Administrative user—Someone who has privileges on the Oracle Communications Services Gatekeeper management tool. This person has an administrative user name and password.
- Alarm—The result of an unexpected event in the system, often requiring corrective action.

- API—An application programming interface
- Application—A TCP/IP based, telecom-enabled program accessed from either a telephony terminal or a computer
- Application-facing interface—The interface that Application Service Providers use to interact with Oracle Communications Services Gatekeeper
- Application Service Provider—An organization offering application services to end users through a telephony network
- AS—An application server
- Application Instance—An Application Service Provider from the perspective of internal Oracle Communications Services Gatekeeper administration. An Application Instance has a user name and a password
- CBC—Charging based on the nature of the content delivered, not on time used or simple per-use cost. Content based charging.
- CDR—Charging Data Record
- Communication Service—A facade and an enabler that together form the path through which requests travel in Oracle Communications Services Gatekeeper. Each communication service corresponds to a particular service capability.
- CORBA—Common Object Request Broker Architecture
- CPU—Central Processing Unit
- CRM—Customer Relationship Management
- DMZ—Demilitarized Zone, a physical or logical subnetwork that contains and exposes an organization's external services to a larger, untrusted network
- EAR—Enterprise Archive file
- EJB—Enterprise Java Bean
- Enabler—The Oracle Communications Services Gatekeeper layer that performs policy evaluation, routing, and protocol translation. It provides network-facing interfaces.
- End user—The ultimate consumer of the services that an application provides. An end user can be the same as the network subscriber, as in the case of a prepaid service or the end user can be a non-subscriber, as in the case of an automated mail-ordering application

where the subscriber is the mail-order company and the end user is a customer to this company

- Enterprise Operator—See Application Service Provider
- Enterprise Service Bus—A middleware component that supports messaging, routing, XML data transformation, and service orchestration
- ETSI—The European Telecommunications Standards Institute, a telecom standards body
- Event—A traceable, expected occurrence in the system, of interest to the operator
- EDR—Event Data Record
- EWS—Extended Web Services, a set of Web Services interfaces developed by Oracle offering access to network functionality not covered by Parlay X.
- Facade—A set of interfaces exposed to application service developers. A facade functions as a view of an enabler.
- HA—Mechanisms set up to insure high availability
- HTML—Hypertext Markup Language
- HTTP—Hypertext Transfer Protocol
- INAP—Intelligent Network Application Part, a telephony signalling protocol
- Interceptor Stack—A flexible set of chained evaluation steps used in Oracle Communications Services Gatekeeper
- IP—Internet Protocol
- JDBC—Java Database Connectivity, the Java API for database access
- JEE—Java Enterprise Edition
- JMS—Java Message Service
- JMX—Java Management Extensions
- LDAP—Lightweight Directory Access Protocol
- Location Uncertainty Shape—A geometric shape surrounding a base point specified in terms of latitude and longitude. It is used in terminal location.
- MAP—Mobile Application Part

- Marshall—Record the state and codebase(s) of an object in such a way that when the marshalled object is "unmarshalled," a copy of the original object is obtained, possibly by automatically loading the class definitions of the object.
- Mated pair—Two physically distributed installations of Oracle Communications Services Gatekeeper nodes sharing a subset of data allowing for high availability between the nodes
- MIB—Management Information Base
- MLP—Mobile Location Protocol
- MM7—A multimedia messaging protocol specified by 3GPP
- MMS—Multimedia Message Service or an instance of this service
- MMSC—Multimedia Message Service Center
- Network plug-in—The Oracle Communications Services Gatekeeper module that implements the interface to a network node or OSA/Parlay SCS through a specific protocol
- NS—Network Simulator
- OAM—Operation, Administration, and Maintenance
- OASIS—The Organization for the Advancement of Structured Information Standards, an e-business and web standards consortium
- OCSG—Oracle Communications Services Gatekeeper
- Operator—The party that manages Oracle Communications Services Gatekeeper. Usually the network operator
- On-boarding—Registering applications and service providers to enable their access to Oracle Communications Services Gatekeeper and the underlying network
- ORB—Object request broker
- OSA/Parlay—The Open Service Access interfaces used by a Parlay gateways
- OSS—Operation Support Systems
- Out of the box—The level of functionality available in the default installation of Oracle Communications Services Gatekeeper
- PAP—Push Access Protocol
- Parlay—The Parlay Group, a telecom standards body

- Parlay Gateway—A telecom gateway implementing Parlay interfaces
- Parlay X—A set of telecom Web Services interfaces specified by the Parlay Group
- Plug-in—See Network Plug-in
- Plug-in Manager—The OCSG module charged with routing an application-initiated request to the appropriate network plug-in
- POJO—Plain Old Java Object
- Presence information—A status indicator that conveys the accessibility and the willingness of a potential communication partner
- Presentity—A supplier of presence information.
- PRM—Partner Relationship Management
- Quotas—An access rule based on an aggregated number of invocations. See also Rates
- RAM—Random Access Memory
- RAID—Redundant Array of Independent Disks
- Rates—An access rule based on allowable invocations per time period. See also Quotas
- RESTful—Interfaces that follow Representation State Transfer style
- Rf—The Diameter offline charging mode
- RMI—Remote Method Invocation
- Ro—The Diameter online charging mode
- SAML—Security Assertion Markup Language
- SCF—Service Capability Function or Service Control Function, in the OSA/Parlay sense.
- SCS—Service Capability Server, in the OSA/Parlay sense. Oracle Communications Services Gatekeeper can interact with these on its network-facing side
- Service Capability—Support for a specific kind of traffic within Oracle Communications Services Gatekeeper. Defined in terms of communication services
- SIP—Session Initiation Protocol
- SLA—A service level agreement

- SMPP—Short Message Peer-to-Peer Protocol
- SMS—Short Message Service, or an instance of this service
- SMSC—Short Message Service Center
- SNMP—Simple Network Management Protocol
- SOA—Service Oriented Architecture
- SOAP—A protocol for exchanging Web Services messages
- SPI—Service Provider Interface
- SQL—Structured Query Language
- SS7—Signalling System #7, a signaling protocol used in traditional telecom networks
- Subscriber—A person or organization that signs up for access to an application. The subscriber is charged for the application service usage. See End user
- TCP—Transmission Control Protocol
- TUPS—Transaction Units Per Second
- URI—Uniform Resource Identifier
- URL—Uniform Resource Locator
- USSD—Unstructured Supplementary Service Data
- VAS—Value Added Service
- VASP—Value Added Service Provider
- VLAN—Virtual Local Area Network
- VPN—Virtual Private Network
- W3C—The World Wide Web Consortium, a web standards group
- WAP Push—A protocol for sending WAP content (an encoded message including a link to a WAP address) that is pushed to a subscriber's handset
- Watcher—A consumer of presence information
- WS-Security—An OASIS security standard for Web Services

- WSDL—Web Services Definition Language
- XML—Extensible Markup Language

Related Documentation

This application development guide is a part of the Oracle Communications Services Gatekeeper documentation set. The other documents include:

- [System Administrator's Guide](#)
- [Integration Guidelines for Partner Relationship Management](#)
- [*SDK User Guide*](#)
- [Managing Accounts and SLAs](#)
- [Statement of Compliance and Protocol Mapping](#)
- [Concepts and Architectural Overview](#)
- [Communications Services Reference](#)
- [Handling Alarms](#)
- [Licensing](#)
- [Installation Guide](#)
- [Platform Development Studio - Developer's Guide](#)
- [Application Development Guide](#)

Overview of Oracle Communications Services Gatekeeper

As the worlds of Internet applications and of telephony-based functionality continue to converge, many application developers have become frustrated by the unfamiliar and often complex telecom interfaces that they need to master to add even simple telephony-based features to their programs. By using Oracle Communications Services Gatekeeper, telecom operators can instead offer developers a secure, easy-to-develop-for single point of contact with their networks, made up of simple SOAP and Restful Web Service interfaces that can easily be accessed from the Internet using a wide range of tools and languages.

The following chapter presents an overview of Oracle Communications Services Gatekeeper's functionality, and the ways that application developers can use this functionality to simplify their development projects, including:

- [Basic Concepts](#)
- [Functional Overview](#)
- [Application Testing Workflow](#)

Basic Concepts

There are a few basic concepts you need to understand to create applications that can interact with Oracle Communications Services Gatekeeper:

- [Communication Services](#)
- [Traffic Types](#)

- [Application-initiated Traffic](#)
- [Network-triggered Traffic](#)
- [Management Structures](#)

Communication Services

The basic functional unit in Oracle Communications Services Gatekeeper is the communication service. A communication service consists of a service type (Short Messaging, User Location, etc.), an application-facing interface (also called a “north” interface), and a network-facing interface (also called a “south” interface). A request for service enters through one interface, is subjected to internal processing, including evaluation for policy and protocol translation, and is then sent on using the other interface.

Note: Because a single application-facing interface may be connected to multiple protocols and hardware types in the underlying telecom network, it’s important to understand that an application is communicating, finally, with a specific communication service, and not just the north interface. So in some cases it is possible that an application request sent to two different operators, with different underlying network structures, might behave in slightly different ways, even though the initial request uses exactly the same north interface.

Traffic Types

In some Oracle Communications Services Gatekeeper communication services, request traffic can travel in two directions - from the application to the underlying network and from the underlying network to the application - and in others traffic flows in one direction only.

Application-initiated Traffic

In application-initiated traffic, the application sends a request to Oracle Communications Services Gatekeeper, the request is processed, and a response of some kind is returned synchronously. So, for example, an application could use the Third Party Call interface to set up a call. The initial operation, `Make Call`, is sent to Oracle Communications Services Gatekeeper (which sends it on to the network) and a string that identifies the call is returned to the application synchronously. To query the status of the call, the application makes a new request, `Get Call Information`, using the call identifier string to identify the specific call, and then receives the requested information back from Oracle Communications Services Gatekeeper synchronously.

Network-triggered Traffic

In many cases, application-initiated traffic provides all the functionality necessary to accomplish the desired tasks. But there are certain situations in which useful information may not be immediately available for return to the application. For example, the application might send an SMS to a mobile phone that the user has turned off. The network won't deliver the message until the user turns the phone back on, which might be hours or even days later. The application can poll to find out whether or not the message has been delivered, using `Get SMS Delivery Status`, which functions much like `Get Call Information` described above. But given the possibly extended period of time involved, it would be convenient simply to have the network *notify* the application once delivery to the mobile phone has been accomplished. To do this, two things must happen:

- The application must inform Oracle Communications Services Gatekeeper that it wishes to receive information that originates from the network. It does this by beginning a *notification* via an application-initiated request. Depending on the underlying network configuration, Oracle Communications Services Gatekeeper itself, or the operator using manual steps, informs the underlying network about the kind of data that is requested. These notifications may be status updates, as described above, or, in some instances, may even include short or multimedia messages from a terminal on the telecom network
- The application must arrange to receive the network-triggered information, either by setting up a Bayeaux *subscription* with Oracle Communications Services Gatekeeper or by polling Oracle Communications Services Gatekeeper directly to retrieve them. Notifications are kept in Oracle Communications Services Gatekeeper for retrieval for only limited amounts of time.

Management Structures

In order to help telecom operators organize their relationships with application providers, Oracle Communications Services Gatekeeper uses a hierarchical system of accounts. Each application is assigned a unique application instance ID which is tied to an Application Account. All the Application Accounts that belong to a single entity are assigned to a Service Provider Account. Application Accounts with similar requirements are put into Application Groups and Service Providers with similar requirements are put into Service Provider Groups. Each Application Group is associated with an Application Group Service Level Agreement (SLA) and each Service Provider Group are associated with Service Provider Group SLAs. These SLAs define and regulate the contractual agreements between the telecom operator and the application service provider, and cover such things as which services the application may access and the maximum bandwidth available for use.

Functional Overview

The communication services based on the RESTful Web Services APIs support the following functionality:

- Third Party Call

The Third Party Call interface allows an application to set up a call, get information on that call, cancel the call request before it is successfully completed, or end a call that has been successfully set up

- Call Notification

The Call Notification interface allows an application to set up and remove call notifications (in which the application is informed of a particular state - busy, unreachable, etc. - of the call) or to set up and remove call direction notifications (in which the application is queried for information on handling a call that is in a particular state).

- Short Messaging

The SMS interface allows an application to send an SMS, a ringtone, or a logo, and to fetch SMSs and delivery status reports for the application that have been received and stored on Oracle Communications Services Gatekeeper. It also allows an application to start and stop a notification.

- Multimedia Messaging

The MMS interface allows an application to send an MMS and to fetch information on MMSs for the application that have been received and stored on Oracle Communications Services Gatekeeper. It also allows the application to fetch those messages. The application can also get delivery status on sent messages, and start and stop a notification.

- Terminal Location

The Terminal Location interface allows an application to get a location for an individual terminal or a group of terminals; to get the distance of the terminal from a specific location; and to start and stop notifications, based on geographic location or on a periodic interval.

- Presence

The Presence interface allows an application to act as either of two different parties to a presence interaction: as a presentity or as a watcher. A presentity agrees to have certain data (called attributes) such as current activity, available communication means, and contact addresses made available to others while a watcher is a consumer of such

information. As a watcher, an application can request to subscribe to all or a subset of a presentity's data, poll for that data, and start and end presence notifications. As a presentity, an application can publish presence data about itself, check to see if any new watchers wish to subscribe to its presence data, authorize those watchers it chooses to authorize, block those it wishes not to have access, and get a list of currently subscribed watchers.

- **Payment**

The Payment interface allows an application to charge an amount to an end-user's account using Diameter, refund amounts to that account, and split charge amounts among multiple end-users. An application can also reserve amounts, reserve additional amounts, charge against the reservation or release the reservation.

- **WAP Push**

The WAP Push interface allows an application to send a WAP Push message and to receive status notifications about that message. I

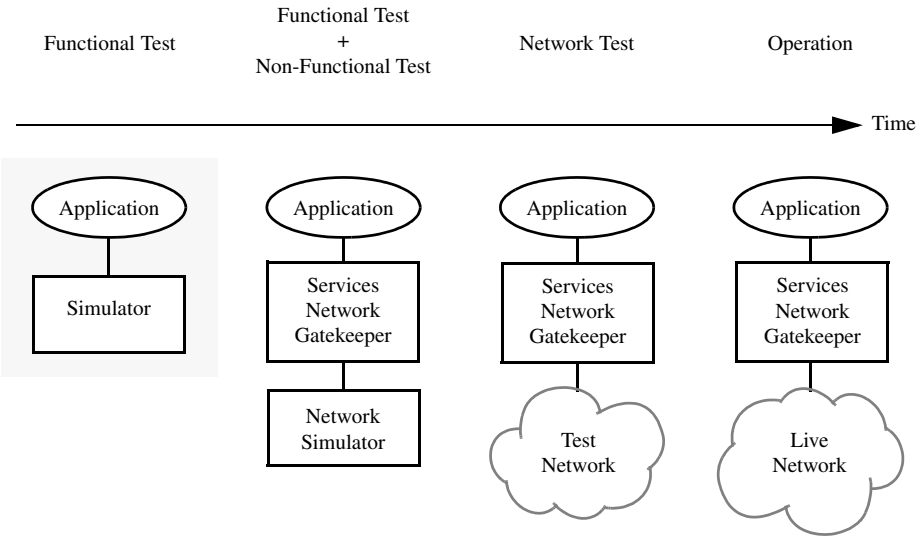
- **Session Manager**

The Session Manager interface allows an application to establish a session with the Oracle Communications Services Gatekeeper operator. Operators can choose to use a session-based mode or not.

Application Testing Workflow

Application testing in a telecom environment is usually conducted in a stepwise manner. For the first step, applications are run against a simulator (Note: the OCSG SDK simulator does not support RESTful interfaces.). Once basic functional issues are sorted through, the application is connected to an instance of the Oracle Communications Services Gatekeeper attached to a network simulator for non-functional testing. Next the application is tested against a test network, to eliminate any network related issues. Finally, the application can be placed into production on a live network. [Figure 2-1](#) shows the complete application test flow, from the developer's functional tests to deployment in a live network. While simulator-based tests may be performed in-house by an Application Service Provider, the other tests require the cooperation of the target network operator.

Figure 2-1 Application Testing Cycle



Interacting with the RESTful Facade

In order to interact with Oracle Communications Services Gatekeeper, applications use the RESTful Web Services Facade, or set of interfaces. The applications use standard HTTP requests in certain specialized ways. They add specific information to the header, and, in some cases, they send their message payload as an attachment. The following chapter provides an overview of how Oracle Communications Services Gatekeeper's RESTful Web Services work, including:

- [Format](#)
- [Methods](#)
- [Authentication and Security](#)
- [Notifications and Publish/Subscribe](#)
- [Errors and Exceptions](#)

Format

Requests made using the RESTful facade have three basic parts:

- [URI](#)
- [Header](#)
- [Body](#)

In addition, MMS and WAP Push use HTTP [Attachments](#).

Note: A generated HTML version of interface information is available from the root of each RESTful interface. For SMS, for example, the HTML is available at `http://host:port/rest/sms/index.html` where host and port are the values appropriate for an instance of Service Gatekeeper.

URI

The most important part of a request made to the RESTful facade is the URI. It is made up of a sequence of sections, concatenated together. The general format of the URI is as follows:

```
https://host:port/{rest-facade-context-root}/{URI}/{path-info-param}/query-string
```

- host:port

The hostname and port of the Oracle Communications Services Gatekeeper installation

- rest-facade-context-root

The location of the set of resources that the particular interface uses. For out-of-the-box communication services, this is always `rest`

- URI

The location of a specific kind of functionality within the interface.

- path-info-param

An identifier of a specific resource. Does not occur in all URIs

- query-string

A set of name-value pairs that describes what is being requested. Does not occur in all URIs.

Header

The headers of all requests must include the HTTP Basic credentials “userid/password” as described in RFC 2617. The userid is the application’s application instance ID (for more information on the application instance ID, see [Management Structures](#) in the “Overview” chapter of this document.) Every request is authenticated by Oracle Communications Services Gatekeeper and must have these credentials.

Requests may also include other header data, including

- Session ID, if the Oracle Communications Services Gatekeeper installation is running in session mode. This is a key-value pair with key=`X-Session-ID`. For more information on sessions, see [Chapter 4, “Session Manager”](#).
- Service correlation ID, if the application wishes to set up service correlation. This is a key-value pair with key=`X-SCID`. For more information on service correlation, see [Service Correlation](#)
- Tunneled parameters, if the application wishes to supply parameters to be passed on to the network that are not supported in the RESTful interface itself. These are key-value pairs. For example:

- `X-Param-Keys=key1,key2`
- `X-Param-Values=value1,value=2`

For more information on tunneled parameters, sometimes called xparams, see [Parameter Tunneling](#)

In addition, the headers to the responses to certain requests may include a `Location` header value in the form of a key/value pair. This information designates the URI for the Bayeux server, used for retrieving notification or delivery status information concerning the initial request.

Service Correlation

In some cases the service that an application provides to its end-users may involve accessing multiple Oracle Communications Services Gatekeeper communication services. For example, a mobile user might send an SMS to an application asking for the pizza place nearest to his current location. The application then makes a Terminal Location request to find the user’s current location, looks up the address of the closest pizza place, and then sends the user an MMS with all the appropriate information. Three Oracle Communications Services Gatekeeper communication services are involved in executing what for the application is a single service. In order to be able to correlate the three communication service requests, Oracle Communications Services Gatekeeper uses a Service Correlation ID, or SCID. This is a string that is captured in all the CDRs and EDRs generated by Oracle Communications Services Gatekeeper. The CDRs and EDRs can then be orchestrated in order to provide special treatment for a given chain of service invocations, by, for example, applying charging to the chain as a whole rather than to the individual invocations.

The SCID is not provided by Oracle Communications Services Gatekeeper. When the chain of services is initiated by an application-initiated request, the application must provide, and ensure the uniqueness of, the SCID within the chain of service invocations.

Note: In certain circumstances, it is also possible for a custom service correlation service to supply the SCID, in which case it is the custom service's responsibility to ensure the uniqueness of the SCID.

When the chain of services is initiated by a network-triggered request, Oracle Communications Services Gatekeeper calls an external interface to get the SCID. This interface must be implemented by an external system. No utility or integration is provided out of the box; this must be a part of a system integration project. It is the responsibility of the external system to provide, and ensure the uniqueness of, the SCID.

Parameter Tunneling

Parameter tunneling is a feature that allows an application to send additional parameters through Oracle Communications Services Gatekeeper so that a plug-in can use these parameters. This feature makes it possible for an application to tunnel parameters that are not defined in the application-facing interface but may be of use in Oracle Communications Services Gatekeeper's interaction with the underlying network. It can be seen as an extension to the application-facing interface.

Body

The body of the requests made to the RESTful facade may include additional data required for the request. This data is formatted using JSON. Depending on the request type, the body may be empty.

Attachments

The RESTful-based communication services for Multimedia Messaging and WAP Push use HTTP attachments to transport their content. Both interfaces support multipart/form-data POST requests. The content must have the following MIME header: `Content-ID:messagePart` `Content-Type:application/json`. In the RESTful interfaces delivered out of the box in Oracle Communications Services Gatekeeper multiple attachments are supported in both application-initiated and network-triggered messages.

Methods

The RESTful interfaces use standard HTTP methods in conjunction with the formats described in [Format](#). In general, the RESTful facade uses these methods in the following way:

HTTP GET

Retrieves the state of a resource.

- The request is a query string and the resource is identified in that string. Complex object data structures are represented by JSON encoded strings in the URI. The body is empty. For example, to get the delivery status of an SMS, GET on `http://host:port/rest/sms/delivery-status/requestID`.
- The response is an HTTP status code:
 - Success
 - 200 with the body of the response containing a JSON representation of the resource
 - Error
 - 500, for Service Exceptions of value SVC001
 - 400, for all other Service and Policy Exceptions and other general errors such as Resource Not Found.

HTTP POST

Creates a new resource that does not yet have a URI; the request URI represents a parent or factory resource and multiple requests can create multiple new resources.

- The URI of the request represents the factory resource. For example, to send an SMS, POST on `http://host:port/rest/sms/messages`. The body of the request contains a JSON representation of the resource to be created. The content type is `application/json`.
- The response is an HTTP status code:
 - Success
 - 201 with a JSON representation of the new resource in the body and a special `Location` header value that is equal to the URI of the new resource, if applicable.
 - 204 with an empty body
 - Error
 - 500, for Service Exceptions of value SVC001
 - 400, for all other Service and Policy Exceptions and other general errors such as Resource Not Found.

HTTP PUT

Creates a resource that has a pre-determined URI; this can be used to update a resource (or to start a stateful process).

- The URI of the request is the pre-determined resource. For example, to start SMS notification, PUT on `http://host:port/rest/sms/notification/`. The body of the request contains a JSON representation of the resource to be created. The content type is `application/json`.
- The response is an HTTP status code:
 - Success
 - 201 and, if a new resource (for example, a notification) is created, a special `Location` header value that is equal to the URI of the new resource. The body contains a JSON representation of the new resource
 - Error
 - 409 if a resource already exists and cannot be updated. For Service Exceptions of value `SVC005` and `SVC008`.
 - 500, for Service Exceptions of value `SVC001`
 - 400, for all other Service and Policy Exceptions and other general errors such as `Resource Not Found`.

HTTP DELETE

Removes a resource.

- The URI of the request is a string identifying the resource. For example, to stop an SMS notification, DELETE on `http://host:port/rest/sms/notification/correlatorID`. The body is empty.
- The response is an HTTP status code:
 - Success
 - 204 with an empty body
 - Error
 - 404 for Service Exceptions of value `SVC002`
 - 500, for Service Exceptions of value `SVC001`

- 400, for all other Service and Policy Exceptions and other general errors such as Resource Not Found.

Authentication and Security

The RESTful interfaces use HTTP basic authentication, using username/password. SSL is required. For more information on HTTP basic authentication, see RFC 2617 at <http://www.ietf.org/rfc/rfc2617.txt>

Notifications and Publish/Subscribe

In order to support the essentially asynchronous nature of notifications in an HTTP environment, the RESTful interfaces in Oracle Communications Services Gatekeeper rely on the publish/subscribe model supported by the Publish-Subscribe Server functionality of Oracle WebLogic Server 10.3. For more information on this model, please see “Using the HTTP Publish-Subscribe Server” in *Developing Web Applications, Servlets, and JSPs For Oracle WebLogic Server* at http://download.oracle.com/docs/cd/E12840_01/wls/docs103/webapp/.

The application client must use the Bayeux protocol to communicate with the server. In this model, clients subscribe to a channel (similar to a topic in JMS) and receive messages as they become available. For more information on the Bayeux protocol, see the “Bayeux Protocol 1.0draft1” document at <http://svn.xantus.org/shortbus/trunk/bayeux/bayeux.html>.

The Bayeux client manages connecting to the server and subscribing to a channel. If the channel does not exist when the client subscribes to it, the channel is created. The channel name must begin with “/bayeux/\${appInstanceId}”, where “appInstanceId” is the application’s application instance account ID. (For more information on application instances, see “Creating and Maintaining Service Provider and Application Accounts” in *Managing Accounts and SLAs*, another document in this set.) Applications can subscribe only to their own notifications: attempts to subscribe to notifications for other applications are rejected.

Note: The mechanisms for connecting to the server and subscribing to a channel are covered by the Bayeux protocol itself. Oracle Communications Services Gatekeeper delivers notifications to the channel; it is the client’s responsibility to interact with the Publish-Subscribe Server to access them. These mechanisms are outside the scope of the Oracle Communications Services Gatekeeper RESTful APIs.

Once the Bayeux client has connected to the server and subscribed to a channel, the RESTful client can start the notification. It does this by supplying the channel name as the endpoint in the start notification request.

Errors and Exceptions

If an operation is unsupported, a status code of 405 is returned. Policy Exceptions and Service Exceptions are mapped as described in [Methods](#). In addition, Service Exception and Policy Exception objects are represented in the response body as JSON with the following form:

```
{ "error":  
    {  
        "type": "class name of the error object"  
        "message": "error message, variable substitution will be performed  
for PX exceptions"  
    }  
}
```

Session Manager

Operations

Although a particular operator may choose to run an installation in a sessionless mode, by default Oracle Communications Services Gatekeeper requires that applications acquire a Oracle Communications Services Gatekeeper session before beginning to send request traffic. The RESTful Session Manager interfaces provide access to the Session Manager Web Service. Applications use the Session Manager interface to get a session ID. The application then adds this session ID to the header of all its requests. Oracle Communications Services Gatekeeper uses this value to keep track of all the traffic that an application sends for the duration of the session.

The RESTful Session Manager interfaces allow an application to get a session ID, to get the time remaining in a session's lifetime, and to destroy a session.

Note: An HTML version of this information is at

`http://host:port/rest/session_manager/index.html` where host and port depend on the Oracle Communications Services Gatekeeper installation.

Get Session

Creates a session with an ID. This ID is used in the `X-Session-ID` header of all subsequent traffic requests.

Request

URI

`http://host:port/rest/session_manager/sessions`

HTTP Method

POST

Response

Body

```
{ "getSessionReturn": "String" }
```

The session ID, a string

Error Response

[General Exception](#)

Get Session Remaining Lifetime

Gets the time remaining in this session, in milliseconds.

Request

URI

`http://host:port/rest/session_manager/session/${sessionId}`

`${sessionId}`: The session ID, a string

HTTP Method

GET

Response

Body

```
{ "getSessionRemainingLifeTimeReturn": "Integer" }
```

The remaining time, in milliseconds, an integer.

Error Response

[General Exception](#)

Destroy a Session

Destroys this session.

Request

URI

`http://host:port/rest/session_manager/session/${sessionId}`

`${sessionId}`: The session ID, a string

HTTP Method

DELETE

Response

Body

```
{"destroySessionReturn": "Boolean"}
```

A boolean, true if the session was destroyed

Error Response

[General Exception](#)

Errors

General Exception

```
{"error": {  
  "type": "com.bea.wsdل.wlcp.wlng.session_manager.service.GeneralException"  
  "message": "String"  
}}
```


Short Messaging

Operations

The RESTful Short Messaging interfaces allow an application to send an SMS, a ringtone, or a logo, and to fetch SMSs and delivery status reports for the application that have been received and stored on Oracle Communications Services Gatekeeper. They also allows an application to start and stop a notification.

In the listings below, **bold** indicates a required attribute, <> a descriptive comment.

Note: An HTML version of this information is at

`http://host:port/rest/sms/index.html` where host and port depend on the Oracle Communications Services Gatekeeper installation.

Send Sms

Sends a text SMS

Request

URI

`http://host:port/rest/sms/messages`

Request Content-Type

`application/json`

Body

```
{
  "addresses": [ "URI" ], <The recipient(s)>
  "message": "String", <The message to be sent>
  "charging": { <This entire attribute is optional>
    "description": "String", <Text to be used for information and billing>
    "amount": "BigDecimal", <The amount to be charged>
    "code": "String", <A charging code, from an existing contractual description>
    "currency": "String" <A currency identifier as defined in ISO 4217 [9]>
  },
  "receiptRequest": { <This entire attribute is optional>
    "correlator": "String", <A correlator used to connect the receipt to the initial message>
    "endpoint": "URI", <The endpoint address to which the receipt should be delivered>
    "interfaceName": "String" <Descriptive only, a string to identify the type of receipt>
  },
  "senderName": "String" <The sender of the SMS>
}
```

Note: The request bodies for [Send Sms](#), [Send Sms Ringtone](#), and [Send Sms Logo](#) may contain a `receiptRequest` attribute in the input message. This is a way of requesting a delivery receipt for the message being sent, and it includes an application endpoint for returning the delivery notification. Currently it is possible to specify either a SOAP (a Web Service implemented by the application) endpoint or a RESTful endpoint (a channel beginning with “/bayeux/\${appInstanceId}”). Specifying a SOAP endpoint for a notification for a message sent using RESTful interfaces, however, works only if the SOAP and RESTful interfaces reside in the same cluster. For more information on channels, see [Notifications and Publish/Subscribe](#).

HTTP Method

POST

Response

Response Content-Type

application/json

Body

```
{ "result": "String" }
```

The request identifier, a string used to identify the request for later correlation

Header

key= "Location" value = "host:port/rest/sms/delivery-status/\${result}"

`${result}`: The request identifier returned in the body

The URI of the pub/sub server

Error Response

[Policy Exception](#)

[Service Exception](#)

Send Sms Ringtone

Sends a ringtone

Request

URI

`http://host:port/rest/sms/ringtones`

Request Content-Type

application/json

Body

```
{
  "addresses": [ "URI" ], <The recipient(s)>
  "ringtone": "String", <The ringtone data in RTX format>
}
```

```
"smsFormat": "Ems|SmartMessaging",<The ringtone encoding>
"charging": {<This entire attribute is optional>
  "description": "String",<Text to be used for information and billing>
  "amount": "BigDecimal",<The amount to be charged>
  "code": "String",<A charging code, from an existing contractual description>
  "currency": "String" <A currency identifier as defined in ISO 4217 [9]>
},
"receiptRequest": {<This entire attribute is optional>
  "correlator": "String",<A correlator used to connect the receipt to the initial message>
  "endpoint": "URI",<The endpoint address to which the receipt should be delivered>
  "interfaceName": "String" <A descriptive string to identify the type of receipt>
},
"senderName": "String"
}
```

HTTP Method

POST

Response

Response Content-Type

application/json

Body

```
{"result": "String"}
```

The request identifier, a string used to identify the request for later correlation

Header

key= "Location" value = host:port/rest/sms/delivery-status/\${result}"

\${result}: A string, the request identifier returned in the body

The URI of the pub/sub server

Error Response

[Policy Exception](#)

[Service Exception](#)

Send Sms Logo

Sends a logo

Request

URI

`http://host:port/rest/sms/logos`

Request Content-Type

`application/json`

Body

```
{
  "addresses": [ "URI" ], <The recipient(s)>
  "image": "base64Binary", <The logo data, base64 encoded image in gif, jpg, or png
format>
  "smsFormat": "Ems|SmartMessaging", <The logo encoding>
  "charging": { <This entire attribute is optional>
    "description": "String", <Text to be used for information and billing>
    "amount": "BigDecimal", <The amount to be charged>
    "code": "String", <A charging code, from an existing contractual description>
    "currency": "String" <A currency identifier as defined in ISO 4217 [9]>
  },
  "receiptRequest": { <This entire attribute is optional>
    "correlator": "String", <A correlator used to connect the receipt to the initial message>
    "endpoint": "URI", <The endpoint address to which the receipt should be delivered>
    "interfaceName": "String" <A descriptive string to identify the type of receipt>
  },
}
```

```
"senderName": "String"  
}
```

HTTP Method

POST

Response

Response Content-Type

application/json

Body

```
{"result": "String"}
```

The request identifier, a string used to identify the request for later correlation

Header

key=“Location” value = "host:port/rest/sms/delivery-status/\${result}"

\${result}: A string, the request identifier returned in the body

The URI of the pub/sub server, for getting the delivery status

Error Response

[Policy Exception](#)

[Service Exception](#)

Get Received Sms

Polls Oracle Communications Services Gatekeeper for SMSs for the application that have been received from the network

Request

URI

http://host:port/rest/sms/receive-messages/\${registrationIdentifier}

\${registrationIdentifier}: A String that is set up with the off-line provisioning step that enables the application to receive notification of SMS reception according to specified criteria

HTTP Method

POST

Response

Response Content-Type

application/json

Body

```
{ "result": [ {  
    "message": "String" , <The text of the SMS message>  
    "senderAddress": "URI" , <The sender of the message>  
    "smsServiceActivationNumber": "URI" , <The number associated with the invoked  
    Message service, that is the destination address used to send the message>  
    "dateTime": "Calendar" <The time the message was sent in ISO 8601 extended format >  
} ] }
```

Error Response

[Policy Exception](#)

[Service Exception](#)

Get Sms Delivery Status

Gets the delivery status of a previously sent message

Request

URI

`http://host:port/rest/sms/delivery-status/${requestIdentifier}`
`${requestIdentifier}`: The identifier returned in the result object of the initial request

HTTP Method

GET

Note: It is possible to query the delivery status of an SMS only if a callback reference was not defined (in the `receiptRequest` attribute of the input message) when the SMS was sent. If a callback reference was defined, `Notify Sms Delivery Receipt` is invoked by Oracle Communications Services Gatekeeper and the delivery status is not stored. If the delivery status is stored in Oracle Communications Services Gatekeeper, it is stored for a configurable period of time.

Response

Response Content-Type

application/json

Body

```
{ "result": [{ <This is an optional attribute>
  "address": "URI", <The address to which the initial message was sent>
  "deliveryStatus":
    "DeliveredToNetwork|DeliveryUncertain|DeliveryImpossible|MessageWaiting|De
liveredToTerminal|DeliveryNotificationNotSupported" <Possible status values: see
Table 5-1>
}] }
```

Table 5-1 Enumeration values

Value	Meaning
DeliveredToNetwork	Successful delivery to network. In case of concatenated messages, only when all the SMS-parts have been successfully delivered to the network
DeliveryUncertain	Delivery status unknown, for example, if it was handed off to another network.
DeliveryImpossible	Unsuccessful delivery; the message could not be delivered before it expired.
MessageWaiting	The message is still queued for delivery. This is a temporary state, pending transition to one of the preceding states.

Table 5-1 Enumeration values

Value	Meaning
DeliveredToTerminal	Successful delivery to terminal. In case of concatenated messages, only when all the SMS-parts have been successfully delivered to the terminal.
DeliveryNotificationNotSupported	Unable to provide delivery receipt notification.

Error Response

[Policy Exception](#)

[Service Exception](#)

Start Sms Notification

Starts a notification for the application

Request

URI

`http://host:port/rest/sms/notification`

Request Content-Type

`application/json`

Body

```
{
  "reference": {<Notification endpoint information>
    "correlator": "String", <A correlator used to identify the notification>
    "endpoint": "URI", <The endpoint address to which the notification should be delivered>
    "interfaceName": "String" <A descriptive string to identify the type of notification>
  },
  "smsServiceActivationNumber": "URI", <The number associated with the invoked
  Message service, that is the destination address used to send the message>
```

```
"criteria": "String" <Text to match against in determining whether or not the application  
should receive the notification.  
}
```

HTTP Method

PUT

Response

Note: For the actual body of the notification as it is delivered to the designated endpoint, see [Notifications and Receipts](#).

Header

key= "Location" value = "host:port/rest/sms/notifications"

The URI of the pub-sub server

Error Response

[Policy Exception](#)

[Service Exception](#)

Stop Sms Notification

Stops a notification for network-triggered messages in Oracle Communications Services Gatekeeper

Request

URI

http://host:port/rest/sms/notification/\${correlator}

\${correlator}: A string, the correlator that identifies the notification, passed in with the initial start notification request

HTTP Method

DELETE

Response

Empty

Error Response

[Policy Exception](#)

[Service Exception](#)

Notifications and Receipts

The content of notifications and receipts delivered to the designated endpoint

Notify Sms Reception

```
{ "notifySmsReception": {
  "correlator": "String", <A correlator used to identify the notification>
  "message": {
    "message": "String", <The contents of the received SMS>
    "senderAddress": "URI", <The sender of the SMS>
    "smsServiceActivationNumber": "URI", <The number associated with the invoked
Message service, that is the destination address used to send the message>
    "dateTime": "Calendar" <The time the message was sent in ISO 8601 extended format >
  }
}}
```

Notify Sms Delivery Receipt

```
{ "notifySmsDeliveryReceipt": {
  "correlator": "String", <A correlator used to connect the receipt to the initial message>
  "deliveryStatus": {
    "address": "URI", <The recipient of the initial message>
    "deliveryStatus":
"DeliveredToNetwork|DeliveryUncertain|DeliveryImpossible|MessageWaiting|De
liveredToTerminal|DeliveryNotificationNotSupported" <Possible status values: see
Table 5-1>
  }
}
```

```
}}
```

Errors

The content of possible error messages.

Service Exception

```
{ "error": {  
  "type": "org.csapi.schema.parlayx.common.v2_1.ServiceException"  
  "message": "String"  
}}
```

Policy Exception

```
{ "error": {  
  "type": "org.csapi.schema.parlayx.common.v2_1.PolicyException"  
  "message": "String"  
}}
```

Multimedia Messaging

Operations

The RESTful Multimedia Messaging interfaces allow an application to send an MMS and to fetch information on MMSs for the application that have been received and stored on Oracle Communications Services Gatekeeper. They allow the application to fetch those messages. The application can also get delivery status on sent messages, and start and stop a notification.

Note: The actual message is sent as an attachment. See [Attachments](#) for more information.

In the listings below, **bold** indicates a required attribute, <> a descriptive comment.

Note: An HTML version of this information is at

`http://host:port/rest/multimedia_messaging/index.html` where host and port depend on the Oracle Communications Services Gatekeeper installation.

Send Message

Sends a multimedia message

Request

URI

`http://host:port/rest/multimedia_messaging/messages`

Request Content-Type

`multipart/form-data`

Message Part Name

messagePart

Message Part Content-Type

application/json

Message Part Content

```
{
  "addresses": [ "URI" ], <The recipient(s)>
  "charging": { <This entire attribute is optional>
    "description": "String", <Text to be used for information and billing>
    "amount": "BigDecimal", <The amount to be charged>
    "code": "String", <A charging code, from an existing contractual description>
    "currency": "String" <A currency identifier as defined in ISO 4217 [9]>
  },
  "priority": "Default|Low|Normal|High", <The message priority>
  "receiptRequest": { <This entire attribute is optional>
    "correlator": "String", <A correlator used to connect the receipt to the initial message>
    "endpoint": "URI", <The endpoint address to which the receipt should be delivered>
    "interfaceName": "String" <Descriptive only, a string to identify the type of receipt>
  },
  "senderAddress": "String", <The sender of the MMS>
  "subject": "String" <The subject line of the MMS>
}
```

Note: The request bodies for [Send Message](#) may contain a `receiptRequest` attribute in the input message. This is a way of requesting a delivery receipt for the message being sent, and it includes an application endpoint for returning the delivery notification. Currently it is possible to specify either a SOAP (a Web Service implemented by the application) endpoint or a RESTful endpoint (a channel beginning with `"/bayeux/${appInstanceId}"`). Specifying a SOAP endpoint for a notification for a message sent using RESTful interfaces, however, works only if the SOAP and RESTful interfaces reside in the same cluster. For more information on channels, see [Notifications and Publish/Subscribe](#).

HTTP Method

POST

Response**Response Content-Type**

application/json

Body

```
{ "result": "String" }
```

The request identifier, a string used to identify the request for later correlation

Header

key= "Location" value = host:port/rest/multimedia_messaging/delivery-status/\${result}"

\${result}: The request identifier returned in the body

The URI of the pub/sub server

Error Response

[Service Exception](#)

[Policy Exception](#)

Get Received Messages

Retrieves an array of network-triggered messages for the application that have arrived at Oracle Communications Services Gatekeeper. If the content of the messages is pure ASCII, the response body contains the message. Otherwise the response body contains an identifier that can be used to fetch the actual message.

Request**URI**

http://host:port/rest/multimedia_messaging/receive-messages

Request Content-Type

application/json

Body

```
{  
  "registrationIdentifier": "String", <A String that is set up with the off-line  
  provisioning step that enables the application to receive notification of MMS reception according  
  to specified criteria>a  
  
  "priority": "Default|Low|Normal|High" <The priority of the messages to poll for. All  
  messages of the specified priority and higher will be retrieved. The default value is the same as  
  Low: all messages are returned.>  
}
```

HTTP Method

POST

Response

Response Content-Type

application/json

Body

```
{ "result": [{  
  "messageServiceActivationNumber": "String", <The number associated with the  
  invoked Message service, that is the destination address used to send the message>  
  "priority": "Default|Low|Normal|High", <The message priority>  
  "senderAddress": "URI", <The sender of the message>  
  "dateTime": "Calendar", <The time the message was sent in ISO 8601 extended format >  
  "message": "String", <The text of the message, if the message is purely ASCII text>  
  "messageIdentifier": "String", <An identifier used for fetching the message if it is not  
  pure ASCII text>  
  "subject": "String" <The subject line of the MMS>  
}]}
```


Error Response

[Service Exception](#)

[Policy Exception](#)

Get Message

Fetches a network-triggered message for the application from Oracle Communications Services Gatekeeper. The actual message is returned as an attachment.

Request

URI

`http://host:port/rest/multimedia_messaging/message/${messageRefIdentifier}`

`${messageRefIdentifier}`: A string, the message identifier from [Get Received Messages](#) or a Notify Message Reception.

HTTP Method

GET

Response

Response Content-Type

multipart/mixed

Body

Empty

Error Response

[Service Exception](#)

[Policy Exception](#)

Get Message Delivery Status

Gets the delivery status of a previously sent message

Request

URI

`http://host:port/rest/multimedia_messaging/delivery-status/${requestIdentifier}`

`${requestIdentifier}`: The identifier returned in the result object of the initial request

HTTP Method

GET

Note: It is possible to query the delivery status of an MMS only if a callback reference was not defined (in the `receiptRequest` attribute of the input message) when the MMS was sent. If a callback reference was defined, Notify Message Delivery Receipt is invoked by Oracle Communications Services Gatekeeper and the delivery status is not stored. If the delivery status is stored in Oracle Communications Services Gatekeeper, it is stored for a configurable period of time.

Response

Response Content-Type

application/json

Body

```
{ "result": [ {<This is an optional attribute>
  "address": "URI", <The address to which the initial message was sent>
  "deliveryStatus":
    "DeliveredToTerminal|DeliveryUncertain|DeliveryImpossible|MessageWaiting|D
eliveredToNetwork|DeliveryNotificationNotSupported" <Possible status values: see
Table 6-1>
  } ] }
```

Table 6-1 Enumeration values

Value	Meaning
DeliveredToNetwork	Successful delivery to network enabler responsible for distributing the message further in the network.
DeliveryUncertain	Delivery status unknown, for example, if it was handed off to another network.
DeliveryImpossible	Unsuccessful delivery; the message could not be delivered before it expired.
MessageWaiting	The message is still queued for delivery. This is a temporary state, pending transition to one of the preceding states.
DeliveredToTerminal	Successful delivery to terminal.
DeliveryNotificationNotSupported	Unable to provide delivery receipt notification.

Error Response

[Service Exception](#)

[Policy Exception](#)

Start MMS Notification

Starts a notification for the application

Request

URI

`http://host:port/rest/multimedia_messaging/notification`

Request Context-Type

`application/json`

Body

```
{
```

```
"messageServiceActivationNumber": "URI" , <The number associated with the invoked
Message service, that is the destination address used to send the message>
"reference": {
  "correlator": "String" , <A correlator used to identify the notification>
  "endpoint": "URI" , <The endpoint address to which the notification should be delivered>
  "interfaceName": "String" <A descriptive string to identify the type of notification>
},
"criteria": "String" <Text to match against in determining whether or not the application should
receive the notification>
}
```

HTTP Method

PUT

Response

Note: For the actual body of the notification as it is delivered to the designated endpoint, see [Notifications and Receipts](#).

Header

key= "Location" value = host:port/rest/multimedia_messaging/notifications"

The URI of the pub-sub server

Error Response

[Service Exception](#)

[Policy Exception](#)

Stop Message Notification

Stops a notification for the application

Request

URI

http://host:port/rest/multimedia_messaging/notification/\${correlator}

`${correlator}`: A string, the correlator that identifies the notification, passed in with the initial start notification request

HTTP Method

DELETE

Response

Empty

Error Response

[Service Exception](#)

[Policy Exception](#)

Notifications and Receipts

The content of notifications and receipts delivered to the designated endpoint

Notify Message Reception

```
{ "notifyMessageReception": {
  "correlator": "String", <A correlator used to identify the notification>
  "message": {
    "messageServiceActivationNumber": "String", <The number associated with the
    invoked Message service, that is the destination address used to send the message>
    "priority": "Default|Low|Normal|High", <The message priority>
    "senderAddress": "URI", <The sender of the MMS>
    "dateTime": "Calendar", <The time the message was sent in ISO 8601 extended format >
    "message": "String", <The text of the message, if the message is purely ASCII text>
    "messageIdentifier": "String", <An identifier used for fetching the message if it is
    not pure ASCII text>
    "subject": "String" <The subject line of the MMS>
  }
}}
```

Notify Message Delivery Receipt

```
{ "notifyMessageDeliveryReceipt": {  
  "correlator": "String", <A correlator used to connect the receipt to the initial message>  
  "deliveryStatus": {  
    "address": "URI", <The recipient of the initial message>  
    "deliveryStatus":  
      "DeliveredToNetwork|DeliveryUncertain|DeliveryImpossible|MessageWaiting|De  
liveredToTerminal|DeliveryNotificationNotSupported" <Possible status values: see  
Table 6-1>  
  }  
}}
```

Errors

The content of possible error messages.

Service Exception

```
{ "error": {  
  "type": "org.csapi.schema.parlayx.common.v2_1.ServiceException"  
  "message": "String"  
}}
```

Policy Exception

```
{ "error": {  
  "type": "org.csapi.schema.parlayx.common.v2_1.PolicyException"  
  "message": "String"  
}}
```

Terminal Location

Operations

The RESTful Terminal Location interfaces allow an application to get a location for an individual terminal or a group of terminals; to get the distance of the terminal from a specific location; and to start and stop notifications, based on geographic location or on a periodic interval.

In the listings below, **bold** indicates a required attribute, <> a descriptive comment.

Note: An HTML version of this information is at `http://host:port/rest/terminal_location/index.html` where host and port depend on the Oracle Communications Services Gatekeeper installation.

Get Location

Gets the location of a single terminal

Note: If a group URI is provided, an error response is returned.

Request

URI

```
http://host:port/rest/terminal_location/location?query=${query}
${query}:
{
```

"acceptableAccuracy": "Integer" ,<The range that the application considers useful. If the location cannot be determined within this range, the application would prefer not to receive the information.>

"address": "URI" ,<address of the terminal to be located>

"requestedAccuracy": "Integer" ,<The range over which the application wishes to receive location information. This may influence the choice of location technology to use (for instance, cell sector location may be suitable for requests specifying 1000 meters, but GPS technology may be required for requests below 100 meters).>

"tolerance": "NoDelay|LowDelay|DelayTolerant" ,<The priority of response time versus accuracy. See [Table 7-1](#) for more information.>.

"maximumAge": {<The maximum acceptable age of the location information>

"metric": "Millisecond|Second|Minute|Hour|Day|Week|Month|Year" , <The metric to use for time measurement>

"units": "Integer" <The number of units of the metric>

},

"responseTime": {<The maximum response time that the application can accept>

"metric": "Millisecond|Second|Minute|Hour|Day|Week|Month|Year" , <The metric to use for time measurement>

"units": "Integer"<The number of units of the metric>

}

}

Table 7-1 Enumeration values

Value	Meaning
NoDelay	The server should immediately return any location estimate that it currently has. If no estimate is available, the server return a failure indication. It may optionally initiate procedures to obtain a location estimate (e.g. to be available for a later request

Table 7-1 Enumeration values

Value	Meaning
LowDelay	The response time is more important than requested accuracy. The server attempts to fulfil any accuracy requirement, but not if it adds delay. A quick response with lower accuracy is more desirable than waiting for a more accurate response.
DelayTolerant	The network is expected to return a location with the requested accuracy even if this means not complying with the requested response time.

HTTP Method

GET

Response

ResponseContent-Type

application/json

Body

```
{ "result": {  
  "accuracy": "Integer", <Accuracy of location in meters>  
  "latitude": "Float", <Location latitude>  
  "longitude": "Float", <Location longitude>  
  "timestamp": "Calendar", <Date and time the location was collected in ISO 8601 extended  
format >  
  "altitude": "Float" <Location altitude>  
}}
```

Error Response

[Service Exception](#)

[Policy Exception](#)

Get Location For Group

Gets the location for a group of terminals.

Request

URI

```
http://host:port/rest/terminal_location/location?queryForGroup=${queryForGroup}
```

```
${queryForGroup}
```

```
{
```

```
    "acceptableAccuracy": "Integer", <The range that the application considers useful. If the location cannot be determined within this range, the application would prefer not to receive the information.>
```

```
    "addresses": [ "URI" ], <Addresses of the terminals to be located>
```

```
    "requestedAccuracy": "Integer", <The range over which the application wishes to receive location information. This may influence the choice of location technology to use (for instance, cell sector location may be suitable for requests specifying 1000 meters, but GPS technology may be required for requests below 100 meters).>
```

```
    "tolerance": "NoDelay|LowDelay|DelayTolerant", <The priority of response time versus accuracy. See Table 7-1 for more information.>
```

```
    "maximumAge": { <The maximum acceptable age of the location information>
```

```
        "metric": "Millisecond|Second|Minute|Hour|Day|Week|Month|Year", <The metric to use for time measurement>
```

```
        "units": "Integer" <The number of units of the metric>
```

```
    },
```

```
    "responseTime": { <The maximum response time that the application can accept>
```

```
        "metric": "Millisecond|Second|Minute|Hour|Day|Week|Month|Year", <The metric to use for time measurement>
```

```
        "units": "Integer" <The number of units of the metric>
```

```
    }
```

```
}
```

HTTP Method

GET

Response

Response Content-Type

application/json

Body

```
{ "result": [ {
  "address": "URI", <Address of the specific terminal for which this item is relevant>
  "reportStatus": "Retrieved|NotRetrieved|Error", <Retrieval status for this address.
This allows for partial reports to avoid timeouts, etc.>
  "currentLocation": { <Location of the terminal. Only provided if reportStatus = Retrived>
    "accuracy": "Integer", <Accuracy of location in meters>
    "latitude": "Float", <Location latitude>
    "longitude": "Float", <Location longitude>
    "timestamp": "Calendar", <Date and time the location was collected in ISO 8601
extended format >
    "altitude": "Float" <Location altitude>
  },
  "errorInformation": { <If reportStatus=Error, this is the error. This provides error
information for a single item out of a group.>
    "messageId": "String", <Fault definition message identifier>
    "text": "String", <Message text, with replacement variables>
    "variables": [ "String" ] <Variables to substitute into text string>
  }
} ] }
```

Error Response

[Service Exception](#)

[Policy Exception](#)

Get Terminal Distance

Gets the distance between the terminal and a specified location

Request

URI

http://host:port/rest/terminal_location/distance?query=\${query}

\${query}:

```
{  
  "address": "URI" ,<The address of the terminal>  
  "latitude": "Float" ,<The latitude of the specified location>  
  "longitude": "Float" <The longitude of the specified location>  
}
```

HTTP Method

GET

Response

Response Context-Type

application/json

Body

```
{ "result": "Integer" }<Distance in meters>
```

Error Response

[Service Exception](#)

[Policy Exception](#)

Start Geographical Notifications

Starts a notification that is based on whether terminals enter or leave a specified geographic location

Request

URI

`http://host:port/rest/terminal_location/geographical-notification`

Request Content-Type

`application/json`

Body

```
{
  "addresses": [ "URI" ], <Addresses of the terminals to monitor>

  "checkImmediate": "Boolean", <Whether the location of the terminals should be checked
as soon as the notification is established>

  "criteria": "Entering|Leaving", <Whether the notification should occur when the
terminal is entering or leaving the specified location>

  "frequency": { <Maximum frequency of notifications - can be seen as minimum time
between notifications>

    "metric": "Millisecond|Second|Minute|Hour|Day|Week|Month|Year", <The
metric to use for time measurement>

    "units": "Integer" <The number of units of the metric>
  },

  "latitude": "Float", <The latitude of the center point of the location>

  "longitude": "Float", <The longitude of the center point of the location>

  "radius": "Float", <The radius of the circle around the center point, in meters>

  "reference": { <Notification endpoint information>

    "correlator": "String", <A correlator used to identify the notification>

    "endpoint": "URI", <The endpoint address to which the notification should be delivered.
This should be a channel name that begins: "/bayeux/${appInstanceId}">

    "interfaceName": "String" <A descriptive string to identify the type of notification>
  },

  "trackingAccuracy": "Float", <Acceptable error, in meters>
}
```

```
"count": "Integer", <The maximum number of notifications. For no maximum, leave this
blank or specify zero>

"duration": { <The length of time over which notifications should occur. Do not specify to
use the default time>

    "metric": "Millisecond|Second|Minute|Hour|Day|Week|Month|Year", <The
metric to use for time measurement>

    "units": "Integer" <The number of units of the metric>

}
}
```

HTTP Method

PUT

Response

Note: For the actual body of the notification as it is delivered to the designated endpoint, see [Notifications](#).

Header

key= "Location" value = "host:port/rest/terminal_location/notifications"

The URI of the pub-sub server

Error Response

[Service Exception](#)

[Policy Exception](#)

Start Periodic Notification

Starts a notification for a set of terminals at a defined interval

Request

URI

http://host:port/rest/terminal_location/periodic-notification

Request Content-Type

application/json

Body

```
{
  "addresses": [ "URI" ], <Addresses of the terminals to monitor>
  "frequency": { <Maximum frequency of notifications - can be seen as minimum time
between notifications>
    "metric": "Millisecond|Second|Minute|Hour|Day|Week|Month|Year" , <The
metric to use for time measurement>
    "units": "Integer" <The number of units of the metric>
  },
  "reference": { <Notification endpoint information>
    "correlator": "String" , <A correlator used to identify the notification>
    "endpoint": "URI" , <The endpoint address to which the notification should be delivered.
This should be a channel name that begins: "/bayeux/${appInstanceId}">
    "interfaceName": "String" <A descriptive string to identify the type of notification>
  },
  "requestedAccuracy": "Integer" , <Acceptable error, in meters>
  "duration": { <The length of time over which notifications should occur. Do not specify to use
the default time>
    "metric": "Millisecond|Second|Minute|Hour|Day|Week|Month|Year" , <The
metric to use for time measurement>
    "units": "Integer" <The number of units of the metric>
  }
}
```

HTTP Method

PUT

Response

Note: For the actual body of the notification as it is delivered to the designated endpoint, see [Notifications](#).

Header

key= “Location” value = host:port/rest/terminal_location/notifications”

The URI of the pub-sub server

Error Response

[Service Exception](#)

[Policy Exception](#)

End Notification

Stops both types of notification

Request

URI

http://host:port/rest/terminal_location/notification/\${correlator}

\${correlator}: The correlator passed in the reference attribute when the notification was established.

HTTP Method

DELETE

Response

Body

Empty

Error Response

[Service Exception](#)

[Policy Exception](#)

Notifications

The content of notifications delivered to the designated endpoint.

Location End

The notifications have ended for this correlator. Either count or duration has been completed. Not delivered in the case of an error or of the application ending the notification using End Notification.

```
{ "locationEnd": { "correlator": "String" } }
```

The correlator that was passed in the initial set up of the notification.

Location Error

The notification for a terminal or an entire notification is being ended because of an error.

```
{ "locationError": {
  "correlator": "String", <The correlator passed in when the notification was set up>
  "reason": { <The description of the error>
    "messageId": "String", <Fault definition message identifier>
    "text": "String", <Message text, with replacement variables>
    "variables": [ "String" ] <Variables to substitute into text string>
  },
  "address": "URI" <The address of the affected terminal>
} }
```

Location Notification

The notification

```
{ "locationNotification": {
  "correlator": "String", <The correlator passed in when the notification was set up>
  "data": [ { <The location information for a terminal>
    "address": "URI", <The address of the terminal>
    "reportStatus": "Retrieved|NotRetrieved|Error", <Retrieval status for this address. This allows for partial reports to avoid timeouts, etc.>
    "currentLocation": { <Location of the terminal. Only provided if reportStatus = Retrieved>
      "accuracy": "Integer", <Accuracy of location in meters>
```

```
    "latitude": "Float", <Location latitude>
    "longitude": "Float", <Location longitude>
    "timestamp": "Calendar", <Date and time the location was collected in ISO 8601
extended format >
    "altitude": "Float" <Location altitude>
  },
  "errorInformation": { <If reportStatus=Error, this is the error. This provides error
information for a single item out of a group.>
    "messageId": "String", <Fault definition message identifier>
    "text": "String", <Message text, with replacement variables>
    "variables": [ "String" ] <Variables to substitute into text string>
  }
},
  "criteria": "Entering|Leaving" <The criterion that triggered the notification. For
geographical notifications only>
}}
```

Errors

The content of possible error messages.

Service Exception

```
{ "error": {
  "type": "org.csapi.schema.parlayx.common.v2_1.ServiceException"
  "message": "String"
}}
```

Policy Exception

```
{ "error": {
  "type": "org.csapi.schema.parlayx.common.v2_1.PolicyException"
  "message": "String"
}}
```

```
}}
```

Terminal Location

Third Party Call

Operations

The RESTful Third Party Call interfaces allow an application to set up a call, get information on that call, cancel the call request before it is successfully completed, or end a call that has been successfully set up.

In the listings below, **bold** indicates a required attribute, <> a descriptive comment.

Note: An HTML version of this information is at

`http://host:port/rest/third_party_call/index.html` where host and port depend on the Oracle Communications Services Gatekeeper installation.

Make Call

Sets up a call between two parties, the calling party and the called party.

Request

URI

`http://host:port/rest/third_party_call/calls`

Request Content-Type

`application/json`

Third Party Call

Body

```
{
  "calledParty": "URI", <The address of the party being called>
  "callingParty": "URI", <The address of the party making the call>
  "charging": {<This entire attribute is optional>
    "description": "String", <Text to be used for information and billing>
    "amount": "BigDecimal", <The amount to be charged>
    "code": "String", <A charging code, from an existing contractual description>
    "currency": "String" <A currency identifier as defined in ISO 4217 [9]>
  }
}
```

HTTP Method

POST

Response

Response Content-Type

application/json

Body

```
{ "result": "String" }
```

The call identifier, a string used to identify the call for later correlation

Header

key= "Location" value = "host:port/rest/third_party_call/call/\${result}"

\${result}: The call identifier returned in the body

The URI of the pub/sub server, from which call information can be fetched

Error Response

[Service Exception](#)

[Policy Exception](#)

Get Call Information

Get information on a previously established call. If the call has terminated, the information is available for an operator configurable amount of time.

Request

URI

`http://host:port/rest/third_party_call/call/${callIdentifier}`

`${callIdentifier}`: A string, the call identifier returned from the initial setup request

Response Content-Type

application/json

Body

```
{ "result": {
  "callStatus": "CallInitial|CallConnected|CallTerminated", <The current status
of the call. See Table 8-1 for more information>

  "duration": "Integer", <Only when callStatus=CallTerminated. The duration of the
call in seconds>

  "startTime": "Calendar", <Only when callStatus in not CallInitial. The time of the
beginning of the call in ISO 8601 extended format >

  "terminationCause":
"CallingPartyNoAnswer|CalledPartyNoAnswer|CallingPartyBusy|CalledPartyBusy
|CallingPartyNotReachable|CalledPartyNotReachable|CallHangUp|CallAborted"
  <Only when callStatus=CallTerminated. The reason for the termination.>
}}
```

Table 8-1 Call Status

Value	Meaning
CallInitial	The call is being established
CallConnected	The call is active
CallTerminated	The call was terminated

Error Response

[Service Exception](#)

[Policy Exception](#)

Cancel Call Request

Cancels a previously requested call that is not yet active. If the call is established, this operation has no effect.

Request

URI

`http://host:port/rest/third_party_call/cancel-call/${callIdentifier}`

`${callIdentifier}`: A string, the call identifier returned from the initial call setup

HTTP Method

POST

Response

Body

Empty

Error Response

[Service Exception](#)

[Policy Exception](#)

End Call

Ends a previously requested call that is active. (Calls in initial state should be cancelled instead.)

Request

URI

`http://host:port/rest/third_party_call/end-call/${callIdentifier}`

`${callIdentifier}`: A string, the call identifier returned from the initial call setup

HTTP Method

POST

Response

Body

Empty

Error Response

[Service Exception](#)

[Policy Exception](#)

Errors

The content of possible error messages.

Service Exception

```
{ "error": {  
  "type": "org.csapi.schema.parlayx.common.v2_1.ServiceException"  
  "message": "String"  
}}
```

Policy Exception

```
{ "error": {  
  "type": "org.csapi.schema.parlayx.common.v2_1.PolicyException"  
  "message": "String"
```

Third Party Call

Call Notification

Operations

The RESTful Call Notification interfaces allow an application to set up and remove call notifications (in which the application is informed of a particular state - busy, unreachable, etc. - of the call) or to set up and remove call direction notifications (in which the application is queried for information on handling a call that is in a particular state).

In the listings below, **bold** indicates a required attribute, <> a descriptive comment.

Note: An HTML version of this information is at

`http://host:port/rest/call_notification/index.html` where host and port depend on the Oracle Communications Services Gatekeeper installation.

Start Call Notification

Sets up a call notification

Request

URI

`http://host:port/rest/call_notification/call-notification`

Request Content-Type

`application/json`

Body

```
{
  "addresses": [ "String" ], <The addresses to be monitored>
  "reference": {
    "correlator": "String", <A correlator used to identify the notification>
    "endpoint": "URI", <The endpoint address to which the notification should be delivered.
    This should be a channel name that begins: "/bayeux/${appInstanceId}">.
    "interfaceName": "String" <Descriptive only, a string to identify the notification>
  },
  "criteria": [ "Busy|NotReachable|NoAnswer|CalledNumber" ] <Call events for
  which notification is required. If this is not specified, all events trigger notification. See Table 9-1
  for more information>
}
```

Table 9-1 Criteria values

Value	Meaning
Busy	Called party is busy
NotReachable	Called party is not reachable
NoAnswer	Called party does not answer
CalledNumber	A call between two parties is being attempted.

HTTP Method

PUT

Response

Note: For the actual body of the notification as it is delivered to the designated endpoint, see [Notifications](#).

Header

key= "Location" value = "host:port/rest/call_notification/notifications"

The URI of the pub-sub server

Error Response

[Service Exception](#)

[Policy Exception](#)

Stop Call Notification

Stops a previously set up notification

Request

URI

`http://host:port/rest/call_notification/call-notification/${correlator}`

`${correlator}`: The correlator passed in with the initial request

HTTP Method

DELETE

Response

Body

Empty

Error Response

[Service Exception](#)

[Policy Exception](#)

Start Call Direction Notification

Sets up a call direction notification.

Note: Because Call Direction Notifications require a response from the application to Oracle Communications Services Gatekeeper, the descriptions of the notifications include a reply-to private channel on the pub-sub server and a description of the data that must be returned. The application publishes the appropriate information to the pub-sub server channel, to which Oracle Communications Services Gatekeeper is subscribed.

Request

URI

http://host:port/rest/call_notification/call-direction

Request Content-Type

application/json

Body

```
{
  "addresses": [ "String" ], <The addresses to be monitored>
  "reference": {
    "correlator": "String", <A correlator used to identify the notification>
    "endpoint": "URI", <The endpoint address to which the notification should be delivered.
    This should be a channel name that begins: "/bayeux/${appInstanceId}">
    "interfaceName": "String" <Descriptive only, a string to identify the notification>
  },
  "criteria": [ "Busy|NotReachable|NoAnswer|CalledNumber" ] <Call events for
  which notification is required. If this is not specified, all events trigger notification. See Table 9-2
  for more information>
}
```

HTTP Method

PUT

Table 9-2 Criteria values

Value	Meaning
Busy	Called party is busy
NotReachable	Called party is not reachable
NoAnswer	Called party does not answer
CalledNumber	A call between two parties is being attempted.

Response

Note: For the actual body of the notification as it is delivered to the designated endpoint, see [Notifications](#).

Header

key= “Location” value = “host:port/rest/call_notification/notifications”

The URI of the pub-sub server

Error Response

[Service Exception](#)

[Policy Exception](#)

Stop Call Direction Notification

Stops a previously set up call direction notification

Request

URI

http://host:port/rest/call_notification/call-direction/\${correlator}

\${correlator}: The correlator passed in with the initial request

HTTP Method

DELETE

Response

Body

Empty

Error Response

[Service Exception](#)

[Policy Exception](#)

Notifications

The content of notifications delivered to the designated endpoint

Notify Called Number

```
{ "notifyCalledNumber": {  
  "calledParty": "URI", <The address of the called party>  
  "callingParty": "URI", <The address of the calling party>  
  "correlator": "String", <The correlator passed in with the set up request>  
  "callingPartyName": "String" <The name of the calling party>  
}}
```

Notify Busy

```
{ "notifyBusy": {  
  "calledParty": "URI", <The address of the called party>  
  "callingParty": "URI", <The address of the calling party>  
  "correlator": "String", <The correlator passed in with the set up request>  
  "callingPartyName": "String" <The name of the calling party>  
}}
```

Notify No Answer

```
{ "notifyNoAnswer": {  
  "calledParty": "URI", <The address of the called party>  
  "callingParty": "URI", <The address of the calling party>  
  "correlator": "String", <The correlator passed in with the set up request>  
  "callingPartyName": "String" <The name of the calling party>  
}}
```

Notify Not Reachable

```
{ "notifyNotReachable": {  
  "calledParty": "URI", <The address of the called party>
```



```

    "callingParty": "URI", <The address of the calling party>
    "correlator": "String", <The correlator passed in with the set up request>
    "callingPartyName": "String" <The name of the calling party>
  }}

```

Handle Called Number

{**"reply-to":** "\${replyToChannel}", <The endpoint address to which the handle instructions from the application should be delivered. This is a private Bayeux channel set up by Oracle Communications Services Gatekeeper.>

```

    "data": { "handleCalledNumber": {
      "calledParty": "URI", <The address of the called party>
      "callingParty": "URI", <The address of the calling party>
      "correlator": "String", <The correlator passed in with the set up request>
      "callingPartyName": "String" <The name of the calling party>
    }}

```

Handle Called Number Response

Applications respond to the Handle Called Number notification by publishing this data to the `reply-to`: channel provided in the notification.

```

{ "handleCalledNumberResponse": { "result": {
  "actionToPerform": "Route|Continue|EndCall", <What Oracle Communications Services Gatekeeper should do with the call. See Table 9-3 for more information>
  "charging": { <This entire attribute is optional>
    "description": "String", <Text to be used for information and billing>
    "amount": "BigDecimal", <The amount to be charged>
    "code": "String", <A charging code, from an existing contractual description>
    "currency": "String" <A currency identifier as defined in ISO 4217 [9]>
  },
  "routingAddress": "URI" <The address to which the call should be routed>
}}

```

Table 9-3 Handle Response

Value	Meaning
Route	Re-route the call to a specified address
Continue	Handle the call in the normal way
EndCall	Terminate the call

Handle Busy

`{"reply-to": "${replyToChannel}" , <The endpoint address to which the handle instructions from the application should be delivered. This is a private Bayeux channel set up by Oracle Communications Services Gatekeeper.>`

```
  "data": { "handleBusy": {
    "calledParty": "URI" , <The address of the called party>
    "callingParty": "URI" , <The address of the calling party>
    "correlator": "String" , <The correlator passed in with the set up request>
    "callingPartyName": "String" <The name of the calling party>
  } } }
```

HandleBusyResponse

Applications respond to the Handle Busy notification by publishing this data to the `reply-to` channel provided in the notification.

```
{ "handleBusyResponse": { "result": {
  "actionToPerform": "Route|Continue|EndCall" , <What Oracle Communications Services Gatekeeper should do with the call. See Table 9-3 for more information>
  "charging": { <This entire attribute is optional>
    "description": "String" , <Text to be used for information and billing>
    "amount": "BigDecimal" , <The amount to be charged>
    "code": "String" , <A charging code, from an existing contractual description>
    "currency": "String" <A currency identifier as defined in ISO 4217 [9]>
```

```

    },
    "routingAddress": "URI"<The address to which the call should be routed>
  }}}

```

Handle No Answer

{**reply-to**: "\${replyToChannel}" , <The endpoint address to which the handle instructions from the application should be delivered. This is a private Bayeux channel set up by Oracle Communications Services Gatekeeper.>

```

  "data": { "handleNoAnswer": {
    "calledParty": "URI" , <The address of the called party>
    "callingParty": "URI" , <The address of the calling party>
    "correlator": "String" , <The correlator passed in with the set up request>
    "callingPartyName": "String" <The name of the calling party>
  }}}

```

Handle No Answer Response

Applications respond to the Handle No Answer notification by publishing this data to the reply-to: channel provided in the notification.

```

{ "handleNoAnswerResponse": { "result": {
  "actionToPerform": "Route|Continue|EndCall" , <What Oracle Communications Services Gatekeeper should do with the call. See Table 9-3 for more information>
  "charging": { <This entire attribute is optional>
    "description": "String" , <Text to be used for information and billing>
    "amount": "BigDecimal" , <The amount to be charged>
    "code": "String" , <A charging code, from an existing contractual description>
    "currency": "String" <A currency identifier as defined in ISO 4217 [9]>
  },
  "routingAddress": "URI"<The address to which the call should be routed>
}}}

```

Handle Not Reachable

{**reply-to**: "\${replyToChannel}" , <The endpoint address to which the handle instructions from the application should be delivered. This is a private Bayeux channel set up by Oracle Communications Services Gatekeeper.>

```

    "data": { "handleNotReachable": {
      "calledParty": "URI" , <The address of the called party>
      "callingParty": "URI" , <The address of the calling party>
      "correlator": "String" , <The correlator passed in with the set up request>
      "callingPartyName": "String" <The name of the calling party>
    } } }

```

Handle Not Reachable Response

Applications respond to the Handle Not Reachable notification by publishing this data to the `reply-to`: channel provided in the notification.

```

{ "handleNoAnswerResponse": { "result": {
  "actionToPerform": "Route|Continue|EndCall" , <What Oracle Communications Services Gatekeeper should do with the call. See Table 9-3 for more information>
  "charging": { <This entire attribute is optional>
    "description": "String" , <Text to be used for information and billing>
    "amount": "BigDecimal" , <The amount to be charged>
    "code": "String" , <A charging code, from an existing contractual description>
    "currency": "String" <A currency identifier as defined in ISO 4217 [9]>
  },
  "routingAddress": "URI" <The address to which the call should be routed>
} } }

```

Errors

The content of possible error messages.

Service Exception

```
{ "error": {  
  "type": "org.csapi.schema.parlayx.common.v2_1.ServiceException"  
  "message": "String"  
}}
```

Policy Exception

```
{ "error": {  
  "type": "org.csapi.schema.parlayx.common.v2_1.PolicyException"  
  "message": "String"  
}}
```

Call Notification

Presence

Operations

The RESTful Presence interfaces allow an application to act as either of two different parties to a presence interaction: as a presentity or as a watcher. A presentity agrees to have certain data (called *attributes*) such as current activity, available communication means, and contact addresses made available to others while a watcher is a consumer of such information. As a watcher, an application can request to subscribe to all or a subset of a presentity's data, poll for that data, and start and end presence notifications. As a presentity, an application can publish presence data about itself, check to see if any new watchers wish to subscribe to its presence data, authorize those watchers it chooses to authorize, block those it wishes not to have access, and get a list of currently subscribed watchers.

In the listings below, **bold** indicates a required attribute, <> a descriptive comment.

Note: An HTML version of this information is at
`http://host:port/rest/presence/index.html` where host and port depend on the
Oracle Communications Services Gatekeeper installation.

Get Open Subscriptions

Polls for any watchers who want to subscribe to this presentity's data.

Request

URI

http://host:port/rest/presence/subscriptions?status=open

HTTP Method

GET

Response

Response Content-Type

application/json

Body

```
{ "result": [ {
    "application": "String", <A descriptive name for the application that operates on behalf
of the watcher. Information only>

    "attributes": [ "Activity|Place|Privacy|Sphere|Communication|Other" ], <The
attributes this watcher wishes to see. An empty array means all attributes. See Table 10-1 for
more information.>

    "watcher": "URI" <The address of the watcher whose request this is>
  } ] }
```

Table 10-1 Presence Attributes

Value	Meaning
Activity	The presentity's activity (available, busy, lunch, etc.)
Place	The kind of place where the presentity is (home, office, etc.)
Privacy	The amount of privacy that the presentity wants (public, private, etc.)
Sphere	The presentity's current environment (work, home,)
Communication	The presentity's preferred means of communications (email, phone, etc.)
Other	A key-value pair to describe any arbitrary information

Error Response

[Service Exception](#)

[Policy Exception](#)

Get My Watchers

Gets an array of current watchers

Request

URI

`http://host:port/rest/presence/subscriptions?filter=watcher`

HTTP Method

GET

Response

Response Content-Type

application/json

Body

```
{"result": ["URI"]}
```

The addresses of the current watchers

Error Response

[Service Exception](#)

[Policy Exception](#)

Update Subscription Authorization

Used to add watchers who have recently asked for subscriptions or to change permissions for any current watchers.

Request

URI

`http://host:port/rest/presence/subscription/authorization`

Request Content-Type

`application/json`

Body

```
{
  "decisions": [ {<
    "decision": "Boolean", <Whether to allow the watcher>
    "presenceAttribute":
      "Activity|Place|Privacy|Sphere|Communication|Other"
  } ], <The attributes this watcher wishes to see. An empty array means all attributes. See
  Table 10-1 for more information.>
  "watcher": "URI" <The watcher whose request is being evaluated>
}
```

HTTP Method

`PUT`

Response

Body

Empty

Error Response

[Service Exception](#)

[Policy Exception](#)

Subscribe Presence

Requests to subscribe to a presentity's data. No data will be available until the presentity authorizes the watcher using Update Subscription Authorization.

Request

URI

`http://host:port/rest/presence/subscription`

Request Content-Type

`application/json`

Body

```
{
  "application": "String", <A descriptive name for the application whose data the watcher
wishes to access. Informational only>
  "presentity": "URI", <The address of the presentity or group whose data is being
requested>
  "reference": {
    "correlator": "String", <A correlator identifying this request>
    "endpoint": "URI", <The endpoint address to which the Notfiy Subscription notification
is delivered>
    "interfaceName": "String" <Descriptive only>
  },
  "attributes": ["Activity|Place|Privacy|Sphere|Communication|Other"] <The
attributes this watcher wishes to see. An empty array means all attributes. See Table 10-1 for
more information.>
}
```

HTTP Method

`PUT`

Response

Body

Empty

Error Response

[Service Exception](#)

[Policy Exception](#)

Block Subscription

Allows a presentity to block a watcher's access to the presentity's data. The watcher is notified with a Subscription Ended notification.

Request

URI

`http://host:port/rest/presence/subscription/${watcher}`

`${watcher}`: The URI of the watcher to block

HTTP Method

DELETE

Response

Body

Empty

Error Response

[Service Exception](#)

[Policy Exception](#)

Start Presence Notification

Begin delivering presence data to the endpoint defined in the reference attribute. This operation is only functional if the presentity has approved the watcher.

Request

URI

`http://host:port/rest/presence/notification`

Request Content-Type

application/json

Body

```
{
  "checkImmediate": "Boolean", <Whether to check immediately after establishing the
notification>

  "frequency": { <Maximum frequency of notifications (can also be considered minimum
time between notifications). In the case of a group subscription the service must make sure this
frequency is not violated by notifications for various members of the group, especially in
combination with checkImmediate.>

    "metric": "Millisecond|Second|Minute|Hour|Day|Week|Month|Year", <The
metric in which to measure the frequency>

    "units": "Integer" <The number of units of the metric>
  },
  "presentity": "URI", <The presentity or group whose data is being accessed>
  "reference": {
    "correlator": "String", <A correlator identifying this notification>
    "endpoint": "URI", <The endpoint address to which the notification is to be delivered>
    "interfaceName": "String" <Descriptive only, the type of notification>
  },
  "attributes": [ "Activity|Place|Privacy|Sphere|Communication|Other" ], <The
attributes this watcher wishes to see. An empty array means all attributes. See Table 10-1 for
more information.>

  "count": "Integer", <The maximum number of notifications. Zero or unspecified means
no maximum>

  "duration": { <The length of time over which notifications occur. For service default, do
not specify>

    "metric": "Millisecond|Second|Minute|Hour|Day|Week|Month|Year", <The
metric in which to measure the duration>

    "units": "Integer" <The number of units of the metric>
  }
}
```

```
}
```

HTTP Method

PUT

Response

Response Content-Type

application/json

Body

```
{"result": ["URI"]}
```

The presentities to whose attributes the watcher did not successfully subscribe. Empty if there were no issues.

Header

`http://host:port/rest/presence/notifications`

The URI of the pub/sub server.

Error Response

[Service Exception](#)

[Policy Exception](#)

End Presence Notification

Stops a presence notification

Request

URI

`http://host:port/rest/presence/notification/${correlator}`

`${correlator}:String`

The correlator passed in when the notification was started.

HTTP Method

DELETE

Response

Body

Empty

Error Response

[Service Exception](#)

[Policy Exception](#)

Get User Presence

Gets the aggregated presence data of a presentity to whose data the watcher has previously successfully subscribed. Only the attributes that the watcher is authorized to see are returned.

Request

URI

`http://host:port/rest/presence/data?query=${query}`

`${query}:`

```
{  
  "presentity": "URI", <The address of the presentity being requested>  
  "attributes": [ "Activity|Place|Privacy|Sphere|Communication|Other" ] <The  
attributes being requested. An empty array means all attributes. See Table 10-1 for more  
information.>  
}
```

HTTP Method

GET

Response

Response Content-Type

application/json

Body

```
{ "result": [ {
  "lastChange": "Calendar", <The time and date the data last changed in ISO 8601 extended
format >

  "typeAndValue": { <The presence information>

    "unionElement":
"Activity|Place|Privacy|Sphere|Communication|Other", <See Table 10-1.
Determines what data is presented.>

    "activity":
"ActivityNone|Available|Busy|DoNotDisturb|OnThePhone|Steering|Meeting|Away
|Meal|PermanentAbsence|Holiday|Performance|InTransit|Travel|Sleeping|Activ
ityOther", <What the presentity is doing. ActivityNone indicates the value has not been set.
ActivityOther refers to any non-listed activity type.>

    "communication": { "means": [ { <Connection information for presentity's preferred
forms of communication>

      "contact": "URI", <The contact address for this particular means>

      "priority": "Float", <The priority of this particular means. Between 0 and 1, with 1
indicating most preferred means>

      "type": "Phone|Chat|Sms|Video|Web|Email|Mms|MeansOther" <The type of
contact client for this particular means. Video refers to a video phone. MeansOther refers to any
other client type>

    } } ],

    "other": { <An arbitrary name/value pair for the Other element of Table 10-1>

      "name": "String",

      "value": "String"

    },

    "place":
"PlaceNone|Home|Office|PublicTransport|Street|Outdoors|PublicPlace|Hotel|T
heatre|Restaurant|School|Industrial|Quiet|Noisy|Aircraft|Ship|Bus|Station|
```


Mall|Airport|Train|PlaceOther", <The type of location for the presentity. PlaceNone means no value has been set. PlaceOther refers to any other type of place.>

"privacy":

"PrivacyNone|PrivacyPublic|PrivacyPrivate|PrivacyQuiet|PrivacyOther", <The level of privacy in the presentity's current environment. PrivacyNone means the value has not been set. PrivacyOther refers to any other privacy level>

"sphere": "SphereNone|SphereWork|SphereHome|SphereOther" <The sphere within which the presentity is currently acting. SphereNone means the value has not been set. SphereOther refers to any other sphere.

> }, <

"note": "String" <An explanatory note>

}]}

Error Response

[Service Exception](#)

[Policy Exception](#)

Publish

Allows the presentity to publish presence information

Request

URI

http://host:port/rest/presence/data

Request Content-Type

application/json

Body

```
{ "presence": [ {
```

"lastChange": "Calendar" , <The time and date the data last changed in ISO 8601 extended format >

"typeAndValue": { <The presence information>

"unionElement":

"Activity|Place|Privacy|Sphere|Communication|Other", <See [Table 10-1](#).

Determines what data is presented.>

"activity":

"ActivityNone|Available|Busy|DoNotDisturb|OnThePhone|Steering|Meeting|Away|Meal|PermanentAbsence|Holiday|Performance|InTransit|Travel|Sleeping|ActivityOther", <What the presentity is doing. ActivityNone indicates the value has not been set. ActivityOther refers to any non-listed activity type.>

"communication": { "means": [<Connection information for presentity's preferred forms of communication>

"contact": "URI", <The contact address for this particular means>

"priority": "Float", <The priority of this particular means. Between 0 and 1, with 1 indicating most preferred means>

"type": "Phone|Chat|Sms|Video|Web|Email|Mms|MeansOther" <The type of contact client for this particular means. Video refers to a video phone. MeansOther refers to any other client type>

}},

"other": { <An arbitrary name/value pair for the Other element of [Table 10-1](#)>

"name": "String",

"value": "String"

},

"place":

"PlaceNone|Home|Office|PublicTransport|Street|Outdoors|PublicPlace|Hotel|Theatre|Restaurant|School|Industrial|Quiet|Noisy|Aircraft|Ship|Bus|Station|Mall|Airport|Train|PlaceOther", <The type of location for the presentity. PlaceNone means no value has been set. PlaceOther refers to any other type of place.>

"privacy":

"PrivacyNone|PrivacyPublic|PrivacyPrivate|PrivacyQuiet|PrivacyOther", <The level of privacy in the presentity's current environment. PrivacyNone means the value has not been set. PrivacyOther refers to any other privacy level>

"sphere": "SphereNone|SphereWork|SphereHome|SphereOther" <The sphere within which the presentity is currently acting. SphereNone means the value has not been set. SphereOther refers to any other sphere.

> },<

```
"note": "String"<An explanatory note>
}}}
```

HTTP Method

PUT

Response

Body

Empty

Error Response

[Service Exception](#)

[Policy Exception](#)

Notifications

The content of notifications delivered to the designated endpoint.

Status Changed

```
{ "statusChanged": {
  "changedAttributes": [ {
    "lastChange": "Calendar" , <The time and date the data last changed in ISO 8601
extended format >
    "typeAndValue": { <The presence information>
      "unionElement":
"Activity|Place|Privacy|Sphere|Communication|Other" , <See Table 10-1.
Determines what data is presented.>
      "activity":
"ActivityNone|Available|Busy|DoNotDisturb|OnThePhone|Steering|Meeting|Away
|Meal|PermanentAbsence|Holiday|Performance|InTransit|Travel|Sleeping|Activ
ityOther" , <What the presentity is doing. ActivityNone indicates the value has not been set.
ActivityOther refers to any non-listed activity type.>
```

```

    "communication": { "means": [ {<Connection information for presentity's preferred
forms of communication>

    "contact": "URI", <The contact address for this particular means>

    "priority": "Float", <The priority of this particular means. Between 0 and 1, with 1
indicating most preferred means>

    "type": "Phone|Chat|Sms|Video|Web|Email|Mms|MeansOther" <The type of
contact client for this particular means. Video refers to a video phone. MeansOther refers to any
other client type>

    } } },

    "other": { <An arbitrary name/value pair for the Other element of Table 10-1>

        "name": "String",

        "value": "String"

    },

    "place":
    "PlaceNone|Home|Office|PublicTransport|Street|Outdoors|PublicPlace|Hotel|T
heatre|Restaurant|School|Industrial|Quiet|Noisy|Aircraft|Ship|Bus|Station|
Mall|Airport|Train|PlaceOther", <The type of location for the presentity. PlaceNone
means no value has been set. PlaceOther refers to any other type of place.>

    "privacy":
    "PrivacyNone|PrivacyPublic|PrivacyPrivate|PrivacyQuiet|PrivacyOther", <The
level of privacy in the presentity's current environment. PrivacyNone means the value has not
been set. PrivacyOther refers to any other privacy level>

    "sphere": "SphereNone|SphereWork|SphereHome|SphereOther" <The sphere within
which the presentity is currently acting. SphereNone means the value has not been set.
SphereOther refers to any other sphere.

> } , <

    "note": "String" <An explanatory note>

} ]

    "correlator": "String", <The correlator passed in with the notification set up>

    "presentity": "URI" <The presentity whose data this is>

} }

```

Notify Subscription

```
{ "notifySubscription": {
  "presentity": "URI", <The address of the presentity who has or has not authorized the
  watcher to whom this notification is sent>
  "decisions": [{
    "decision": "Boolean", <Whether the subscription is authorized>
    "presenceAttribute":
    "Activity|Place|Privacy|Sphere|Communication|Other"
  ] <What attributes are authorized. See Table 10-1>
}}
```

Status End

```
{ "statusEnd": { "correlator": "String" } }
```

The notification indicated by the correlator has ended, because either the duration or count was completed. This is not delivered in the case of an error or the use of End Presence Notification.

Subscription Ended

```
{ "subscriptionEnded": {
  "presentity": "URI", <The presentity in question>
  "reason": "String" <Timeout or blocked>
}}
```

The subscription has been terminated, either blocked by the presentity or because of a timeout or connection failure.

Errors

The content of possible error messages.

Service Exception

```
{ "error": {
  "type": "org.csapi.schema.parlayx.common.v2_1.ServiceException"
  "message": "String"
}
```

Presence

```
}}
```

Policy Exception

```
{ "error": {  
  "type": "org.csapi.schema.parlayx.common.v2_1.PolicyException"  
  "message": "String"  
}}
```

Payment

Operations

The RESTful Payment interface allows an application to charge an amount to an end-user's account using Diameter, refund amounts to that account, and split charge amounts among multiple end-users. An application can also reserve amounts, reserve additional amounts, charge against the reservation or release the reservation.

In the listings below, **bold** indicates a required attribute, <> a descriptive comment.

Note: An HTML version of this information is at
`http://host:port/rest/payment/index.html` where host and port depend on the Oracle Communications Services Gatekeeper installation.

Charge Amount

Charges an amount directly to an end-user's account using Diameter

Request

URI

`http://host:port/rest/payment/charge-amount`

Request Content-Type

`application/json`

Payment

Body

```
{
  "charge": {
    "description": "String", <Text to be used for information and billing>
    "amount": "BigDecimal", <The amount to be charged>
    "code": "String", <A charging code, from an existing contractual description>
    "currency": "String" <A currency identifier as defined in ISO 4217 [9]>
  },
  "endUserIdentifier": "URI", <The address of the end-user to be charged>
  "referenceCode": "String" <A unique identifier in case of disputes>
}
```

HTTP Method

POST

Response

Body

Empty

Error Response

[Service Exception](#)

[Policy Exception](#)

Refund Amount

Refunds an amount directly to an end-user's account using Diameter

Request

URI

`http://host:port/rest/payment/refund-amount`

Request Content-Type

application/json

Body

```
{
  "charge": {
    "description": "String", <Text to be used for information and billing>
    "amount": "BigDecimal", <The amount to be refunded>
    "code": "String", <A charging code, from an existing contractual description>
    "currency": "String" <A currency identifier as defined in ISO 4217 [9]>
  },
  "endUserIdentifier": "URI", <The address of the end-user to be refunded to>
  "referenceCode": "String" <A unique identifier in case of disputes>
}
```

HTTP Method

POST

Response

Body

Empty

Error Response

[Service Exception](#)

[Policy Exception](#)

Charge Split Amount

Charges an amount directly to multiple end users concurrently (for example, for charging multiple participants in a conference)

Request

URI

`http://host:port/rest/payment/charge-split-amount`

Request Content-Type

`application/json`

Body

```
{
  "charge": {
    "description": "String", <Text to be used for information and billing>
    "amount": "BigDecimal", <The amount to be refunded>
    "code": "String", <A charging code, from an existing contractual description>
    "currency": "String" <A currency identifier as defined in ISO 4217 [9]>
  },
  "referenceCode": "String", <A unique identifier in case of disputes>
  "splitInfo": [ {<How the charge should be split>
    "endUserIdentifier": "URI", <The address of the end-user>
    "percent": "Integer" <The percentage this end-user should be charged>
  } ]
}
```

HTTP Method

`POST`

Response

Body

Empty

Error Response

[Service Exception](#)

Policy Exception

Reserve Amount

Reserves an amount for an account indicated by the end-user identifier.

Request

URI

`http://host:port/rest/payment/reserve-amount`

Request Content-Type

`application/json`

Body

```
{
  "charge": {
    "description": "String", <Text to be used for information and billing>
    "amount": "BigDecimal", <The amount to be reserved>
    "code": "String", <A charging code, from an existing contractual description>
    "currency": "String" <A currency identifier as defined in ISO 4217 [9]>
  },
  "endUserIdentifier": "URI", <The address of the end-user against whose account the
reservation is made>
}
```

HTTP Method

`POST`

Response

Response Content-Type

`application/json`

Body

```
{"result": "String"}
```

An identifier for the reservation

Error Response

[Service Exception](#)

[Policy Exception](#)

Reserve Additional Amount

Reserves an additional amount against an end-user account

Request

URI

`http://host:port/rest/payment/reserve-additional-amount`

Request Content-Type

`application/json`

Body

```
{
  "charge": {
    "description": "String", <Text to be used for information and billing>
    "amount": "BigDecimal", <The amount to be reserved>
    "code": "String", <A charging code, from an existing contractual description>
    "currency": "String" <A currency identifier as defined in ISO 4217 [9]>
  },
  "reservationIdentifier": "String" <The result value returned from the initial reservation request>
}
```

HTTP Method

POST

Response

Body

Empty

Error Response

[Service Exception](#)

[Policy Exception](#)

Charge Reservation

Charges a previously reserved amount against an end-user account

Request

URI

`http://host:port/rest/payment/charge-reservation`

Request Content-Type

`application/json`

Body

```
{
  "charge": {
    "description": "String", <Text to be used for information and billing>
    "amount": "BigDecimal", <The amount to be reserved>
    "code": "String", <A charging code, from an existing contractual description>
    "currency": "String" <A currency identifier as defined in ISO 4217 [9]>
  },
  "referenceCode": "String", <A unique identifier in case of disputes>
  "reservationIdentifier": "String" <The result value returned from the initial
reservation request>
}
```

Payment

HTTP Method

POST

Response

Body

Empty

Error Response

[Service Exception](#)

[Policy Exception](#)

Release Reservation

Returns funds left in a reservation to the account against which this reservation was made

Request

URI

`http://host:port.rest/payment/release-reservation`

Request Content-Type

application/json

Body

```
{"reservationIdentifier": "String"}
```

The result value returned from the initial reservation request

HTTP Method

POST

Response

Body

Empty

Error Response

[Service Exception](#)

[Policy Exception](#)

Errors

The content of possible error messages.

Service Exception

```
{ "error": {  
  "type": "org.csapi.schema.parlayx.common.v3_1.ServiceException"  
  "message": "String"  
}}
```

Policy Exception

```
{ "error": {  
  "type": "org.csapi.schema.parlayx.common.v3_1.PolicyException"  
  "message": "String"  
}}
```

Payment

Subscriber Profile

Operations

The RESTful Subscriber Profile interfaces allow an application to query an operator's database for individual subscriber profile attributes (such as a user's terminal type) or entire subscriber profiles.

In the listings below, **bold** indicates a required attribute, <> a descriptive comment.

Note: An HTML version of this information is at http://host:port/rest/subscriber_profile/index.html where host and port depend on the Oracle Communications Services Gatekeeper installation.

Get

Gets specific subscriber profile properties. The properties that can be accessed are defined in the service provider and application SLAs associated with the application.

Request

URI

```
http://host:port/rest/subscriber_profile/profile?query=${query}
```

```
${query}:
```

```
{
```

```
  "address": "URI", <The address associated with the subscriber whose data is being
accessed. Supported schemes include tel, id, imsi, and ipv4.>
```

```
"pathNames": [ "String" ]<The requested subscriber profile properties. These are expressed as a relative UNIX path, such as serviceName/accessControlId/accessControlId>
}
```

HTTP Method

GET

Response

Response Content-Type

application/json

Body

```
{ "properties": [ {<An array of the requested properties, as key value pairs>
  "pathName": "String",<The pathname of the requested property>
  "propertyValue": "String"<The value associated with the requested property>
} ] }
```

Error Response

[Service Exception](#)

[Policy Exception](#)

Get Profile

Gets an entire subscriber profile.

Request

URI

```
http://host:port/rest/subscriber_profile/profile?id=${id}
${id}:
{
```

```
  "profileID": "String",<The ID of the profile,which acts as a set of filters limiting the attributes that can be accessed based on the SLAs associated with the application. May be ignored if Oracle Communications Services Gatekeeper connects to the network using certain protocols.>
```

```

    "subscriberID": "String"<An ID that uniquely identifies the subscriber whose profile is
being accessed>
}

```

HTTP Method

GET

Response

Response Content-Type

application/json

Body

```

{"result": [ {<An array of properties from the requested profile>
    "pathName": "String", <The pathname of the property>
    "propertyValue": "String"<The value associated with that property>
  } ]}

```

Error Response

[Service Exception](#)

[Policy Exception](#)

Errors

The content of possible error messages.

Service Exception

```

{ "error": {
    "type": "com.bea.wlcp.wlng.schema.ews.common.ServiceException"
    "message": "String"
  } }

```

Policy Exception

```
{ "error": {  
  "type": "com.bea.wlcp.wlng.schema.ews.common.PolicyException"  
  "message": "String"  
}}
```

WAP Push

Operation

The RESTful WAP Push interface allows an application to send a WAP Push message. The content of the message is coded as a PAP message. The message payload must adhere to the following:

- WAP Service Indication Specification, as specified in Service Indication Version 31-July-2001, Wireless Application Protocol WAP-167-ServiceInd-20010731-a.
- WAP Service Loading Specification, as specified in Service Loading Version 31-Jul-2001, Wireless Application Protocol WAP-168-ServiceLoad-20010731-a.
- WAP Cache Operation Specification, as specified in Cache Operation Version 31-Jul-2001, Wireless Application Protocol WAP-175-CacheOp-20010731-a.

See <http://www.openmobilealliance.org/tech/affiliates/wap/wapindex.html> for links to the specifications.

Note: The actual message is sent as an HTTP attachment. See [Attachments](#) for more information.

In the listing below, **bold** indicates a required attribute, <> a descriptive comment.

Note: An HTML version of this information is at http://host:port/rest/push_message/index.html where host and port depend on the Oracle Communications Services Gatekeeper installation.

Send Push Message

Sends a WAP Push message.

Request

URI

`http://host:port/rest/push_message/messages`

Request Content-Type

`multipart/form-data`

Message Part Name

`messagePart`

Message Part Content-Type

`application/json`

Message Part Content

```
{  
  "destinationAddresses": [ "String" ], <An array of terminal addresses. The addresses  
  should be formatted according to the Push Proxy Gateway Service Specification  
  (WAP-249-PPGService-20010713-a)>  
  "pushId": "String", <A unique id provided by the application. Supported types are PLMN  
  and USER>  
  "replaceMethod": "all|pending-only", <Defines how to replace a previously sent  
  message. Used in conjunction with the replacePushId parameter. Ignored if replacePushId  
  is NULL.>  
  "requesterID": "String", <The application ID as given by the operator>  
  "serviceCode": "String", <A code for charging purposes>  
  "additionalProperties": [ { <Name-value pairs. Used to add additional properties.  
  Supported types include pap.priority, pap.delivery-method, pap.network, pap.network-required,  
  pap.bearer, pap.bearer-required.>  
    "name": "String", <The property's name>
```

```

    "value": "String"<The value associated with the property>
  },
  "deliverAfterTimestamp": "Calendar", <Specifies the date and time in ISO 8601
  extended format after which the content should be delivered to the wireless device. If the network
  node does not support this parameter, the message is be rejected.>
  "deliverBeforeTimestamp": "Calendar", <Defines the date and time in ISO 8601
  extended format by which the content must be delivered to the end-user terminal .If the network
  node does not support this parameter, the message is rejected.>
  "progressNotesRequested": "Boolean", <If true, the application wants to receive
  progress notes at the address specified by result1NotificationEndpoint>
  "replacePushId": "String", <The PushId of the message that is to be replaced. If this is
  set to NULL, the message is treated as a new message. If this is set to an ID, message replacement
  will occur for all currently pending messages. Already delivered messages cannot be cancelled,
  and therefore cannot be replaced>
  "resultNotificationEndpoint": "URI", <The endpoint for notifications. This should be
  a channel name that begins: "/bayeux/${appInstanceId}" . If this is set, result notifications are
  requested. If the application does not want notifications, this must be NULL>
  "sourceReference": "String"<The name of the service provider>
}

```

HTTP Method

POST

Response

Response Content-Type

application/json

Body

```

{ "result": {
  "pushId": "String", <A unique id provided by the application>
  "result": {

```

"code": "String", <An outcome code generated by the network node. See [Table 13-1](#) for more information>

"description": "String" <A textual description>

},

"additionalProperties": [{

"name": "String", <One of a set of supported additional properties:
pap.stage|pap.note|pap.time>

"value": "String" <The value for the additional property>

}],

"replyTime": "Calendar", <The date and time in ISO 8601 extended format >

"senderAddress": "String", <The sender's address>

"senderName": "String" <The sender's name>

}}

Table 13-1 Outcome codes

Code	Description
1000	OK
1001	Accepted for processing
2000	Bad request
2001	Forbidden
2002	Address error
2003	Address not found
2004	Push ID not found
2005	Capabilities mismatch
2006	Required capabilities not supported
2007	Duplicate push ID

Table 13-1 Outcome codes

Code	Description
2008	Cancellation not possible
3000	Internal server error
3001	Not implemented
3002	Version not supported
3003	Not possible
3004	Capability matching not possible
3005	Multiple addresses not supported
3006	Transformation failure
3007	Specified delivery method not possible
3008	Capabilities not available
3009	Required network not available
3010	Required bearer not available
3011	Replacement not supported
4000	Service failure
4001	Service unavailable

Error Response

[Service Exception](#)

[Policy Exception](#)

Notifications

The content of notifications delivered to the designated endpoint.

Result Notification Message

```
{ "resultNotificationMessage": {
  "address": "String", <The terminal address for this message>
  "code": "String", <Final status of the message>
  "messageState":
    "rejected|pending|delivered|undeliverable|expired|aborted|timeout|cancelled|unknown", <State of the message>
  "pushId": "String", <Defined in the initial request. Used for correlation>
  "additionalProperties": [ { <Name-value pairs. Used to add additional properties.
    Supported types include pap.priority, pap.delivery-method, pap.network, pap.network-required,
    pap.bearer, pap.bearer-required. Dependent on network node.>
    "name": "String", <The property's name>
    "value": "String" <The value associated with the property>
  } ],
  "description": "String", <A description of the notification, provided by the network.
    May not be provided.>
  "eventTime": "Calendar", <The date and time the message reached the destination
    terminal in ISO 8601 extended format >
  "receivedTime": "Calendar", <The date and time the message was received at the network
    node in ISO 8601 extended format >
  "senderAddress": "String", <Address of the network node. May not be present>
  "senderName": "String" <Name of the network node. May not be present>
}}
```

Errors

The content of possible error messages.

Service Exception

```
{ "error": {
  "type": "com.bea.wlcp.wlng.schema.ews.common.ServiceException"
  "message": "String"
}
```

```
}}
```

Policy Exception

```
{ "error": {  
  "type": " com.bea.wlcp.wlng.schema.ews.common.PolicyException"  
  "message": "String"  
}}
```

