Oracle® Communications Services Gatekeeper

RESTful Application Developer's Guide

Release 5.1

E37533-01

June 2013

ORACLE®

Oracle Communications Services Gatekeeper RESTful Application Developer's Guide, Release 5.1

E37533-01

Contents

12 Payment

13 Audio Call

14 Quality of Service

15 Presence

16 Device Capabilities

17 Binary Short Messaging

18 Session Manager

19 Subscriber Profile

20 WAP Push

Preface

This document describes the RESTful Web Services offered by Oracle Communications Services Gatekeeper. It includes a high-level overview of the interfaces and operations.

Audience

This book is intended for software developers who wish to integrate functionality provided by telecom networks into their programs by using the RESTful Web Services in Services Gatekeeper.

Documentation Accessibility

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

Access to Oracle Support

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

Related Documents

For more information, see the following documents in the Oracle Communications Services Gatekeeper set:

- *Oracle Communicatioins Services Gatekeeper Accounts and SLAs Guide*
- *Oracle Communicatioins Services Gatekeeper Alarms Handling Guide*
- *Oracle Communicatioins Services Gatekeeper Application Developer's Guide*
- *Oracle Communicatioins Services Gatekeeper Communication Service Guide*
- *Oracle Communicatioins Services Gatekeeper Concepts Guide*
- *Oracle Communicatioins Services Gatekeeper Installation Guide*
- *Oracle Communications Services Gatekeeper icensing Guide*
- *Oracle Communicatioins Services Gatekeeper Partner Relationship Management Guide*
- *Oracle Communicatioins Services Gatekeeper Platform Development Studio Developer's Guide*

- *Oracle Communicatioins Services Gatekeeper Platform Test Environment* Guide
- *Oracle Communicatioins Services Gatekeeper SDK User's Guide*
- *Oracle Communicatioins Services Gatekeeper Statement of Compliance*
- *Oracle Communicatioins Services Gatekeeper System Administrator's Guide*
- *Oracle Communicatioins Services Gatekeeper System Backup and Restore Guide*

1

Overview of How Applications Interact with Services Gatekeeper

This chapter presents an overview of Oracle Communications Services Gatekeeper functionality, and the ways that application developers can use this functionality to interact with Services Gatekeeper.

Basic Concepts

This section provides an overview of the basic concepts you need to know in order to create applications that successfully interact with Services Gatekeeper.

Communication Services

All application service request data flows through Services Gatekeeper using communication services. A communication service consists of a service type (such as Multimedia Messaging and Terminal Location and so on), an application-facing interface (also called a "north" interface), and a network-facing interface (also called a "south" interface).

Services Gatekeeper supplies facades for traditional SOAP Web Services interfaces, RESTful interfaces, and, in some cases, native telephony interfaces. This document focuses on the RESTful interfaces in Services Gatekeeper.

For more information on communication services, see *Oracle Communications Services Gatekeeper Communication Service Guide*.

Traffic Types

In some Services Gatekeeper communication services, request traffic can travel in two direction: from the application to the underlying network and from the underlying network to the application. In other communication services, traffic flows in one direction only.

Application-initiated Traffic

In application-initiated traffic, the application sends a request to Services Gatekeeper, the request is processed, and a response is returned synchronously.

For example, an application uses the Third Party Call interface to set up a call. The initial operation, **Make Call**, is sent to 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 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 returning to the application.

For example, an application might send an SMS to a mobile phone that the user has turned off. The network will not deliver the message until the user turns the phone back on. 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.

The period of time involved in delivering the message could be hours or extend to days. As a result, 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 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, 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 subscription with Services Gatekeeper or by polling Services Gatekeeper directly to retrieve them. (Bayeux is the protocol that Services Gatekeeper requires applications to use to communicate with the server in association with RESTful interfaces. See "Bayeux Protocol".) Such notifications are kept in Services Gatekeeper for retrieval for only limited amounts of time.

Management Structures

Services Gatekeeper uses a hierarchical system of accounts and service level agreements to regulate the contractual agreements between the telecom operator and the application service provider. For more information on management structures, see *Oracle Communications Services Gatekeeper Accounts and SLAs Guide*.

Included Interfaces

RESTful Web Services consist of interfaces that provide applications with the operations they use to interact with Services Gatekeeper. The following interfaces are included in Services Gatekeeper:

- Third Party Call

  Applications use the Third Party Call interface 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

  Applications use the Call Notification interface to set up and remove call notifications. Such notifications inform the application about the state of a call

(busy, unreachable and so on) 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

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

- Multimedia Messaging

  Applications use the MMS interface to send an MMS and to fetch information on MMSs for the application that have been received and stored on 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

  Applications use the Terminal Location interface 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.

- Audio Call

  Applications use the Audio Call interface to get an audio file from an independent location and send it to a terminal.

- Presence

  Applications use the Presence interface 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. 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.

  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.

- Payment

  Applications use the Payment interface 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

  Applications use the WAP Push interface to send a WAP Push message and to receive status notifications about that message.

- Subscriber Profile

  Applications use the Subscriber Profile interface to query an operator's database for the attributes of an individual subscriber profile (for example, the terminal type), or for entire subscriber profiles.

- Session Manager

  Applications use the Session Manager interface to establish a session with the Services Gatekeeper operator. Operators can choose whether to use Services Gatekeeper in session-based mode.

Application Testing Workflow

You can develop applications and test them using the Application Test Environment (ATE).

This tool hosts a set of virtual communication services that correspond to many of the communication services provided by Services Gatekeeper. For example, you can test tasks such as opening sessions, sending and receiving messages, and examining delivery reports through a set of virtual communication services without having to connect to the network operator´s Services Gatekeeper installation.

For more information on using the Application Test Environment, see *Oracle Communications Services Gatekeeper SDK User's Guide*.

2

Interacting with the RESTful Facade

This chapter provides an overview of how Oracle Communications Services Gatekeeper's RESTful Web Services work.

It describes the ways in which applications use standard HTTP methods in association with the RESTful Web Services facade to interact with Services Gatekeeper. Applications add specific information to the request headers, and, in some cases, they send their message payload as an attachment.

General Format of an Operation

The following basic elements are present in the requests that an application makes to the RESTful facade and/or the responses it receives from the facade:

- Request-URI and HTTP Methods in requests and Status-Line in responses
- Headers
- Message Body
- Attachments: Sometimes messages (for example, multimedia messaging) contain attachments. Special headers are provided to specify the details when a message has multiple parts. See "Headers for Multipart Messages with Attachments".

Request-URI and HTTP Methods

Applications use one of four methods, "GET", "POST", "PUT", or "DELETE", to request a required action to be performed on an abstract or physical resource, a specific Uniform Resource Identifier (URI). The Request-URI, the abstract or physical resource which an HTTP method acts upon or uses is therefore the most important part of any request that an application makes to the RESTful facade.

Here is the GET method used to query for the status of a terminal:

```
GET /rest/terminal_status/status?query="%7B%22address%22%3A%22tel%3A123%22%7D"
HTTP/1.1
```

where, the string `%7B%22address%22%3A%22telA123%22%7D` is the address of the terminal (or the `{"address":"tel:123"}` JavaScript Object Notation (JSON) object).

General Format of a Request-URI

A fully qualified Request-URI is made up of a sequence of sections, concatenated as:

`https://`*`host`*`:`*`port`*`/`*`rest_facade_context_root`*`/`*`URI`*`/`*`path_info_param`*`/`*`query-string`*

where,

- *host*:*port*: The hostname and port of your Services Gatekeeper installation; for example, *127.0.0.1* and *8001*.
- *rest_facade_context_root*: The location of the set of resources that the particular interface uses; for example, `rest`, in the above section.

  In Services Gatekeeper, the *rest_facade_context_root* entry is always **rest**.
- URI: The location of a specific kind of functionality within the interface; for example `terminal_status`, as in the above section.
- *path_info_param*: An identifier of a specific resource, for example, `calls`. This is seen in the Make Call request, `POST /rest/third_party_call/calls HTTP/1.1`. See Example 2–4. The path-info-param entry does not occur in all URIs.
- *query_string*: A set of name-value pairs that describes what is being requested; for example, `status?query="%7B%22address%22%3A%22tel%3A123%22%7D`, in the above section. The query-string entry is not seen in every URI.

POST

The POST method accesses a resource factory to create a new resource that does not yet have a URI. Multiple requests to a resource factory can create multiple new resources.

The following statement will set up a call between two parties:

```
POST /rest/third_party_call/calls HTTP/1.1
```

For the POST method:

- The URI in the request represents the factory resource accessed to create a new resource. In the above example, (**/rest/third_party_call/calls**) is the factory resource accessed to create a resource.
- The request body contains the information required to create the resource. See Example 2–4.
- If the resource is created, the response body will contain the identifier for the new resource. See Example 2–5. If the operation fails, the response body will contain the error response.

PUT

The PUT method creates a resource that has a pre-determined URI. This method can be used to update a resource (or to start a stateful process). For example, an application uses the following statement to start notifications on a specific terminal:

```
PUT /rest/terminal_location/periodic_notification HTTP/1.1
```

For the PUT method:

- The URI in the request represents the resource to update or to start a stateful process. In the example, (**/rest/terminal_location/periodic_notification**) represents the resource accessed to start periodic notifications on a terminal's location.
- The request body for this operation will contain the required information. (The JSON object will contain for example, information on the terminal, the criteria to monitor, the frequency and duration for the monitoring, where to place the notification, and the correlator to identify the session.

- The Location header in the response will specify the location that contains the resulting notifications. For example, the application may see the following header for the above PUT request:

  ```
  Location://10.182.100.245:9001/rest/terminal/notifications
  ```

  In order to complete the operation, the application must access the specified location and use the correlator to retrieve the notifications.

- If the operation fails, the response body will contain the error response.

GET

The GET method retrieves the state of a specific resource that has been previously set up. The specific resource is identified in the query string, as shown above. An application attempts to retrieve the status of a terminal whose address is "tel:123" with the following:

```
GET /rest/terminal_status/status?query=%7B%22address%22%3A%22tel%3A456%22%7D
HTTP/1.1
```

For the GET method:

- The URI in the request represents the query string that uniquely identifies the resource whose status the application wishes to retrieve. In the example, the value for (**status?query=**) is the unique address of the terminal. (It is the address of the terminal, `{"address":"tel:456"}` in JSON representation).
- The request body for this operation will be empty.
- The response will provide information on the state of the resource. If the operation fails, it will contain the error response

  In order to complete the operation, the application must access the specified location and use the correlator to retrieve the notification.

DELETE

The DELETE method removes a specified resource. The application provides the correlator or the identifier for the resource that must be removed in the Request-URI. For example:

```
DELETE /rest/terminal_status/notifications/6789 HTTP/1.1
```

For the DELETE method:

- The URI in the request contains the correlator, a value that uniquely identifies the resource the application wishes to remove. In the example, *6789* is the value which the application provided as the correlator when it requested for notifications on a terminal's status.
- The request body for this operation will be empty.
- The response body will be empty. If the operation fails, it will contain the error response.

Status-Line

The Status-Line is the first line in any response that an application receives when it interacts with the RESTful services interface in Services Gatekeeper. It takes the form:

```
HTTP/1.1 Status-Code Reason-Phrase
```

where,

- Status-Code: A three-digit indicator of the success or failure to fulfill the request.
- Reason-Phrase: A brief description of the (successful) action performed; or the reason for the failure.

For example:

```
HTTP/1.1 201 Created
```

Table 2–1 lists some of the status codes and reason-phrases commonly encountered when interacting with Services Gatekeeper's RESTful interfaces:

Table 2–1 A Sampling of Status Codes and Reason Phrases

Status-Code	Reason-Phrase	Description
200	OK	Success.
201	Created	Success. The requested resource was created.
204	No Content	Success
501	Internal Server Error	Indicates failure. An unexpected condition prevented the server from fulfilling the request.

For a complete listing of the HTTP Status Codes and their definitions, see RFC 2616 at

http://www.ietf.org/rfc/rfc2616.txt

Headers

The requests and responses for RESTful operations include the following header fields:

- Authorization: The Authorization header field is a required field and is found in all requests. It indicates the type of authentication and security. For example:

  ```
  Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
  ```

  For more information, see "Authentication and Security".

- Session ID: When Services Gatekeeper is running in Session mode, the X-Session-ID header must be present in all request messages to identify the application.

  In session mode, an application's first task is to obtain a session ID from the Session Manager. All traffic requests (for that session) include this identifier in the key-value pair for the X-Session-ID key. For example:

  ```
  X-Session-ID: app: -1780934689905632396
  ```

  The X-Session-ID header is not present when Services Gatekeeper is running in Sessionless mode. For more information on sessions, see "Session Manager"

- Service correlation ID (X-SCID): The X-SCID header will be present 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: Tunneled parameters (also called xparams) are present if the application wishes to supply parameters that are not supported in the RESTful interface itself and need to be passed on to the network. The key-value pairs are:

  - X-Param-Key and X-Param-Values: The X-Param-Key and X-Param-Values headers are found in the requests.
  - X-Plugin-Param-Key and X-Plugin-Param-Values: The X-Plugin-Param-Key and X-Plugin-Param-Values headers are returned in the response headers.

  See the appropriate sections in the *Oracle Communications Services Gatekeeper Communication Service Guide* for descriptions of the tunneled parameters that are applicable to your communication service.

- Location: Location headers are found in responses to certain requests. They are used for the identification of a new resource or to redirect the recipient to a location other than the Request-URI for completion of the request.

  The Location header in Example 2–5 contains the call identifier for a newly-setup call. In Example 11–12, it specifies the location that the application must access to receive notifications about a terminal (for which the application had previously initiated a notification request).

- Content-Length: The length of the request (or response) body.
- Content-Type: The MIME-type value for the Content-Type header field may be `multipart/form-data` or `application/json`. The `multipart/form-data` value for the Content-Type header field is described in the next section.

Headers for Multipart Messages with 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. When you use RESTful interfaces with the base Services Gatekeeper product, multiple attachments are supported in both application-initiated and network-triggered messages.

When a request message contains one or more messages embedded within it, a specified boundary is placed between the parts of the message and at the beginning and end of the message. For multipart message requests:

- The MIME-type value for the Content-Type header field may be `multipart/form-data` or `application/json`. If the MIME-type value for the Content-Type header field is `multipart/form-data`, the **boundary=** entry is used to provide a value for the boundary between the message parts.
- Each message part contains the following:
  - Content-Disposition header field whose value is `form-data` and a name attribute with the appropriate value. For example, the Message Part name is `messagePart`.
  - Content-Type header field whose value is `application/json` and the `charset` attribute with the appropriate value.
  - Content-Transfer-Encoding field with the appropriate value.
- If the content of the message is pure ASCII, the response body contains the message. Otherwise the response body contains an identifier that is used to fetch the actual message.

Example 2–1 Example of a Multipart Message Request

```
POST /rest/multimedia_messaging/messages HTTP/1.1
X-Session-ID: app: -123456789012346789
Authorization: Basic ZG9tY1uX3VzZXI6ZG9tYWluX3VzZXI=
X-Param-Keys:
```

```
X-Param-Values:
User-Agent: Jakarta Commons-HttpClient/3.0
Host: localhost:8001
Content-Length: 1215
Content-Type: multipart/form-data; boundary=kboiiFPAakDPYKeY7hBAW9I5c0rT48

--kboiiFPAakDPYKeY7hBAW9I5c0rT48
Content-Disposition: form-data; name="messagePart"
Content-Type: application/json; charset=US-ASCII
Content-Transfer-Encoding: 8bit

{
  "addresses":["tel:8765"],
  "subject":"Hello World",
  "priority":null,
  "senderAddress":"tel:1234",
  "charging":null
  "receiptRequest":
         {
           "correlator":"981234",
           "endpoint":"http://10.182.100.229:13444/jaxws/MessageNotification",
           "interfaceName":"interfaceName",
         }
 }
--kboiiFPAakDPYKeY7hBAW9I5c0rT48
Content-Disposition: form-data; name="Attachment-txt-1"
Content-Type: text/plain; charset=US-ASCII
Content-Transfer-Encoding: 8bit

This is the attachment text.
--kboiiFPAakDPYKeY7hBAW9I5c0rT48
```

Service Correlation

In some cases the service that an application provides to its end-users may involve accessing multiple Services Gatekeeper communication services.

For example, a mobile user might send an SMS to an application asking for a pizza restaurant 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 restaurant, and then sends the user an MMS with all the appropriate information. Three Services Gatekeeper communication services are involved in executing what for the application is a single service.

Services Gatekeeper uses a Service Correlation ID to correlate the three communication service requests. The Service Correlation ID (SCID) is a string that is captured in all the Charging Data Records (CDRs) and Event Data Records (EDRs) generated by 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.

How Service Correlation IDs are Provided Services Gatekeeper does not provide the Service Correlation ID. The type of request determines the Service Correlation ID:

- Application-Initiated Requests: 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.

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.

- Network-Triggered Requests: When the chain of services is initiated by a network-triggered request, Services Gatekeeper calls an external interface to get the SCID.

  Note that, this interface must be implemented by an external system. Integration of such an external interface 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.

Message Body

The message body for a request or response is present only when required. The message body is a JSON object.

Request Body

When present, the request body provides additional data required to complete the specific request. The following request body for an example Make Call operation provides the addresses of the called and calling party and any charges to apply for the call:

Example 2–2 Request Body for Make Call

```
{"calledParty":"sip:ann@10.182.100.229:1",
 "charging":null,
 "callingParty":"sip:zach@10.182.100.229:5020"
}
```

Response Body

When present, the response body provides data that the application will need for later action. The following response body for the Make Call operation provides the application with the identifier for the call that was set up. The application will use this identifier to end the call, when necessary.

Example 2–3 Response Body for Make a Make Call Operation

```
{"result":"app-1q39oi07wpvjl|e9674e8214447c1663a016d434c@10.182.100.229|-50d94925a
b34bf0"}
```

Example of a Request and Response

Example 2–4 shows an application's request to set up a call between two parties using the Make Call operation in Service Gatekeeper's RESTful interface.

Example 2–4 Request associated with a Make Call Operation

```
POST /rest/third_party_call/calls HTTP/1.1
X-Session-ID: app: -1780934689905632396
Authorization: Basic ZG9tY1uX3VzZXI6ZG9tYWluX3VzZXI=
X-Param-Keys:
X-Param-Values:
User-Agent: Jakarta Commons-HttpClient/3.0
Host: localhost:8001
Content-Length: 105
```

```
Content-Type: application/json

{"calledParty":"sip:alice@10.182.100.229:1",
 "charging":
      {
       "description":"init_call",
       "amount":"11",
       "code":"1111",
       "currency":"rmb"
      },
 "callingParty":"sip:bob@10.182.100.229:5020"
}
```

Example 2–5 shows the response which the application receives for a successful setup of the requested call.

Example 2–5 Response associated with a Make Call Operation

```
HTTP/1.1 201 Created
Date: Wed, 20 Oct 2010 06:58:06 GMT
Location: http://local:host:8001/rest/third_party_
call/call/app-1q39oi07wpvjl|e9674e8214447c1663a016d434c@10.182.100.229|-50d94925ab
34bf0
Content-Length: 96
Content-Type: application/json
X-Plugin-Param-Keys:
X-Plugin-Param-Values:
X-Powered-By: Servlet/2.5 JSP/2.1

{"result":"app-1q39oi07wpvjl|e9674e8214447c1663a016d434c@10.182.100.229|-50d94925a
b34bf0"}
```

Authentication and Security

The RESTful interfaces use HTTP basic authentication, using username/password. SSL is required. For instance:

```
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
```

For more information on HTTP basic authentication, see RFC 2617 at http://www.ietf.org/rfc/rfc2617.txt

Notifications and Publish/Subscribe

When an application needs to receive a notification, (about a message delivery receipt for example), the application uses the publish/subscribe functionality in Services Gatekeeper.

An application can subscribe only to its own notifications (that is, to the notifications associated with its start notification requests). Any attempt to subscribe to notifications for other applications will be rejected.

Supported Endpoint Addresses

The application provides an endpoint address that resides on a publish/subscribe server. It is possible to specify one of the following endpoint addresses:

- RESTful: a Bayeux protocol channel name

- SOAP: a Web service implemented by the application

A SOAP endpoint for the notification of a message sent using RESTful interfaces in Services Gatekeeper is valid only if the SOAP and RESTful interfaces reside in the same cluster.

Endpoint Addresses for RESTful Interfaces

The RESTful interfaces in Services Gatekeeper rely on the publish/subscribe model supported by the Publish-Subscribe Server functionality of Oracle WebLogic Server 10.3.1.

For more information on the publish/subscribe model, please see "Using the HTTP Publish-Subscribe Server" in *Oracle Fusion Middleware Developing Web Applications, Servlets, and JSPs for Oracle WebLogic Server* at:
http://download.oracle.com/docs/cd/E15523_01/web.1111/e13712/pubsub.htm.

Bayeux Protocol

When using RESTful interfaces, 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 (notifications) as they become available. The **endpoint** address resides on a Bayeux server.

Connections and Subscriptions

The mechanisms for connecting to the server and subscribing to a channel are covered by the Bayeux protocol itself. The Bayeux client manages connections to the server and subscriptions to a channel. If the channel does not exist when the client subscribes to it, the channel is created.

Once the Bayeux client connects to the server and subscribes to a channel, the RESTful client can start sending notifications. It does so by providing the Bayeux protocol channel name as the **endpoint** in a start notification request.

Bayeux Protocol Channel Name

The Bayeux protocol channel name begins with **`/bayeux/`***`appInstanceID`* where *appInstanceID* is the client application's application instance account ID. In Example 2–6, the *appInstanceID* is `domain-user`.

An application places the endpoint address for such delivery notifications in the body of the request message. Below, it is inside a **reference** object.

Example 2–6 Example of an Endpoint Address in a Reference Object

```
...
 "reference":
        {"interfaceName":"interfaceName",
         "correlator":"6789",
         "endpoint":"/bayeux/domain-user/ts"
        }
...
```

See Example 10–5 for the complete request body.

For more information on application instances, see "Creating and Maintaining Service Provider and Application Accounts" in *Oracle Communications Services Gatekeeper Accounts and SLAs Guide*.)

Using the Data at the Endpoint Address

Services Gatekeeper delivers notifications to the Bayeux channel name provided by the application in the associated request. It is the client's responsibility to interact with the Publish-Subscribe Server to access the messages/data placed at the **endpoint** address. The mechanisms to do so are outside the scope of the Services Gatekeeper RESTful Web Services.

For more information about the Bayeux protocol, see the Bayeux specification document at

http://svn.xantus.org/shortbus/trunk/bayeux/bayeux.html

Errors and Exceptions

The Status-Line in the response message indicates the protocol version, the three-digit Status Code and the reason for the failure of the request.

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"
           }
  }
```

For Service Exceptions, the value for **type** is:

```
"type":"org.csapi.schema.parlayx.common.v2_1.ServiceException"
```

For Policy Exceptions, the value for **type** is:

```
"type":"org.csapi.schema.parlayx.common.v2_1.PolicyException"
```

3

Third Party Call

This chapter describes the operations in the Third Party Call interface of the RESTful Web Services provided in Oracle Communications Services Gatekeeper.

About RESTful Third Party Call Interface

Applications use the RESTful Third Party Call interface in Services Gatekeeper 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.

Additionally, applications use this interface to specify the data required for the billing operation associated with the calls.

REST Service Descriptions Available at Run-time

When the Administration Server for your Services Gatekeeper domain is in the running state, the REST service descriptions of these operations can be found at

`http://host:port/rest/third_party_call/index.html`

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Make Call

The Make Call operation sets up a call between two parties referred to as the calling party and the called party.

To set up a call, provide the SIP-formatted URI of the calling party and the called party in the body of the request for the Make Call operation. Optionally, the request can also indicate any cost-charging parameters to be applied to the call.

If the Make Call operation is successful, the response header will contain the URI of the newly created resource as the value of the **Location** header field. Additionally, the response body will contain the call identifier for the newly created call object. Use this call identifier to reference the call for later.

Authorization

Basic

HTTP Method

`POST`

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Make Call operation accepts the following parameters:

- **calledParty**. String. Required. The address (URI) of the party to whom the call is made. Specified as `sip:`*`uname@destination_ip:destination_port`*.
- **callingParty**. String. Required. The address (URI) of the party making the call. Specified as `sip:`*`uname@origin_ip:origin_port`*.
- **charging**. a JSON object. Optional. This object defines the cost charging properties for the call. A call with no charging parameters can be entered as `"charging": null`.

  If a charge is to be applied, provide values for the following in the **charging** object:

  - **description**. String. Required if the **charging** object is present in the body of the request. The text to be used for information and billing.
  - **amount**. Number (integer, or decimal). Optional. The amount to be charged.
  - **code**. String. Optional. The charging code, from an existing contractual description.
  - **currency**. String. Optional. The currency identifier as defined in ISO 4217 [9].

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "calledParty": "URI",
  "callingParty": "URI",
  "charging": {
    "description": "String",
    "amount": "BigDecimal",
    "code": "String",
    "currency": "String"
  }
}
```

Response Header

The value in the Location header field is a single absolute URI in the following format:

`http://host:port/rest/third_party_call/${result}`

where *${result}* is a String-formatted call identifier for the newly-created call. See Example 3–2.

If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The response body for the Make Call operation contains the call identifier returned in the Location header field. The call identifier is the value for the **result** term in the response body represented by the following name/value pair structure:

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

See Example 3–2.

Examples

Example 3–1 Make Call Request

```
POST /rest/third_party_call/calls HTTP/1.1
X-Session-ID: app: -1780934689905632396
Authorization: Basic ZG9tY1uX3VzZXI6ZG9tYWluX3VzZXI=
X-Param-Keys:
X-Param-Values:
User-Agent: Jakarta Commons-HttpClient/3.0
Host: localhost:8001
Content-Length: 105
Content-Type: application/json

{"calledParty":"sip:alice@10.182.100.229:1",
 "charging":
      {
       "description":"testing",
       "amount":"11",
       "code":"1111",
       "currency":"rmb"
      },
 "callingParty":"sip:bob@10.182.100.229:5020"
}
```

Example 3–2 Make Call Response

```
HTTP:/1.1 201 Created
Date: Wed, 20 Oct 2010 06:58:06 GMT
Location: http://local:host:8001/rest/third_party_
call/call/app-1q39oi07wpvjl|e9674e8214447c1663a016d434c@10.182.100.229|-50d94925ab
34bf0
Content-Length: 96
Content-Type: application/json
X-Plugin-Param-Keys:
X-Plugin-Param-Values:
X-Powered-By: Servlet/2.5 JSP/2.1

{"result":"app-1q39oi07wpvjl|e9674e8214447c1663a016d434c@10.182.100.229|-50d94925a
b34bf0"}
```

Get Call Information

The Get Call Information operation retrieves the information on a previously-established call.

To retrieve the information on such a call, provide the appropriate call identifier in the GET request for this operation. This identifier should have been obtained by the initial set up request for the call.

If the operation is successful, the response body will contain the time the call started and the current status of the call. Additionally, if the call was terminated, the response body will indicate the total duration of the call and the reason for its termination.

Authorization

Basic

HTTP Method

`GET`

URI

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

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *${callIdentifier}* is the call identifier obtained from the response to the Make Call request.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

When the Get Call Information operation is successful, the response body contains the appropriate values for the following parameters that describe the call:

- **callStatus**. String. The current status of the call as one of the following values:
  - **CallInitial**: The call is being established.
  - **CallConnected**: The call is active.
  - **CallTerminated**: The call was terminated.

- **duration**: Integer. The duration of the call in seconds. Present in the response body when **callStatus** is **CallTerminated**.
- **startTime**. String. The start time for the call in ISO 8601 extended format `yyyy-mm-ddThh-mm-ss`.
- **terminationCause**. String. The reason for the termination of the call. Present in the response body only when **callStatus** is **CallTerminated**. Its value can be one of the following:
  - **CallAborted**
  - **CalledPartyBusy**
  - **CalledPartyNoAnswer**
  - **CalledPartyNotReachable**
  - **CallHangUp**
  - **CallingPartyBusy**
  - **CallingPartyNoAnswer**
  - **CallingPartyNotReachable**

These values are placed in a data structure as the value for **result** in the following JSON structure, where the value part of each name/value pair indicates its data type:

```
{"result": {
  "callStatus": "CallInitial|CallConnected|CallTerminated",
  "duration": "Integer",
  "startTime": "Calendar",
  "terminationCause":"CallingPartyNoAnswer|CalledPartyNoAnswer|CallingPartyBusy|Ca
lledPartyBusy|CallingPartyNotReachable|CalledPartyNotReachable|CallHangUp|CallAbor
ted"
}}
```

Examples

Example 3–3 Get Call Information Request

The GET command to get information on the call contains the call identifier.

```
GET /rest/third_party_
call/call/app-1q39oi07wpvjl|e9674e8214447c1663a016d434c@10.182.100.229|-50d94925ab
34bf0 HTTP/1.1
X-Session-ID: app: -123456789012346789
Authorization: Basic ZG9tY1uX3VzZXI6ZG9tYWluX3VzZXI=
X-Param-Keys:
X-Param-Values:
User-Agent: Jakarta Commons-HttpClient/3.0
Host: localhost:8001
```

Example 3–4 Get Call Information Response

```
HTTP:/1.1 201 Created
Date: Wed, 20 Oct 2010 06:58:06 GMT
Content-Length: 124
Content-Type: application/json
X-Plugin-Param-Keys:
X-Plugin-Param-Values:
X-Powered-By: Servlet/2.5 JSP/2.1
```

```
{"result":
  {"startTime":"2010-10-20T4:58:18.254+08:00",
   "terminationCause":null,
   "duration":"0",
   "callStatus":"CallConnected"
  }
}
```

Cancel Call Request

The Cancel Call Request operation cancels a previously-requested call that is in its initial state and not yet active. This operation will have no effect if the call is already established.

To cancel a call, provide the appropriate call identifier in the Request-URI for this operation. This identifier should have been obtained by the initial setup request for the call.

There is no request or response body for the Cancel Call operation. If the request fails, the body of the error response will contain the call identifier and the type of exception.

Authorization

Basic

HTTP Method

`POST`

URI

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

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *${callIdentifier}* is the call identifier obtained from the response to the Make Call request.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body.

Examples

Example 3–5 Cancel Call Request

The POST command to cancel the call contains the required call identifier.

```
POST /rest/third_party_
call/cancel-call/app-1q39oi07wpvjl|e9674e8214447c1663a016d434c@10.182.100.229|-50d
```

```
94925ab34bf0 HTTP/1.1
X-Session-ID: app: -123456789012346789
Authorization: Basic ZG9tY1uX3VzZXI6ZG9tYWluX3VzZXI=
X-Param-Keys:
X-Param-Values:
User-Agent: Jakarta Commons-HttpClient/3.0
Host: localhost:8001
Content-length: 0
```

Example 3–6 Cancel Call Response

If the Cancel Call Request operation succeeds, you will see a response similar to the following:

```
HTTP:/1.1 204 No Content
Date: Wed, 20 Oct 2010 07:48:14 GMT
Content-Length: 0
X-Plugin-Param-Keys:
X-Plugin-Param-Values:
X-Powered-By: Servlet/2.5 JSP/2.1
```

Example 3–7 Error Response Example

If the Cancel Call Request operation fails, you will see an error response similar to the following. The error body will contain the call identifier as the value for the `correlator` attribute.

```
HTTP:/1.1 500 Internal Server Error
Date: Wed, 20 Oct 2010 07:48:26 GMT
Content-Length: 261
Content-Type: application/json
X-Powered-By: Servlet/2.5 JSP/2.1

{"error" :
   {"message":"Invalid input value for message part Could not find a plugin for
this message: correlator:
app-1q39oi07wpvjl|e9674e8214447c1663a016d434c@10.182.100.229|-50d94925ab34bf0",
    "type":"org.csapi.schema.parlayx.common.v2_1.ServiceException"
   }
}
```

End Call

The End Call operation ends a call that is active.

To end a call, provide the appropriate call identifier in the Request-URI for this operation. This identifier should have been obtained by the initial setup request for the call.

There is no request or response body for this operation. If the request fails, the body of the error response will contain the call identifier and the type of exception.

Authorization

Basic

HTTP Method

POST

URI

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

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *${callIdentifier}* is the call identifier obtained from the response to the Make Call request.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body.

Examples

Example 3–8 End Call Request

```
POST /rest/third_party_
call/end-call/app-1q39oi07wpvjl|e9674e8214447c1663a016d434c@10.182.100.229|-50d949
25ab34bf0 HTTP/1.1
X-Session-ID: app: -123456789012346789
Authorization: Basic ZG9tY1uX3VzZXI6ZG9tYWluX3VzZXI=
X-Param-Keys:
```

```
X-Param-Values:
User-Agent: Jakarta Commons-HttpClient/3.0
Host: localhost:8001
Content-length: 0
```

Example 3–9 End Call Response

If the End Call operation succeeds, you will see a response similar to the following:

```
HTTP:/1.1 204 No Content
Date: Wed, 20 Oct 2010 07:48:14 GMT
Content-Length: 0
X-Plugin-Param-Keys:
X-Plugin-Param-Values:
X-Powered-By: Servlet/2.5 JSP/2.1
```

4

Anonymous Customer Reference

This chapter describes the use of OneAPI Anonymous Customer Reference (ACR) in Oracle Communications Services Gatekeeper.

About Anonymous Customer References

OneAPI supports using Anonymous Customer References (ACRs) as subscriber identifiers. An ACR represents a unique identifier replacing a subscriber's secure information, such as MSISDN or phone number, ensuring privacy when interacting with Web applications.

For information on the OneAPI ACR specification see:

http://www.gsma.com/oneapi/anonymous-customer-reference-beta

Services Gatekeeper supports ACRs by generating and managing one or more ACRs for a subscriber when requested by a Web application. A Web application requiring an ACR for a subscriber requests one from Services Gatekeeper using the RESTful API.

Services Gatekeeper supports the use of ACRs for all of the supported OneAPI services including SMS, MMS, Terminal Location and Payment.

Support for ACRs Using OAuth Authentication

In addition to basic and session ID authentication, Services Gatekeeper also supports OAuth authentication of applications requesting or querying ACR identifiers. Services Gatekeeper can use OAuth to provide subscribers SSO-like access to resources such as photos or Web sites.

Services Gatekeeper supports allowing OAuth access to subscriber resources using ACRs. Requests can use a valid accessToken to retrieve or update ACR information when submitted. The Services Gatekeeper OAuth Interceptor uses the provided accessToken to confirm a subscriber's identity before releasing ACR information.

For information about Services Gatekeeper OAuth support, see *Oracle Communications Services Gatekeeper OAuth Guide*.

Configuring ACR Support in Services Gatekeeper

Services Gatekeeper supports ACR operations by default. Applications can create, query and refresh ACRs using the RESTful API on a Services Gatekeeper system after you create an ACR plug-in instance.

Creating an ACR Plug-in Instance

To create an instance of the ACR plug-in in the Services Gatekeeper:

1. Log into the Administration Console.
2. Expand the **OCSG** node under **Domain Structure**.
3. Click on the name of the administration or managed server you want to create the ACR plug-in instance on.
4. Expand the **Container Services** node under **Oracle Communications Services Gatekeeper**.
5. Select **PluginManager**.
6. Click **Operations**.
7. In the **Select An Option** pull down menu select **createPluginInstsance**.
8. Enter `Plugin_acr` in the **PluginServiceId** field.
9. Enter a unique name in the **PluginInstanceId** field.
10. Click **Invoke**.
11. Add a route to the ACR plug-in using the **pluginManager** Mbean.

The Platform Test Environment MBean interface can also be used to create and manage ACR plug-ins. For information on using the Platform Test Environment, see *Oracle Communications Services Gatekeeper Platform Test Environment Guide*.

Setting ACR Plug-in Parameters

To configure the ACR plug-in attributes in Table 4–1:

1. Log into the Administration Console.
2. Expand the **OCSG** node under **Domain Structure**.
3. Select the administration or managed server where you created the ACR plug-in.
4. Expand the **Communication Services** node under **Oracle Communications Services Gatekeeper**.
5. Select the ACR plug-in instance to configure.
6. Click **Attributes**.
7. Select the checkboxes of the attributes you wish to change.
8. Enter the new values for the attribute(s).
9. Click **Update Attributes**.

Table 4–1 ACR Plug-in Attributes

Attribute	Type	Description
Ncc	String	The network-code of the operator. In Services Gatekeeper, this is the same as a service provider group.
AcrLifeTime	Integer	The number of seconds a generated or refreshed ACR is valid. Default value: **3600**
TrafficAcrMappingEnabled	Boolean	Whether to enable ACR mapping in network traffic. Default value: **False**

Creating Multiple ACRs for a Single Subscriber

You can create multiple ACRs for the same subscriber or MSISDN. Services Gatekeeper creates a unique ACR for each application. Use this configuration to route application requests containing ACR identifiers to the correct service provider group.

Using ACRs with Services Gatekeeper

Services Gatekeeper supports using ACRs by generating and managing one or more ACRs for a subscriber when requested by a Web application. A Web application requiring an ACR for a subscriber requests one from Services Gatekeeper using the RESTful API described below. Services Gatekeeper supports the use of ACRs for all of the supported OneAPI services including SMS, MMS, Terminal Location and Payment.

For more information on OneAPI services support, see *Oracle Communications Services Gatekeeper OneAPI Application Developer's Guide*.

Create ACR

The Create ACR operation creates a new ACR for a subscriber based on MSISDN.

Authorization

Basic or OAuth

HTTP Method

`POST`

URI

`http://host:port/CustomerReference/version/address`

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *address* is the subscriber identifier MSISDN. Services Gatekeeper supports an address entry of **acr**:*Authorization*, where authorization is an OAuth accessToken.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for Create ACR accepts the following parameters:

- **acr**. String.
- **status**. String.
- **expiry**. String.

Response Header

The response header indicates successful creation of the ACR. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions", for more information.

Response Body

The response body contains the following parameters:

- **acr**. String. The Services Gatekeeper generated ACR.
- **status**. String. The current status of the ACR.
- **expiry**. String. The expiration time of the ACR.

Examples

Example 4–1 shows a sample Create ACR request.

Example 4–1 Create ACR Request Example

```
POST http://example.com/CustomerReference/v1/tel%2B%3A7990123456
HTTP/1.1
Host: example.com:80
Accept: application/json

{"acr":{"status":"Valid"}}
```

Example 4–2 shows a sample Create ACR response.

Example 4–2 Create ACR Response Example

```
HTTP/1.1 201 Created

Content-Type: application/json
Content-Length: 1234
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"acr": "acr:0123456890123456789;ncc=23415",
"status":"Valid",
"expiry":"Thu 11 Jun 2009 02:51:59 GMT"}
```

Query ACR

The Query ACR operation queries the ACR status of the referenced subscriber.

Authorization

Basic or OAuth

HTTP Method

`GET`

URI

`http://host:port/CustomerReference/version/address/acr`

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *address* is the subscriber MSISDN.
- *acr* is the ACR for which the status query is being made including the network-code (ncc).

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The response body contains the following parameters:

- **acr**. String. The Services Gatekeeper generated ACR.
- **status**. String. The current status of the ACR.

Examples

Example 4–3 shows a sample Query ACR request.

Example 4–3 Query ACR Request Example

```
GET
http://example.com/CustomerReference/v1/tel%2B%3A7990123456/acr%2B0123456890123456
789%3Bncc=23415
```

Example 4–4 shows a sample Query ACR response.

Example 4–4 Query ACR Response Example

```
{"acr":{"status":"Expired"}}
```

Change ACR Status

The Change ACR Status operation refreshes an expired ACR to a valid status.

Authorization

Sessionless:Basic, Sessionful: Basic and Session ID, or OAuth

HTTP Method

`POST`

URI

`http://host:port/CustomerReference/version/address/acr`

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *address* is the subscriber MSISDN.
- *acr* is the ACR for which the status query is being made including the network-code (ncc).

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for Create ACR accepts the following parameters:

- **acr**. String.
- **status**. String.
- **expiry**. String.
- **developerId**. String. Required. The OneAPI developer ID.
- **applicationId**. String. Optional. The Services Gatekeeper application ID.

Response Header

The response header indicates successful change of the ACR status. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions", for more information.

Response Body

The response body contains the following parameters:

- **acr**. String. The Services Gatekeeper generated ACR.
- **status**. String. The updated status of the ACR.
- **expiry**. String. The new expiration time of the ACR.

Examples

Example 4–5 shows a sample Change ACR request.

Example 4–5

```
POST
http://example.com/CustomerReference/v1/tel%2B%3A7990123456/acr%2B0123456890123456
789%3Bncc=23415
HTTP/1.1
Host: example.com:80
Accept: application/json

{"acr":{"status":"Valid"}}
```

Example 4–6 shows a sample Change ACR response.

Example 4–6

```
Content-Type: application/json
Content-Length: 1234
Date: Thu, 04 Jun 2009 02:51:59 GMT

{"acr": "acr:0123456890123456789;ncc=23415",
"status":"Valid",
"expiry":"Thu 11 Jun 2014 02:51:59 GMT"
```

Errors and Exceptions and EDRs

Errors and Exceptions

The Status-Line in the response message indicates the protocol version, the three-digit Status Code and the reason for the failure of a request. Table 4–2 lists the possible error codes for failed requests.

Table 4–2 ACR Operations Error Codes

Error Code	Cause
303	create ACR failed, because the acr for the MSISDN has existed
400	The request with {address} = "MSISDN B" try to query/change acr of "MSISDN A"
401	The request from ncc A try to change/query acr of ncc B
404	query an un-existed ACR
503	inner server error

EDRs

Table 4–3 lists the EDRs generated by ACR operations.

Table 4–3 ACR Operations EDRs

EDR	Service	Method	Description
408001	oracle.ocsg.parlayrest.plugin.AcrPlugin	CreateAcrResp createAcr	create acr code for address
408002	oracle.ocsg.parlayrest.plugin.AcrPlugin	QueryAcrResp queryAcr	query acr code for address
408003	oracle.ocsg.parlayrest.plugin.AcrPlugin	ChangeAcrResp ChangeAcr	change acr code for address

5

Application Subscription Management

This chapter describes the use of the Application Subscription Management RESTful interface in Oracle Communications Services Gatekeeper.

About Application Subscription Management

Services Gatekeeper supports Open Mobile Alliance (OMA) General Service Subscription Management (GSSM) functionality including subscription management, subscription profile access and subscription validation with Application Subscription Management.

For information on the OMA GSSM specification see:

http://technical.openmobilealliance.org/Technical/release_program/gssm_v1_0.aspx

Application Subscription Management includes both a communication service and a RESTful interface for managing and querying service subscription status. Applications use the RESTful interface to manage subscriptions and query subscription status. Application Subscription Management grants or restricts application access to a subscriber's communication service(s) depending on the subscription status. Application Subscription Management also supports OAuth authentication when required.

For information on using the Application Subscription Management communication service, including deploying, configuring and monitoring, see the chapter Application Subscription Management in *Oracle Communications Services Gatekeeper Communication Service Guide*.

REST Service Descriptions Available at Run-time

When the Administration Server for your Services Gatekeeper domain is in the running state, the REST service descriptions of these operations can be found at

`http://`*host*`:`*port*`/subscription/application.wadl`

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Subscribe

The Subscribe operation creates a new application service subscription request for a subscriber based on MSISDN.

Authorization

Basic o r OAuth

HTTP Method

`POST`

URI

`http://host:port/subscription`

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for Subscribe accepts the following parameters:

- **applicationName**. String. The name of the application subscribed to.
- **subscriberAddress**. String. The MSISDN of the subscriber making the request.

Response Header

The response header indicates successful creation of the subscription request. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See the chapter on Application Subscription Management in *Oracle Communications Services Gatekeeper Communication Service Guide*, for more information on error messages.

Response Body

The response body contains the following parameters:

- **subscriptionResponse**. String. The Services Gatekeeper generated response containing the subscription ID.
  - **subscriptionID**. String. The ID of the new subscription.

Examples

Example 5–1 shows a sample Subscribe request.

Example 5–1 Subscribe Request Example

```
GET /subscription/query/queryBySubscriptionId/e4e3cb51-994b-46e1-b0d1-0757a8bca25f
HTTP/1.1
```

```
Host: example.com:80
Accept: application/json
```

Example 5–2 shows a sample Subscribe response.

Example 5–2 Subscribe Response Example

```
HTTP/1.1 201 Created
Content-Type: application/json
Date:  Wed, 07 Nov 2012 06:33:13 GMT

{"subscriptionResp":{"subscriptionId":"e4e3cb51-994b-46e1-b0d1-0757a8bca25f"}}
```

Unsubscribe

The Unsubscribe operation is used to request a subscription deletion.

Authorization

Basic or OAuth

HTTP Method

`DELETE`

URI

`http://host:port/subscription`

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The Unsubscribe request header contains the subscription ID to be deleted. The MIME-type for the Content-Type header field is **application/json**.

Request Body

There is no response body.

Response Header

The response header indicates successful change of the ACR status. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See the chapter on Application Subscription Management in *Oracle Communications Services Gatekeeper Communication Service Guide*, for more information on error messages.

Response Body

The response body contains no content

Examples

Example 5–3 shows a sample Unsubscribe request.

Example 5–3 Unsubscribe Request Example

```
DELETE /subscription/e4e3cb51-994b-46e1-b0d1-0757a8bca25f HTTP/1.1
Host: example.com:80
Accept: application/json
```

Example 5–4 shows a sample Unsubscribe response.

Example 5–4 Unsubscribe Response Example

```
HTTP/1.1 204 No Content
```

Suspend

The Suspend operation suspends an existing service subscription for a subscriber based on subscription ID.

Authorization

Basic

HTTP Method

`PUT`

URI

`http://host:port/subscription/suspend/subscriptionID`

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *subscriptionID* is the ID of the subscription to be suspended.

Request Header

The Suspend request header contains the subscription ID to be suspended. The MIME-type for the Content-Type header field is **application/json**

Request Body

The request body for Suspend is empty.

Response Header

The response header indicates successful subscription suspension. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See the chapter on Application Subscription Management in *Oracle Communications Services Gatekeeper Communication Service Guide*, for more information on error messages.

Response Body

The response body contains no content.

Examples

Example 5–5 shows a sample Suspend request.

Example 5–5 Suspend Request Example

```
PUT /subscription/suspend/e4e3cb51-994b-46e1-b0d1-0757a8bca25f HTTP/1.1
Host: example.com:80
Accept: application/json
```

Example 5–6 shows a sample Suspend response.

Example 5–6 Suspend Response Example

```
HTTP/1.1 204 No Content
```

Unsuspend

The Unsuspend operation unsuspends an existing service subscription for a subscriber based on subscription ID.

Authorization

Basic

HTTP Method

`PUT`

URI

`http://host:port/subscription/unsuspend/subscriptionID`

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *subscriptionID* is the ID of the subscription to be suspended.

Request Header

The Unsuspend request header contains the subscription ID to be unsuspended. The MIME-type for the Content-Type header field is **application/json**

Request Body

The request body for unsuspend is empty.

Response Header

The response header indicates successful subscription unsuspension. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See the chapter on Application Subscription Management in *Oracle Communications Services Gatekeeper Communication Service Guide*, for more information on error messages.

Response Body

The response body contains no content.

Examples

Example 5–7 shows a sample Unsuspend request.

Example 5–7 Unsuspend Request Example

```
PUT /subscription/unsuspend/e4e3cb51-994b-46e1-b0d1-0757a8bca25f HTTP/1.1
Host: example.com:80
Accept: application/json
```

Example 5–8 shows a sample Unuspend response.

Example 5–8 Unsuspend Response Example

```
HTTP/1.1 204 No Content
```

Notify

The Notify operation notifies a registered application of a subscription management request.

Authorization

Basic or OAuth

HTTP Method

`POST`

URI

`http://host:port/notify`

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for Notify accepts the following parameters:

- **subscriptionNotification**. The Services Gatekeeper generated notification containing the following parameters:
  - **subscriptionID**. String. The ID of the subscription request.
  - **applicationName**. String. The name of the application receiving the request.
  - **subscriberAddress**. String. The MSISDN of the requesting subscriber.
  - **operation**. String. The subscription request type. For example, **Subscribe** or **Unsubscribe**.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See the chapter on Application Subscription Management in *Oracle Communications Services Gatekeeper Communication Service Guide*, for more information on error messages.

Response Body

The response body contains no content.

Examples

Example 5–9 shows a sample Notify request.

Example 5–9 Notify Request Example

```
POST /notifysubscription HTTP/1.1
Host: example.com:80
Accept: application/json
Content-Type: application/json
Content-Length: 173

{"subscriptionNotification":{"subscriptionId":"e4e3cb51-994b-46e1-b0d1-0757a8bca25
f","applicationName":"Oracle
News","subscriberAddress":"tel:1111","operation":"Subscribe"}}
```

Example 5–10 shows a sample Notify response.

Example 5–10 Notify Response Example

```
HTTP/1.1 204 No Content
```

Confirm

The Confirm operation is used by applications to confirm subscription requests.

Authorization

Basic or OAuth

HTTP Method

`PUT`

URI

`http://host:port/subscription/confirm`

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for Confirm accepts the following parameters:

- **operation**. String. The subscription request operation being confirmed. For example, **Subscribe** or **Unsubscribe**.
- **confirmResult**. String. The subscription request status. For example, **approved**.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See the chapter on Application Subscription Management in *Oracle Communications Services Gatekeeper Communication Service Guide*, for more information on error messages.

Response Body

The response body contains the following parameter:

- **oauthAccessToken**. String. If OAuth is required, Services Gatekeeper returns an OAuth accessToken in the response.

Examples

Example 5–11 shows a sample Confirm request.

Example 5–11 Confirm Request Example

```
PUT /subscription/confirm/e4e3cb51-994b-46e1-b0d1-0757a8bca25f HTTP/1.1
Host: example.com:80
Accept: application/json
Content-Type: application/x-www-form-urlencoded
Content-Length: 42
```

```
operation=Subscribe&confirmResult=Approved
```

Example 5–12 shows a sample Confirm response with an OAuth accessToken.

Example 5–12 Confirm Response Example

```
HTTP/1.1 200 OK
Content-Type: application/json
Date:  Wed, 07 Nov 2012 06:39:50 GMT

{"oauthAccessToken":{"oauthAccessToken":"e4e3cb51-994b-46e1-b0d1-0757a8bca25f"}}
```

queryBySubscriberAddress

The queryBySubscriberAddress operation retrieves a user's subscription information based on the subscriber address.

Authorization

Basic

HTTP Method

`GET`

URI

`http://host:port/subscription/query/queryBySubscriberAddress`

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The queryBySubscriberAddress request header contains the subscriber address (MSISDN) to be queried. The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for queryBySubscriberAddress is empty.

Response Header

The response header includes the request status. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See the chapter on Application Subscription Management in *Oracle Communications Services Gatekeeper Communication Service Guide*, for more information on error messages.

Response Body

The request body for queryBySubscriberAddress contains the following parameters:

- **subscriptionList**. The Services Gatekeeper generated list containing the following parameters:
  - **subscriptionInfo**. Array. Contains the following parameters for each subscription.
    - **applicationName**. String. The name of the subscribed application.
    - **subscriberAddress**. String. The MSISDN of the subscriber.
    - **status**. String. The subscription status. For example, **Pending**.

Examples

Example 5–13 shows a sample queryBySubscriberAddress request.

Example 5–13 queryBySubscriberAddress Request Example

```
GET /subscription/query/queryBySubscriberAddress/tel:1111?offSet=0&batchSize=2
HTTP/1.1
Host: example.com:80
Accept: application/json
```

Example 5–14 shows a sample queryBySubscriberAddress response.

Example 5–14 queryBySubscriberAddress Response Example

```
HTTP/1.1 200 OK
Content-Type: application/json
Date:  Wed, 07 Nov 2012 06:56:05 GMT

{"subscriptionList":{"subscriptionInfo":[{"applicationName":"Oracle
Jokes","subscriberAddress":"tel:1111","status":"SubscribePending"},{"applicationNa
me":"Oracle News","subscriberAddress":"tel:1111","status":"UnSubscribePending"}]}}
```

queryByApplicationName

The queryByApplicationName operation retrieves a list of subscribers to an application based on the application ID.

Authorization

Basic

HTTP Method

`GET`

URI

`http://host:port/subscription/query/queryByApplicationName`

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The queryByApplicationName request header contains the application name to be queried. The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for queryBySubscriberAddress is empty.

Response Header

The response header includes the request status. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See the chapter on Application Subscription Management in *Oracle Communications Services Gatekeeper Communication Service Guide*, for more information on error messages.

Response Body

The request body for queryByApplicationName contains the following parameters:

- **subscriptionList**. The Services Gatekeeper generated list containing the following parameters:
  - **subscriptionInfo**. Array. Contains the following parameters for each subscribed user.
    - **applicationName**. String. The name of the subscribed application.
    - **subscriberAddress**. String. The MSISDN of the subscriber.
    - **status**. String. The subscription status. For example, **Pending**.

Examples

Example 5–15 shows a sample queryByApplicationName request.

Example 5–15 queryByApplicationName Request Example

```
GET /subscription/query/queryByApplicationName/Oracle%20News?offSet=0&batchSize=2
HTTP/1.1
Host: example.com:80
Accept: application/json
```

Example 5–16 shows a sample queryByApplicationName response.

Example 5–16 queryByApplicationName Response Example

```
HTTP/1.1 200 OK
Content-Type: application/json
Date:  Wed, 07 Nov 2012 06:56:05 GMT

{"subscriptionList":{"subscriptionInfo":[{"applicationName":"Oracle
News","subscriberAddress":"tel:1111","status":"UnSubscribePending"},{"applicationN
ame":"Oracle News","subscriberAddress":"tel:1112","status":"Active"}]}}
```

queryBySubscriptionID

The queryBySubscriptionID operation retrieves application subscription information for a subscriber based on the subscription ID.

Authorization

Basic

HTTP Method

GET

URI

`http://host:port/subscription/query/queryBySubscriptionID`

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The queryBySubscriptionID request header contains the subscription ID to be queried. The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for queryBySubscriptionID is empty.

Response Header

The response header includes the request status. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See the chapter on Application Subscription Management in *Oracle Communications Services Gatekeeper Communication Service Guide*, for more information on error messages.

Response Body

The request body for queryBySubscriptionID contains the following:

- **subscriptionInfo**. The Services Gatekeeper generated list with the following parameters:
  - **applicationName**. String. The name of the subscribed application.
  - **subscriberAddress**. String. The MSISDN of the subscriber.
  - **status**. String. The subscription status. For example, Pending.

Examples

Example 5–17 shows a sample queryBySubscriptionID request.

Example 5–17 queryBySubscriptionID Request Example

```
GET /subscription/query/queryBySubscriptionId/e4e3cb51-994b-46e1-b0d1-0757a8bca25f
HTTP/1.1
```

```
Host: example.com:80
Accept: application/json
```

Example 5–18 shows a sample queryBySubscriptionID response.

Example 5–18 queryBySubscriptionID Response Example

```
HTTP/1.1 200 OK
Content-Type: application/json
Date:  Wed, 07 Nov 2012 06:56:05 GMT

{"subscriptionInfo":{"applicationName":"Oracle
News","subscriberAddress":"tel:1111","status":"UnSubscribePending"}}HTTP/1.1 200
OK
```

6

Call Notification

This chapter describes the operations in the Call Notification interface of the RESTful Web Services provided in Oracle Communications Services Gatekeeper.

About the Call Notification Interface

Applications use the RESTful Call Notification interface to set up and remove call notifications which inform the application about the particular state of a call (such as busy, unreachable, and so on).

Additionally, applications use this interface to set up and remove call direction notifications (which require the application to provide further directions on handling a call that is in a particular state).

For such operations, the application client subscribes to and uses the Bayeux channel to which Services Gatekeeper is subscribed. The application provides the Bayeux channel name as the location for Services Gatekeeper to publish the notifications that the application client requires. See "Notifications and Publish/Subscribe".

REST Service Descriptions Available at Run-time

When the Administration Server for your Services Gatekeeper domain is in the running state, the REST service descriptions of these operations can be found at

`http://host:port/rest/call_notification/index.html`

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Common Data

The details of a call event about which the application receives information are identical whether the called party was busy **Busy**, not reachable **NotReachable**, did not answer **NoAnswer**, and a call was being attempted **CalledNumber**.

These details are provided to the application in a JSON object which contains the following parameters:

- **calledParty**. String. Required. The address (URI) of the party to whom the call is made.
- **callingParty**. String. Required. The address (URI) of the party making the call.
- **correlator**. String. Required. The correlator used to identify the notification and provided in the Start Call Notification request.
- **callingPartyName**. String. Optional. The name of the party making the call.

These details are specified in the following structure:

```
{
  "calledParty": "URI",
  "callingParty": "URI",
  "correlator": "String",
  "callingPartyName": "String"
}
```

You will find this object in the following operations:

- Start Call Notification: This object is used to provide the call details in the notifications sent (by Services Gatekeeper) to the application following a successful Start Call Notification Request. See "Notification Data Object Delivered to Bayeux Channel Name".
- Start Call Direction Notification: This object is used to provide the call details in the notifications sent (by Services Gatekeeper) to the application to request instructions (from the application) on handling a call event that has occurred. See "Notification Data Object Requesting Instruction from Application".

Start Call Notification

The Start Call Notification operation sets up call notifications to a specified endpoint address for call events associated with the addresses specified in the request.

To set up a call notification, provide the SIP-formatted URI of the addresses for which the application must receive notifications, the criteria that will trigger notifications, and a reference object for the delivery of the notifications. The criteria can be one or more of four possible call events (**Busy**, **NotReachable**, **NoAnswer**, and **CalledNumber**). If you do not specify any value for **criteria**, each of the four call events will trigger a notification.

The reference object (also a JSON object) contains the **correlator** for the notification, the **endpoint** address to which the notifications must be sent and, optionally, the interface name (a string to identify the notification).

If the Start Call Notification request is successful:

- The response header will contain the URI of the publish/subscribe server.
- Notification data objects appropriate to the call event will be sent to the endpoint address specified in the request body. Each data object will contain the appropriate notification on the call.

When the application receives such a response, it must access the **endpoint** address to retrieve the specific call event notifications.

Authorization

Basic

HTTP Method

`PUT`

URI

`http://`*`host`*`:`*`port`*`/rest/call_notification/call_notification`

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for Start Call Notification accepts the following parameters:

- **addresses**. Array of string values. Required. The set of addresses to be monitored, provided as a list of URIs separated by commas.
- **criteria**. String. Optional. The call event for which the notification is required.

  If you do not specify any value for **criteria**, all of the four call events will trigger a notification. The four call events are:

  – **Busy**: This entry indicates that the called party is busy.

  - **CalledNumber**: This entry indicates that a call between the two parties is being attempted.
  - **NoAnswer**: This entry indicates that the called party does not answer.
  - **NotReachable**: This entry indicates that the called party is not reachable.
- **reference**. A JSON object. Required. Use this object to provide the following information about the endpoint address that is to receive the notification:
  - **correlator**. String. Required. The correlator used to identify the call notifications for the specified addresses.
  - **endpoint**. String. Required. The endpoint address to which the notification should be delivered. This string should be a Bayeux protocol channel name that begins with **`/bayeux/`***`appInstanceID`* where *appInstanceID* is the client application's application instance account ID.

    For more information on application instances, see the discussion on creating and maintaining service provider and application accounts in *Oracle Communications Services Gatekeeper Accounts and SLAs Guide*.
  - **interfaceName**. String. Optional. A descriptive string to identify the notification.

The request body for this operation is represented by the following JSON structure, where the value part of each name/value pair indicates its data type:

```
{
  "addresses": ["String"],
  "reference": {
    "correlator": "String",
    "endpoint": "URI",
    "interfaceName": "String"
  },
  "criteria": ["Busy|NotReachable|NoAnswer|CalledNumber"]
}
```

Response Header

The Location header field contains the URI of the publish/subscribe server as:

`http://`*`host`*`:`*`port`*`/rest/call_notification/notifications`

If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body. The appropriate notifications (described below) are sent to the **endpoint** address provided by the application in the request body of this operation.

Notification Data Object Delivered to Bayeux Channel Name

Services Gatekeeper delivers the appropriate notification for each of the following call events:

- **Busy**: When the called party is busy, the notification is sent as the value for **notifyBusy** represented by the following JSON structure:

```
{"notifyBusy": {
  "calledParty": "URI",
```

```
  "callingParty": "URI",
  "correlator": "String",
  "callingPartyName": "String"
}}
```

- **NotReachable**: When the called party is not reachable, the notification is sent as the value for **notifyNotReachable** represented by the following JSON structure:

```
{"notifyNotReachable": {
  "calledParty": "URI",
  "callingParty": "URI",
  "correlator": "String",
  "callingPartyName": "String"
}}
```

- **NoAnswer**: When the called party does not answer, the notification is sent as the value for **notifyNoAnswer** represented by the following JSON structure:

```
{"notifyNoAnswer": {
  "calledParty": "URI",
  "callingParty": "URI",
  "correlator": "String",
  "callingPartyName": "String"
}}
```

- **CalledNumber**: When the called party call is being attempted, the notification is sent as the value for **notifyCalledNumber** represented by the following JSON structure:

```
{"notifyCalledNumber": {
  "calledParty": "URI",
  "callingParty": "URI",
  "correlator": "String",
  "callingPartyName": "String"
}}
```

Stop Call Notification

The Stop Call Notification operation generates a request that terminates a previously set up call notification.

To stop a previously set up call notification, use the appropriate correlator in the DELETE method for this operation. This correlator should have been specified earlier in the request body of the Start Call Notification operation.

There is no request or response body for the Stop Call Notification operation. If the request fails, the body of the error response will contain the identifier for the notification and the type of exception.

Authorization

Basic

HTTP Method

`DELETE`

URI

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

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *${correlator}* is the correlator for the notification provided in the **reference** object of the initial Start Call Notification request.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body.

Start Call Direction Notification

The Start Call Direction Notification operation initiates notifications to a specified endpoint for call events associated with the addresses specified in the request. The notifications provided by Services Gatekeeper will contain the information necessary for the application to provide instructions on directing the call.

To set up a call direction notification, provide the SIP-formatted URI of the addresses for which the application must receive notifications, the criteria that will trigger notifications, and a reference object for the delivery of the notifications. The criteria can be one more of four possible call events (**Busy**, **NotReachable**, **NoAnswer**, and **CalledNumber**. If you do not specify any value for **criteria**, each of the four call events will trigger a notification.

The reference object (also a JSON object) contains the **correlator** for the notification, the **endpoint** address (a specific Bayeux channel name) to which the call direction notifications must be sent and, optionally, the interface name (a string to identify the notification).

If the Start Call Direction Notification request is successful:

- The response header will contain the URI of the publish/subscribe server.
- A notification data object appropriate to the call event will be sent to the endpoint address specified in the request body. This data object will contain the appropriate notification on the call. It will also contain the reply-to private channel address to be used by the application to provide further instructions on handling and routing the call.

When it receives such a response, the application:

1. Retrieves the notification data object from the endpoint address. See "Notification Data Object Requesting Instruction from Application".
2. Sets up a data object with instructions on how the call must be handled (and charged); and an address to which the call must now be routed. See "Call-Handling Instructions".
3. Publishes this data object to the reply-to private channel address it received. See "Application's Call-Handling Response to Services Gatekeeper".

Authorization

Basic

HTTP Method

`PUT`

URI

`http://host:port/rest/call_notification/call_direction`

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Start Call Direction Notification accepts the following parameters:

- **addresses**. Array of string values. Required. The set of addresses to be monitored, provided as a list of SIP-formatted URIs separated by commas.
- **criteria**. String. Optional. The call event for which the notification is required.

  If you do not specify any value for **criteria**, all of the four call events will trigger a notification. The four call events are:

  - **Busy**: This entry indicates that the called party is busy.
  - **CalledNumber**: This entry indicates that a call between the two parties is being attempted.
  - **NoAnswer**: This entry indicates that the called party does not answer.
  - **NotReachable**: This entry indicates that the called party is not reachable.
- **reference**: a JSON object. Required. Use this object to provide the following information about the call direction notification:
  - **correlator**. String. Required. The correlator used to identify the call direction notification for the specified addresses.
  - **endpoint**. String. Required. The URI which represents the endpoint address to which the call direction notification should be delivered. This string should be a Bayeux protocol channel name that begins with **`/bayeux/`***`appInstanceID`* where *appInstanceID* is the client application's application instance account ID.

    For more information on application instances, see the discussion on creating and maintaining service provider and application accounts in *Oracle Communications Services Gatekeeper Accounts and SLAs Guide*.
  - **interfaceName**. String. Optional. A descriptive string to identify the notification.

The request body for this operation is represented by the following JSON structure, where the value part of each name/value pair indicates its data type:

```
{
  "addresses": ["String"],
  "reference": {
    "correlator": "String",
    "endpoint": "URI",
    "interfaceName": "String"
  },
  "criteria": ["Busy|NotReachable|NoAnswer|CalledNumber"]
}
```

Response Header

The Location header in the response displays the URI of the publish/subscribe server:

`http://`*`host`*`:`*`port`*`/rest/call_notification/notifications`

The application must access this location to retrieve the notification or delivery status information about the request.

If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body.

The **endpoint** address holds the appropriate notification data object containing information on the type of call event when there is a match for the **criteria** value specified in the request body of this operation.

Notification Data Object Requesting Instruction from Application

When the application needs to take further action on a call event, Services Gatekeeper provides the application with a corresponding notification data object for the application to act upon.

It places this notification data object at the **endpoint** address mentioned above. The notification data object is a nested JSON object containing the following parameters:

- **reply-to**. String. The Bayeux channel address to which the application must deliver its instructions for handling the call. This is a private Bayeux channel set up by Services Gatekeeper.
- **data**. Nested JSON object. Services Gatekeeper provides the appropriate call details as the value for **handleBusy**, **handleNotReachable**, **handleNoAnswer**, **handleCalled Number** named for the four types of call events. These details have been described under "Common Data".
  - **handleBusy**: For a **Busy** call event, Services Gatekeeper provides the appropriate call details as the value for the attribute **handleBusy** in the following JSON structure:

```
{"handleBusy": {
  "calledParty": "URI",
  "callingParty": "URI",
  "correlator": "String",
  "callingPartyName": "String"
}}
```

  - **handleNotReachable**: For a **NotReachable** call event, Services Gatekeeper provides the appropriate call details as the value for the attribute **handleNotReachable** in the following JSON structure:

```
{"handleNotReachable": {
  "calledParty": "URI",
  "callingParty": "URI",
  "correlator": "String",
  "callingPartyName": "String"
}}
```

  - **handleNoAnswer**: For a **NoAnswer** call event, Services Gatekeeper provides the appropriate call details as the value for the attribute **handleNoAnswer** in the following JSON structure:

```
{"handleNoAnswer": {
  "calledParty": "URI",
  "callingParty": "URI",
  "correlator": "String",
  "callingPartyName": "String"
}}
```

  - **handleCalledNumber**: For a **CalledNumber** call event, Services Gatekeeper provides the appropriate call details as the value for the attribute **handleCalledNumber** in the following JSON structure:

```
{"handleCalledNumber": {
  "calledParty": "URI",
  "callingParty": "URI",
  "correlator": "String",
  "callingPartyName": "String"
}}
```

Call-Handling Instructions

Whether the call event is **Busy**, **NotReachable**, **NoAnswer** or **CalledNumber**, the application needs to specify the following:

- **actionToPerform**. String. Required. Specifies the required action to take on the call. Table 6–1 lists the actions that Services Gatekeeper must perform with respect to the call:

Table 6–1 Possible Values for Actions to Perform

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

- **charging**. a JSON object. Optional. This object defines the cost charging properties for the call. A call with no charging parameters is entered as `"charging":null`. If a charge is to be applied, values for the following in the **charging** object:
  - **description**. String. Required if the **charging** object is present in the body of the request. The text to be used for information and billing.
  - **amount**. Number (integer, or decimal). Optional. The amount to be charged.
  - **code**. String. Optional. The charging code, from an existing contractual description.
  - **currency**. String. Optional. The currency identifier as defined in ISO 4217 [9].
- **routingAddress**. String. URI. The address to which the call must be routed.

Application's Call-Handling Response to Services Gatekeeper

The application sends the appropriate instruction (described in the previous section) as the response to the **reply-to** address it had retrieved earlier from the **endpoint** address. See "Notification Data Object Requesting Instruction from Application".

The possible responses that the application provides to Services Gatekeeper are:

- Instructions to handle a **Busy** call event: The application provides the instruction data as the value for **handleBusyResponse** in the following JSON structure:

```
{"handleBusyResponse": {"result": {
  "actionToPerform": "Route|Continue|EndCall",
  "charging": {
    "description": "String",
    "amount": "BigDecimal",
    "code": "String",
    "currency": "String"
  },
  "routingAddress": "URI"
}}}
```

- Instructions to handle a **NotReachable** event: The application provides the instruction data as the value for **handleNotReachableResponse** in the following JSON structure:

```
{"handleNotReachableResponse": {"result": {
  "actionToPerform": "Route|Continue|EndCall",
  "charging": {
    "description": "String",
    "amount": "BigDecimal",
    "code": "String",
    "currency": "String"
  },
  "routingAddress": "URI"
}}}
```

- Instructions to handle a **NoAnswer** event: The application provides the instruction data as the value for **handleNoAnswerResponse** in the following JSON structure:

```
{"handleNoAnswerResponse": {"result": {
  "actionToPerform": "Route|Continue|EndCall",
  "charging": {
    "description": "String",
    "amount": "BigDecimal",
    "code": "String",
    "currency": "String"
  },
  "routingAddress": "URI"
}}}
```

- Instructions to handle a **CalledNumber** event: The application provides the instruction data as the value for **handleCalledNumberResponse** call event in the following JSON structure:

```
{"handleCalledNumberResponse": {"result": {
  "actionToPerform": "Route|Continue|EndCall",
  "charging": {
    "description": "String",
    "amount": "BigDecimal",
    "code": "String",
    "currency": "String"
  },
  "routingAddress": "URI"
}}}
```

Stop Call Direction Notification

The Stop Call Direction Notification operation terminates a previously set up call direction notification.

To stop a previously set-up call direction notification, use the appropriate correlator in the DELETE request for this operation. This correlator should have been specified earlier in the request body of the Start Call Direction Notification operation.

There is no request or response body for the Stop Call Direction Notification operation. If the request fails, the body of the error response will contain the identifier for the notification and the type of exception.

Authorization

Basic

HTTP Method

`DELETE`

URI

`http://host:port/rest/call_notification/call_direction/${correlator}`

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *${correlator}* is the correlator for the notification provided in the **reference** object of the initial Start Call Direction Notification request.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body.

7

Short Messaging

This chapter describes the operations in the Short Messaging interface of the RESTful Web Services provided in Oracle Communications Services Gatekeeper.

About the Short Messaging Interface

Applications use the RESTful Short Messaging interface to send an SMS, a ringtone, or a logo, to fetch SMSes and delivery status reports; and to start and stop a notification.

When the request body for an SMS operation contains a request for a delivery receipt, the application provides a correlator for the message being sent and includes an endpoint address for returning the delivery notification.

For such operations, the application client subscribes to and uses the Bayeux channel to which Services Gatekeeper is subscribed. The application provides the Bayeux channel name as the location for Services Gatekeeper to publish the notifications that the application client requires. See "Notifications and Publish/Subscribe".

REST Service Descriptions Available at Run-time

When the Administration Server for your Services Gatekeeper domain is in the running state, the REST service descriptions of these operations can be found at

`http://host:port/rest/sms/index.html`

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Send Sms

The Send Sms operation delivers a text message.

To send an Sms message, provide the SIP-formatted URI of the addresses which must receive the message in the request body. If there is to be a charge for the messaging, the request body should contain the required charging object. If the sender requires a delivery receipt, specify the required parameters for the receipt.

If the Send Sms operation is successful, the **Location** header field in the response will contain the request identifier (which is also provided in the response body for this operation).

If the application requested a receipt for delivery of the message, the application must access the **endpoint** address to retrieve the delivery notifications.

Authorization

Basic

HTTP Method

`POST`

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Send Sms operation accepts the following parameters:

- **addresses**. Array of string values. Required. The set of addresses of the recipients as an array of SIP-formatted URIs.
- **message**. String. Required. The text of the message.
- **charging**. a JSON object. Optional. This object defines the cost charging properties for the operation. The entry `"charging": null` indicates no charge. If a charge is to be applied, provide values for the following in the **charging** object:
  - **description**. String. Required if the **charging** object is present in the body of the request. The text to be used for information and billing.
  - **amount**. Number (integer, or decimal). Optional. The amount to be charged.
  - **code**. String. Optional. The charging code, from an existing contractual description.
  - **currency**. String. Optional. The currency identifier as defined in ISO 4217 [9].
- **receiptRequest**. A JSON object. Optional. If a delivery receipt is required, provide values for each of the following parameters which define this object:

  - **correlator**. String. Required. Used to correlate the receipt with the initial message.

    If the callback reference **correlator** is defined in the **receiptRequest** attribute of the Send Sms request message, Services Gatekeeper will invoke the Notify Sms Delivery Receipt operation and discard the delivery status information.

  - **endpoint**. String. Required. The endpoint address (URI) to which the receipt must be delivered.

  - **interfaceName**. String. Required. A description provided to identify the type of receipt.

- **senderName**. String. Optional. The sender's name.

The request body for this operation is represented by the following JSON structure, where the value part of each name/value pair indicates its data type:

```
{
  "addresses": ["URI"],
  "message": "String",
  "charging": {
    "description": "String",
    "amount": "BigDecimal",
    "code": "String",
    "currency": "String"
  },
  "receiptRequest": {
    "correlator": "String",
    "endpoint": "URI",
    "interfaceName": "String"
  },
  "senderName": "String"
}
```

Response Header

The Location header field contains the URI:

`http://`*`host`*`:`*`port`*`/rest/sms/delivery_status/`*`result`*

where, *result* is the string identifier returned in the response body.

If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The body of the response contains the request identifier as the string value for the **result** attribute. It is the request identifier returned in the **Location** header field of the response message. The application uses this request identifier (in the Get Sms Delivery Status operation) to retrieve the delivery status for the sent message.

The response body for this operation is represented by the following JSON structure, where the value part of the name/value pair indicates its data type:

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

Send Sms Ringtone

The Send Sms Ringtone operation delivers a ringtone.

To send an Sms Ringtone message, provide the SIP-formatted URI of the addresses which must receive the message, the ringtone data and its format in the request body for this operation. If there is to be a charge for the messaging, the request body should contain the required charging object. If the sender requires a delivery receipt, specify the required parameters for the receipt.

If the Send Sms Ringtone operation is successful, the **Location** header field in the response will contain the request identifier (which is also provided in the response body for this operation).

If the application requested a receipt for delivery of the message, the application must access the endpoint address to retrieve the delivery notifications.

Authorization

Basic

HTTP Method

`POST`

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body is a nested JSON object containing the following parameters:

- **addresses**. Array of string values. Required. The set of addresses of the recipients as an array of SIP-formatted URIs.
- **ringtone**. String. Required. The ringtone data in Extended Ringtone (RTX) format.
- **smsFormat**. String. Required. The encoding format for the ringtone entered as one of the following:
  - **Ems**
  - **SmartMessaging**
- **charging**. A JSON object. Optional. This object defines the cost charging properties for the operation. The entry `"charging": null` indicates no charge. If a charge is to be applied, provide values for the following in the **charging** object:
  - **description**. String. Required if the **charging** object is present in the body of the request. The text to be used for information and billing.
  - **amount**. Number (integer, or decimal). Optional. The amount to be charged.

  - **code**. String. Optional. The charging code, from an existing contractual description.
  - **currency**. String. Optional. The currency identifier as defined in ISO 4217 [9].
- **receiptRequest**. A JSON object. Optional. If a delivery receipt is required, provide values for each of the following parameters which define this object:
  - **correlator**. String. Required. Used to correlate the receipt with the initial message.

    If the callback reference **correlator** is defined in the **receiptRequest** object of the Send Sms request message, Services Gatekeeper will invoke the Notify Sms Delivery Receipt operation and discard the delivery status information.
  - **endpoint**. String. Required. The endpoint address (URI) to which the receipt must be delivered.
  - **interfaceName**. String. Required. A description provided to identify the type of receipt.
- **senderName**. String. Optional. The sender's name.

The request body for this operation is represented by the following JSON structure, where the value part of each name/value pair indicates its data type:

```
{
  "addresses": ["URI"],
  "ringtone": "String",
  "smsFormat": "Ems|SmartMessaging",
  "charging": {
    "description": "String",
    "amount": "BigDecimal",
    "code": "String",
    "currency": "String"
  },
  "receiptRequest": {
    "correlator": "String",
    "endpoint": "URI",
    "interfaceName": "String"
  },
  "senderName": "String"
}
```

Response Header

`http://host:port/rest/sms/delivery_status/result`

where, *result* is the string identifier returned in the request body.

If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The body of the response contains the **result** attribute whose value is the request identifier returned in the Location header field of the response message. The application uses this request identifier (in the Get Sms Delivery Status operation) to retrieve the delivery status for the sent message.

The response body for this operation is represented by the following JSON structure, where the value part of the name/value pair indicates its data type:

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

Send Sms Logo

The Send Sms Logo operation delivers a logo.

To send an Sms Logo, provide the SIP-formatted URI of the addresses which must receive the message, the logo data and its format in the request body for this operation. If there is to be a charge for the messaging, the request body should contain the required charging object. If the sender requires a delivery receipt, specify the required parameters for the receipt.

If the Send Sms Logo operation is successful, the **Location** header field in the response will contain the request identifier (which is also provided in the response body for this operation).

If the application requested a receipt for delivery of the message, the application must access the **endpoint** address to retrieve the delivery notifications.

Authorization

Basic

HTTP Method

POST

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Send Sms operation accepts the following parameters:

- **addresses**. Array of string values. Required. The set of addresses of the recipients as an array of SIP-formatted URIs.
- **image**. String. Required. The logo data, a base64-encoded image in GIF, PNG or JPG format.
- **smsFormat**. String. Required. The encoding format for the logo entered as one of the following:
  - **Ems**
  - **SmartMessaging**
- **charging**. a JSON object. Optional. Use this object to define the cost charging properties for the operation. The entry `"charging": null` indicates no charge. If a charge is to be applied, provide values for the following in the **charging** object:
  - **description**. String. Required if the **charging** object is present in the body of the request. The text to be used for information and billing.
  - **amount**. Number (integer, or decimal). Optional. The amount to be charged.

  - **code**. String. Optional. The charging code, from an existing contractual description.
  - **currency**. String. Optional. The currency identifier as defined in ISO 4217 [9].
- **receiptRequest**. a JSON object. Optional. If a delivery receipt is required, provide values for each of the following parameters which define this object:
  - **correlator**. String. Required. Used to correlate the receipt with the initial message.

    If the callback reference **correlator** is defined in the **receiptRequest** object of the Send SMS request message, Services Gatekeeper will invoke the Notify Sms Delivery Receipt operation and discard the delivery status information.
  - **endpoint**. String. Required. The endpoint address (URI) to which the receipt must be delivered.
  - **interfaceName**. String. Required. A description provided to identify the type of receipt.
- **senderName**. String. Optional. The sender's name.

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "addresses": ["URI"],
  "image": "base64Binary",
  "smsFormat": "Ems|SmartMessaging",
  "charging": {
    "description": "String",
    "amount": "BigDecimal",
    "code": "String",
    "currency": "String"
  },
  "receiptRequest": {
    "correlator": "String",
    "endpoint": "URI",
    "interfaceName": "String"
  },
  "senderName": "String"
}
```

Response Header

The Location header field contains the URI:

`http://`*`host`*`:`*`port`*`/rest/sms/delivery_status/`*`result`*

where, *result* is the string identifier returned in the request body.

If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The response body contains the **result** attribute whose value is the request identifier returned in the Location header field of the response message. The application uses this request identifier (in the Get Sms Delivery Status operation) to retrieve the delivery status for the sent message.

The response body for this operation is represented by the following JSON data structure, where the value part of the name/value pair indicates its data type:

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

Get Received Sms

The Get Received Sms operation polls the Services Gatekeeper for the SMSes that have been received from the network for an application.

The request header for the Get Received SMS operation contains the registration identifier necessary to retrieve the required SMSes intended for the application. This registration value should have been set up with the off-line provisioning step that enables the application to receive notification of SMS reception.

There is no request body.

If the Get Received Sms operation is successful, the response body will contain the message, the URI of the sender, the SMS service activation number and the date and time when the message was sent.

Authorization

Basic

HTTP Method

`POST`

URI

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

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *registrationIdentifier* is the value previously set up to enable the application to receive notification of SMS reception according to specified criteria.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The response body contains an array of structures as the value for **result**. Each element in the array contains values for the following parameters.

- **message**. String. The text of the Sms message.
- **senderAddress**. String. The SIP-formatted URI of the sender.
- **smsServiceActivationNumber**. String. URI. The number associated with the invoked Message service, that is the destination address used to send the message.

- **dateTime**. String. The start time for the call in ISO 8601 extended format `yyyy-mm-ddThh-mm-ss`.

The response body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{"result": [{
  "message": "String",
  "senderAddress": "URI",
  "smsServiceActivationNumber": "URI",
  "dateTime": "Calendar"
}]}
```

Get Sms Delivery Status

The Get Sms Delivery Status operation retrieves the delivery status of a message that was previously sent using Send Sms, Send Sms Ringtone or Send Sms Logo operation.

The Get Sms Delivery Status operation is valid only if the **correlator** callback reference was not defined in the receiptRequest attribute of the initial request to send the short message. In such a scenario, Services Gatekeeper stores the delivery status for a configurable period of time.

(When the **correlator** callback reference is defined in the **receiptRequest** object of the initial request to send the short message, Services Gatekeeper will invoke the Notify Sms Delivery Receipt operation and discard the delivery status information.)

The request header for Get Sms Delivery Status contains the request identifier from the initial Send operation for the short message.

If Get Sms Delivery Status is successful, the response body will contain the SIP-formatted URIs from the **address** field in the request body of the Send operation for the short message and corresponding delivery status for each message.

Authorization

Basic

HTTP Method

`GET`

URI

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

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *requestIdentifier* is the identifier returned in the **result** object of the corresponding Send operation.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The response body contains an array of structures as the value for **result**. Each element in the array contains values for the following parameters.

- **address**. String. The SIP-formatted URI to which the initial message was sent.
- **deliveryStatus**: Enumeration value. Table 7–1 lists the possible status:

Table 7–1 Enumeration Values for Delivery Status

Value	Description
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.
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.

The response body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{"result": [{
  "address": "URI",
  "deliveryStatus":
"DeliveredToNetwork|DeliveryUncertain|DeliveryImpossible|MessageWaiting|DeliveredT
oTerminal|DeliveryNotificationNotSupported"
}]}
```

Start Sms Notification

The Start Sms Notification operation starts a notification for the application.

To set up an Sms notification, provide the criteria which will trigger notifications and a reference object for the delivery of the notifications. The criteria can be a string which, when matched, could be the notification of an Sms received or of a delivery receipt. The reference object (also a JSON object) contains the correlator for the notification, the **endpoint** address (a specific Bayeux channel name) to which the call direction notifications must be sent and, optionally, the interface name (a string to identify the notification).

If the Start Sms Notification request is successful:

- The response header will contain the URI of the publish/subscribe server.
- A data object associated with the result of the short message operation will be sent to the **endpoint** address specified in the request body. This data object will contain the appropriate notification (that the message was received or a delivery receipt for the call).

When the Start Sms notification request is successful, the application can access the **endpoint** address to retrieve the specific call event notifications.

Authorization

Basic

HTTP Method

`PUT`

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Start Sms Notification operation accepts the following parameters:

- **criteria**. String. Optional. The text to match against in determining whether or not the application should receive the notification:
  - **notifySmsReception**
  - **notifySmsDeliveryReceipt**
- **reference**. JSON object. Required. Use this object to provide the following information about the endpoint that is to receive the notification:
  - **correlator**. String. Required. The correlator used to identify the notification.

  - **endpoint**. String. Required. The URI which represents the endpoint address to which the notification should be delivered. This string should be a Bayeux protocol channel name that begins with **/bayeux/***appInstanceID* where *appInstanceID* is the client application's application instance account ID.

    For more information on application instances, see the discussion on creating and maintaining service provider and application accounts in *Oracle Communications Services Gatekeeper Accounts and SLAs Guide*.

  - **interfaceName**. String. Required. A descriptive string to identify the type of notification.

- **smsServiceActivationNumber**. String. Required. The number associated with the invoked Message service, that is the destination address used to send the message

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
  "reference": {
    "correlator": "String",
    "endpoint": "URI",
    "interfaceName": "String"
  },
  "smsServiceActivationNumber": "URI",
  "criteria": "String"
}
```

Response Header

The Location header field contains the URI of the publish/subscribe server:

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

If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body. The appropriate notifications (described below) are sent to the **endpoint** address provided by the application in the request body of this operation.

Notification Data Object for Sms Reception (notifySmsReception)

When an Sms is successfully retrieved by the called party and the value for **criteria** in the request body is **notifySmsReception**, Services Gatekeeper will send a nested JSON data object to the **endpoint** address.

This nested JSON object contains the following as the value of the attribute name **notifySmsReception**:

- **correlator**. String. Required. The correlator used to identify the notification.
- **message**. Nested JSON object. Required. It contains the following:
  - **message**. String. Required. The contents of the Sms message.
  - **senderAddress**. String. Required. The SIP-formatted URI of the sender of the Sms message.

- **smsServiceActivationNumber**. String. Required. The number associated with the invoked Message service, that is the destination address used to send the message.
- **dateTime**. String. Optional. The time at which the message was sent in ISO 8601 extended format `yyyy-mm-ddThh-mm-ss`.

The notification data object delivered to the **endpoint** address when the criteria is **notifySmsReception** is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{"notifySmsReception": {
  "correlator": "String",
  "message": {
    "message": "String",
    "senderAddress": "URI",
    "smsServiceActivationNumber": "URI",
    "dateTime": "Calendar"
  }
}}
```

Notification Data Object for Sms Delivery Receipt (notifySmsDeliveryReceipt)

When an Sms is successfully retrieved by the called party and the value for **criteria** in the request body is **notifySmsDeliveryReceipt**, Services Gatekeeper sends a nested JSON data object to the **endpoint** address.

This nested JSON object contains the following as the value of the attribute name **notifySmsDeliveryReceipt**:

- **correlator**. String. The correlator used to identify the notification.
- **deliveryStatus**. Enumeration value. Table 7–2 lists the possible status:

Table 7–2 Enumeration Values for Delivery Status

Value	Description
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.
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.

The notification data object delivered to the **endpoint** address when the criteria is **notifySmsDeliveryReceipt** is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{"notifySmsDeliveryReceipt": {
  "correlator": "String",
  "deliveryStatus": {
    "address": "URI",
```

```
    "deliveryStatus":
"DeliveredToNetwork|DeliveryUncertain|DeliveryImpossible|MessageWaiting|DeliveredT
oTerminal|DeliveryNotificationNotSupported"
  }
}}
```

Stop Sms Notification

The Stop Sms Notification operation terminates a previously set up Sms notification for the application.

To stop a previously set up Sms notification, provide the correlator for the notification passed earlier in the Start Sms Notification request.

There is no request or response body for the Stop Sms Notification operation. If the request fails, the body of the error response will contain the identifier for the notification and the type of exception.

Authorization

Basic

HTTP Method

`DELETE`

URI

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

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *correlator* is the correlator for the notification provided in the **reference** object of the initial Start Sms Notification request body.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body.

Examples

Example 7–1 Stop Message Notification Request

```
DELETE /rest/sms/notification/12345 HTTP/1.1
X-Session-ID: app:4130997928482260925
Authorization: Basic ZG9tYWluX3VzZXI6ZG9tYWluX3VzZXI=
X-Param-Keys:
X-Param-Values:
User-Agent: Jakarta Commons-HttpClient/3.0
Host:10.182.100.218.8002
```

Example 7–2 Stop Message Notification Response

```
HTTP/1.1 204 No Content
Connection: close
Date: Thu, 04 Nov 2101 09:59:05 GMT
Content-Length: 0
X-Plugin-Param-Keys:
X-Plugin-Param-Values:
X-Powered-By: Servlet/2.5 JSP/2.1
```

8

Multimedia Messaging

This chapter describes the operations in the Multimedia Messaging interface of the RESTful Web Services provided in Oracle Communications Services Gatekeeper.

About the Multimedia Messaging Interface

Applications use the RESTful Multimedia Messaging interface to send a multimedia message (MMS) and to fetch information on MMSes that have been received for the applications and stored on Services Gatekeeper.

They use the interface to fetch those messages, get delivery status on sent messages, and start and stop a notification.

The actual message is sent as an attachment. See "Headers for Multipart Messages with Attachments" for more information.

In order for an application to be informed of an MMS that has been received for a party or a delivery receipt for a message sent, the application includes an application **endpoint** address for such delivery notifications in the body of the request message. This address resides on a publish/subscribe server. For more information, see "Notifications and Publish/Subscribe".

REST Service Descriptions Available at Run-time

When the Administration Server for your Services Gatekeeper domain is in the running state, the REST service descriptions of these operations can be found at

`http://host:port/rest/multimedia_messaging/index.html`

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Send Message

The Send Message operation delivers a multimedia message.

To send a multimedia (MMS) message, provide the SIP-formatted URI of the addresses which the must receive the message in the request body. If there is to be a charge for the messaging, the request body should contain the required charging object. If the sender requires a delivery receipt, specify the required parameters for the receipt and the SIP formatted URI address of the sender. Also, a priority value, and a subject line can be provided in the request body.

If the Send Message operation successful, the response header will contain the URI of the pub-sub server and the request identifier which is also provided in the response body for this operation.

Authorization

Basic

HTTP Method

`POST`

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The request headers depend on the type of message:

- If the message content is in the form of an attachment, the request will be multipart and therefore, the request header must contain the header fields that describe the parts of the message.
- If the message does not contain an attachment, the special headers associated with the multipart messaged are not used.

See Example 8–1 for a listing of both types of headers.

Message Part Content

The Message Part Content for the Send Message operation accepts the following parameters:

- **addresses**. Array of string values. Required. The set of addresses of the recipients as an array of URIs.
- **subject**. String. Optional. The text of the message.
- **priority**. String. Optional. The message priority specified as one of the following:
  - **Default**
  - **Low**
  - **Normal**

  - **High**

- **senderAddress**. String. Optional. The sender's address (URI).
- **charging**. a JSON object. Optional. This object defines the cost charging properties for the operation. The entry `"charging":null` indicates no charge. If a charge is to be applied, provide values for the following in the **charging** object:
  - **description**. String. Required if the **charging** object is present in the body of the request. The text to be used for information and billing.
  - **amount**. Number (integer, or decimal). Optional. The amount to be charged.
  - **code**. String. Optional. The charging code, from an existing contractual description.
  - **currency**. String. Optional. The currency identifier as defined in ISO 4217 [9].
- **receiptRequest**: a JSON object. Optional. If a delivery receipt is required, provide values for each of the following parameters which define this object:
  - **correlator**. String. Required. Used to correlate the receipt with the initial message.

    If the callback reference **correlator** is defined in the receiptRequest attribute of the Send Message request, Services Gatekeeper will invoke Notify Message Delivery Receipt and discard the delivery status information.
  - **endpoint**. String. The endpoint address (URI) to which the receipt must be delivered, a Bayeux protocol channel name that is the client application's application instance account ID.
  - **interfaceName**. String. Required. A description provided to identify the type of receipt.

The Message Part Content for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "addresses": ["URI"],
  "charging": {
    "description": "String",
    "amount": "BigDecimal",
    "code": "String",
    "currency": "String"
  },
  "priority": "Default|Low|Normal|High",
  "receiptRequest": {
    "correlator": "String",
    "endpoint": "URI",
    "interfaceName": "String"
  },
  "senderAddress": "String",
  "subject": "String"
}
```

Response Header

The Location header field contains the URI of the publish/subscribe server as:

`http://`*`host`*`:`*`port`*`/rest/multimedia_messaging/delivery_status/`*`result`*

where, *result* is the string identifier for the request that is returned in the response body.

If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The body of the response contains the request identifier returned in the **Location** header field of the response message as value for **result**.

The response body for this operation is represented by the following JSON data structure, where the value part of the name/value pair indicates its data type:

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

Examples

Example 8–1 Send Message Requests

This section contains two examples. The first example shows a POST method with a message.

```
POST /rest/multimedia_messaging/messages HTTP/1.1
X-Session-ID: app:-7603991555266551180
Authorization: Basic ZG9tY1uX3VzZXI6ZG9tYWluX3VzZXI=
X-Param-Keys:
X-Param-Values:
User-Agent: Jakarta Commons-HttpClient/3.0
Host: localhost:8001
Content-Length: 253
Content-Type: application/json

{
  "addresses":["tel:8765","tel:7654","tel:6543"],
  "subject":"Meeting starts at 2 p.m.",
  "priority":null,
  "senderAddress":"tel:1234",
  "charging":null
  "receiptRequest":
         {
           "correlator":"981234",
           "endpoint":"http://10.182.100.229:13444/jaxws/MessageNotification",
           "interfaceName":"interfaceName",
         }
 }
```

The second example shows a POST method with a multipart message. Note the use of header fields to describe the content and the boundary attribute to distinguish the message parts.

```
POST /rest/multimedia_messaging/messages HTTP/1.1
X-Session-ID: app: -123456789012346789
Authorization: Basic ZG9tY1uX3VzZXI6ZG9tYWluX3VzZXI=
X-Param-Keys:
X-Param-Values:
User-Agent: Jakarta Commons-HttpClient/3.0
Host: localhost:8001
Content-Length: 1215
Content-Type: multipart/form-data; boundary=kboiiFPAakDPYKeY7hBAW9I5c0rT48
```

```
--kboiiFPAakDPYKeY7hBAW9I5c0rT48
Content-Disposition: form-data; name="messagePart"
Content-Type: application/json; charset=US-ASCII
Content-Transfer-Encoding: 8bit

{
  "addresses":["tel:8765","tel:7654","tel:6543"],
  "subject":"Hello World",
  "priority":null,
  "senderAddress":"tel:1234",
  "charging":null,
  "receiptRequest":
         {
           "correlator":"981234",
           "endpoint":"http://10.182.100.229:13444/jaxws/MessageNotification",
           "interfaceName":"interfaceName"
         }
 }
--kboiiFPAakDPYKeY7hBAW9I5c0rT48
Content-Disposition: form-data; name="Attachment-txt-1"
Content-Type: text/plain; charset=US-ASCII
Content-Transfer-Encoding: 8bit

Oracle Communications Services Gatekeeper is a powerful, flexible, secure
interface which provides a simple way for application developers to include
telephony-based functionality in their software applications and guarantee the
security, stability, and performance required by network operators and demanded by
their subscribers.
--kboiiFPAakDPYKeY7hBAW9I5c0rT48
```

Get Received Messages

The Get Received Messages operation retrieves an array of network-triggered messages that have arrived at Services Gatekeeper.

The request body for Get Received Messages contains the registration identifier necessary to retrieve the required MMSs intended for the application and, optionally, it may also contain a priority for the retrieval. The registration identifier should have been set up with the off-line provisioning step completed earlier.

If the Get Received Messages operation is successful, the response body will contain the data about the message. If the content of the message is pure ASCII, the response body contains the message. Otherwise the response body contains an identifier that is used to fetch the actual message.

Authorization

Basic

HTTP Method

`POST`

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Get Received Messages operation accepts the following parameters:

- **registrationIdentifier**. String. Required. The value previously set up to enable the application to receive notification of MMS reception according to specified criteria.

  For more information on provisioning offline notifications, see "Managing and Configuring Multimedia Messaging Communication" in the *Oracle Communications Services Gatekeeper System Administrator's Guide*, another document in this document set.

- **priority**. String. Optional. The message priority specified as one of the following:
  - **Default**
  - **Low**
  - **Normal**
  - **High**

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
```

```
  "registrationIdentifier": "String",
  "priority": "Default|Low|Normal|High"
}
```

Response header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The response body contains an array of structures as the value for **result**. Each element in the array contains values for the following parameters.

- **messageServiceActivationNumber**. String. The number associated with the invoked Message service, that is the destination address used to send the message.
- **priority**. String. The message priority value from the request.
- **senderAddress**. String. The SIP-formatted URI of the sender.
- **dateTime**. String. The start time for the call in ISO 8601 extended format `yyyy-mm-ddThh-mm-ss`.
- **message**. String. Present if the text of the message is purely ASCII text.
- **messageIdentifier**. String. An identifier used to fetch the message, when it is not purely ASCII text. This value is used in the URI of the request for the "Get Message" operation.
- **subject**. String. The subject line of the multimedia message.

The response body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{"result": [{
  "messageServiceActivationNumber": "String",
  "priority": "Default|Low|Normal|High",
  "senderAddress": "URI",
  "dateTime": "Calendar",
  "message": "String",
  "messageIdentifier": "String",
  "subject": "String"
}]}
```

Get Message

The Get Message operation fetches a network-triggered message for the application from Services Gatekeeper.

To get a message, provide the appropriate message identifier in the URI for the GET method.

The actual message is returned as an attachment.

Authorization

Basic

HTTP Method

`GET`

URI

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

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *messageRefIdentifier* is the value for *messageIdentifier* in:
  - the response body of the "Get Received Messages" operation
  - **notifyMessageReception** data object. See "Notification Data Object for Message Reception (notifyMessageReception)"

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

There is no request body.

Response Header

The MIME-type for the Content-Type header field is **multipart/mixed**.

If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body.

Get Message Delivery Status

The Get Message Delivery Status operation retrieves the delivery status of a previously sent message.

This operation is valid only if the **correlator** callback reference was not defined in the **receiptRequest** attribute of the initial request to send the MMS message. In such a scenario, Services Gatekeeper stores the delivery status for a configurable period of time. (If the callback reference correlator is defined in the **receiptRequest** attribute of the initial request to send the MMS message, Services Gatekeeper will invoke the Notify Message Delivery Receipt operation and discard the delivery status information.)

The request header for the Get Message Delivery Status operation contains the request identifier from **result** object of the initial request.

If the Get Message Delivery Status operation is successful, the response body will contain the SIP-formatted URI from the address field in the Send Message request and corresponding delivery status for the message.

Authorization

Basic

HTTP Method

`GET`

URI

`http://`*`host`*`:`*`port`*`/rest/multimedia_messaging/delivery_status/`*`${requestIdentifier}`*

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *requestIdentifier* is a string returned in the **result** object of the initial Send Message request.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The response body contains an array of structures as the value for **result**. Each element in the array contains values for the following parameters.

- **address**. String. The SIP-formatted URI to which the initial message was sent.

- **deliveryStatus**. Enumeration value. Table 8–1 lists the possible statuses:

Table 8–1 Enumeration Values for Delivery Status

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

The response body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{"result": [{
  "address": "URI",
  "deliveryStatus":
"DeliveredToTerminal|DeliveryUncertain|DeliveryImpossible|MessageWaiting|Delivered
ToNetwork|DeliveryNotificationNotSupported"
}]}
```

Start Message Notification

The Start Message Notification operation starts a notification for the application.

To set up a multimedia message notification, provide the criteria which will trigger notifications and a reference object for the delivery of the notifications. The criteria can be a string which, when matched, could be the notification of an MMS received or of a delivery receipt. The reference object (also a JSON object) contains the correlator for the notification, the endpoint address (a specific Bayeux channel name) to which the call direction notifications must be sent and, optionally, the interface name (a string to identify the notification).

If the Start Message Notification request is successful:

- The response header will contain the URI of the publish/subscribe server.
- A data object associated with the result of the short message operation will be sent to the endpoint address specified in the request body. This data object will contain the appropriate notification (that the message was received or a delivery receipt for the call).

When the application receives such a response, it must access the endpoint address to retrieve the specific call event notifications.

Authorization

Basic

HTTP Method

`PUT`

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body is a nested JSON object containing the following parameters:

- **reference**. JSON object. Required. Use this object to provide the following information about the endpoint that is to receive the notification:
  - **correlator**. String. Required. The correlator used to identify the notification.
  - **endpoint**. String. Required. The URI which represents the endpoint address to which the notification should be delivered. This string should be a Bayeux protocol channel name that begins with **`/bayeux/`***`appInstanceID`* where *appInstanceID* is the client application's application instance account ID.

    For more information on application instances, see the discussion on creating and maintaining service provider and application accounts in *Oracle Communications Services Gatekeeper Accounts and SLAs Guide*.

  - **interfaceName**. String. Required. A descriptive string to identify the type of notification.
- **messageServiceActivationNumber**. String. Required. The number associated with the invoked Message service, that is the destination address used to send the message
- **criteria**. String. Optional. The text to match against in determining whether or not the application should receive the notification:
  - **notifyMessageReception**
  - **notifyMessageDeliveryReceipt**

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "messageServiceActivationNumber": "URI",
  "reference": {
    "correlator": "String",
    "endpoint": "URI",
    "interfaceName": "String"
  },
  "criteria": "String"
}
```

Response Header

The Location header field contains the URI of the publish/subscribe server:

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

If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body. The appropriate notifications (described below) are sent to the **endpoint** address provided by the application in the request body of this operation.

Notification Data Object for Message Reception (notifyMessageReception)

When a multimedia message is successfully retrieved by the called party and the value for **criteria** in the request body is **notifyMessageReception**, Services Gatekeeper sends a nested JSON data object to the **endpoint** address.

This nested JSON object contains the following as the value of the attribute name **notifyMessageReception**:

- **correlator**. String. The correlator used to identify the notification.
- **message**. Nested JSON object. It contains the following:
  - **message**. String. The contents of the multimedia message.
  - **senderAddress**. String. The SIP-formatted URI of the sender of the multimedia message.

- **smsServiceActivationNumber**. String. The number associated with the invoked Message service, that is the destination address used to send the message.
- **dateTime**. String. The time at which the message was sent in ISO 8601 extended format `yyyy-mm-ddThh-mm-ss`

The notification data object delivered to the **endpoint** address when the criteria is **notifyMessageReception** is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{"notifyMessageReception": {
  "correlator": "String",
  "message": {
    "messageServiceActivationNumber": "String",
    "priority": "Default|Low|Normal|High",
    "senderAddress": "URI",
    "dateTime": "Calendar",
    "message": "String",
    "messageIdentifier": "String",
    "subject": "String"
  }
}}
```

Notification Data Object for MMS Delivery Receipt (notifyMessageDeliveryReceipt)

When a multimedia message is successfully retrieved by the called party and the value for **criteria** in the request body is **notifyMessageDeliveryReceipt**, Services Gatekeeper sends a nested JSON data object to the **endpoint** address.

This nested JSON object contains the following as the value of the attribute name **notifyMessageDeliveryReceipt**:

- **correlator**. String. The correlator used to identify the notification.
- **deliveryStatus**. Enumeration value. Table 8–2 lists the possible statuses:

Table 8–2 Enumeration Values for Delivery Status

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

The notification data object delivered to the **endpoint** address when the criteria is **notifyMessageDeliveryRecipt** is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{"notifyMessageDeliveryReceipt": {
```

```
  "correlator": "String",
  "deliveryStatus": {
    "address": "URI",
    "deliveryStatus":
"DeliveredToTerminal|DeliveryUncertain|DeliveryImpossible|MessageWaiting|Delivered
ToNetwork|DeliveryNotificationNotSupported"
  }
}}
```

Stop Message Notification

The Stop Message Notification operation terminates a previously set up multimedia message notification for the application.

To stop a previously set up MMS notification, provide the correlator for the notification passed earlier in the Start Message Notification request.

There is no request or response body for the Stop Message Notification operation. If the request fails, the body of the error response will contain the identifier for the notification and the type of exception.

Authorization

Basic

HTTP Method

`DELETE`

URI

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

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *correlator* is the correlator for the notification provided in the **reference** object of the initial Start Message Notification operation.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body.

Examples

Example 8–2 Stop Message Notification Request

```
DELETE /rest/multimedia_messaging/notification/6789 HTTP/1.1
X-Session-ID: app:4130997928482260925
Authorization: Basic ZG9tYWluX3VzZXI6ZG9tYWluX3VzZXI=
X-Param-Keys:
X-Param-Values:
User-Agent: Jakarta Commons-HttpClient/3.0
Host:10.182.100.218.8002
```

Example 8–3 Stop Message Notification Response

```
HTTP/1.1 204 No Content
Connection: close
Date: Thu, 04 Nov 2101 09:59:05 GMT
Content-Length: 0
X-Plugin-Param-Keys:
X-Plugin-Param-Values:
X-Powered-By: Servlet/2.5 JSP/2.1
```

9

Email Communication Service

This chapter describes the operations in the Email communication interface of the RESTful Web Services provided in Oracle Communications Services Gatekeeper.

About the Email Communication Interface

Applications use the RESTful Email Communication interface to send an email message and to fetch information on email messages that have been received for the applications and stored on Services Gatekeeper.

They use the interface to fetch those messages, get delivery status on sent messages, and start and stop a notification.

The actual message is sent as an attachment. See "Headers for Multipart Messages with Attachments" for more information.

In order for an application to be informed of an email message that has been received for a party or a delivery receipt for a message sent, the application includes an application **endpoint** address for such delivery notifications in the body of the request message. This address resides on a publish/subscribe server. For more information, see "Notifications and Publish/Subscribe".

REST Service Descriptions Available at Run-time

When the Administration Server for your Services Gatekeeper domain is in the running state, the REST service descriptions of these operations can be found at

`http://host:port/rest/multimedia_messaging/index.html`

where *host* and *port* are the host name and port of the Services Gatekeeper access tier server.

Send Message

The Send Message operation delivers an email message.

To send an email message, provide the SIP-formatted URI of the addresses which the must receive the message in the request body. If there is to be a charge for the messaging, the request body should contain the required charging object. If the sender requires a delivery receipt, specify the required parameters for the receipt and the SIP formatted URI address of the sender. Also, a priority value, and a subject line can be provided in the request body.

The plug-in manager uses the **email:** prefix in the sender address and destination address to route the email message to the email plug-in instance.

If the Send Message operation successful, the response header will contain the URI of the pub-sub server and the request identifier which is also provided in the response body for this operation.

Authorization

Basic

HTTP Method

`POST`

URI

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

where *host* and *port* are the host name and port name of the Services Gatekeeper access tier.

Request Header

The request headers depend on the type of message:

- If the message content is in the form of an attachment, the request will be multipart and therefore, the request header must contain the header fields that describe the parts of the message.
- If the message does not contain an attachment, the special headers associated with the multipart messaged are not used.
- The X-Parameter **ContentInFirstAttachment** is added to indicate whether there is an email body or not. If this X-Parameter is set to "true," or is absent, then the first attachment contains the email body and remaining attachments are handled as regular attachments. If this X-Parameter is set to "false," then all attachments are handled as regular attachments and the email body is blank.

See Example 9–1 for a listing of both types of headers.

Message Part Content

The Message Part Content for the Send Message operation accepts the following parameters:

- **addresses**. Array of string values. Required. Use the **email:** prefix in the sender address and destination address to indicate to the plug-in manager that it is an email message intended for the email plug-in.
- **subject**. String. Optional. The text of the message.
- **priority**. String. Optional. The message priority specified as one of the following:
  - **Default**
  - **Low**
  - **Normal**
  - **High**
- **senderAddress**. String. Use the **email:** prefix.
- **receiptRequest**: a JSON object. Provide values for each of the following parameters which define this object:
  - **correlator**. String. Required. Used to correlate the receipt with the initial message.

    If the callback reference **correlator** is defined in the receiptRequest attribute of the Send Message request, Services Gatekeeper will invoke notifyMessage Delivery Receipt and discard the delivery status information.
  - **endpoint**. String. The endpoint address (URI) to which the receipt must be delivered, a Bayeux protocol channel name that is the client application's application instance account ID.
  - **interfaceName**. String. Required. A description provided to identify the type of receipt.

The Message Part Content for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "addresses": ["URI"],
  "priority": "Default|Low|Normal|High",
  "receiptRequest": {
    "correlator": "String",
    "endpoint": "URI",
    "interfaceName": "String"
  },
  "senderAddress": "String",
  "subject": "String"
}
```

Response Header

The Location header field contains the URI of the publish/subscribe server as:

`http://`*`host`*`:`*`port`*`/rest/multimedia_messaging/delivery-status/`*`result`*

where, *result* is the string identifier for the request that is returned in the response body.

If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The body of the response contains the request identifier returned in the **Location** header field of the response message as value for **result**.

The response body for this operation is represented by the following JSON data structure, where the value part of the name/value pair indicates its data type:

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

Examples

Example 9–1 Send Message Requests

This section contains two examples. The first example shows a POST method with a message.

```
POST /rest/multimedia_messaging/messages HTTP/1.1
X-Session-ID: app:-7603991555266551180
Authorization: Basic ZG9tY1uX3VzZXI6ZG9tYWluX3VzZXI=
X-Param-Keys: ContentInFirstAttachment
X-Param-Values: false
User-Agent: Jakarta Commons-HttpClient/3.0
Host: localhost:8001
Content-Length: 253
Content-Type: application/json

{

"addresses":["email:sam@mycompany.com","email:john@mycompany.com","email:tom@mycom
pany.com"],
  "subject":"Meeting starts at 2 p.m.",
  "priority":null,
  "senderAddress":""email:bob@mycompany.com",
  "charging":null
  "receiptRequest":
         {
           "correlator":"981234",
           "endpoint":"http://10.182.100.229:13444/jaxws/MessageNotification",
           "interfaceName":"interfaceName",
         }
 }
```

The second example shows a POST method with a multipart message. Note the use of header fields to describe the content and the boundary attribute to distinguish the message parts.

```
POST /rest/multimedia_messaging/messages HTTP/1.1
X-Session-ID: app: -123456789012346789
Authorization: Basic ZG9tY1uX3VzZXI6ZG9tYWluX3VzZXI=
X-Param-Keys: ContentInFirstAttachment
X-Param-Values: true
User-Agent: Jakarta Commons-HttpClient/3.0
Host: localhost:8001
Content-Length: 1215
Content-Type: multipart/form-data; boundary=kboiiFPAakDPYKeY7hBAW9I5c0rT48

--kboiiFPAakDPYKeY7hBAW9I5c0rT48
Content-Disposition: form-data; name="messagePart"
Content-Type: application/json; charset=US-ASCII
```

```
Content-Transfer-Encoding: 8bit

{

"addresses":["email:sam@mycompany.com","email:john@mycompany.com","email:tom@mycom
pany.com"],
  "subject":"Hello World",
  "priority":null,
  "senderAddress":"email:bob@mycompany.com",
  "charging":null,
  "receiptRequest":
         {
           "correlator":"981234",
           "endpoint":"http://10.182.100.229:13444/jaxws/MessageNotification",
           "interfaceName":"interfaceName"
         }
 }
--kboiiFPAakDPYKeY7hBAW9I5c0rT48
Content-Disposition: form-data; name="Attachment-txt-1"
Content-Type: text/plain; charset=US-ASCII
Content-Transfer-Encoding: 8bit

Oracle Communications Services Gatekeeper is a powerful, flexible, secure
interface which provides a simple way for application developers to include
telephony-based functionality in their software applications and guarantee the
security, stability, and performance required by network operators and demanded by
their subscribers.
--kboiiFPAakDPYKeY7hBAW9I5c0rT48
```

Get Received Messages

The Get Received Messages operation retrieves an array of network-triggered messages that have arrived at Services Gatekeeper.

The request body for Get Received Messages contains the registration identifier necessary to retrieve the required email messages intended for the application and, optionally, it may also contain a priority for the retrieval. The registration identifier should have been set up with the off-line provisioning step completed earlier.

If Get Received Messages is successful, the response body will contain the data about the message. If the content of the message is pure ASCII, the response body contains the message. Otherwise the response body contains an identifier that is used to fetch the actual message.

Authorization

Basic

HTTP Method

`POST`

URI

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

where *host* and *port* are the host name and port name of the Services Gatekeeper access tier.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Get Received Messages operation accepts the following parameters:

- **registrationIdentifier**. String. Required. The value previously set up to enable the application to receive notification of email reception according to specified criteria.
- **priority**. String. Optional. The message priority specified as one of the following:
  - **Default**
  - **Low**
  - **Normal**
  - **High**

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "registrationIdentifier": "String",
  "priority": "Default|Low|Normal|High"
}
```

Response header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The response body contains an array of structures as the value for **result**. Each element in the array contains values for the following parameters.

- **messageServiceActivationNumber**. String. The number associated with the invoked Message service, that is the destination address used to send the message.
- **priority**. String. The message priority value from the request.
- **senderAddress**. String.
- **dateTime**. String. The start time for the call in ISO 8601 extended format `yyyy-mm-ddThh-mm-ss`.
- **message**. String. Present if the text of the message is purely ASCII text.
- **messageIdentifier**. String. An identifier used to fetch the message, when it is not purely ASCII text. This value is used in the URI of the request for the "Get Message" operation.
- **subject**. String. The subject line of the email message.

The response body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{"result": [{
  "messageServiceActivationNumber": "String",
  "priority": "Default|Low|Normal|High",
  "senderAddress": "URI",
  "dateTime": "Calendar",
  "message": "String",
  "messageIdentifier": "String",
  "subject": "String"
}]}
```

Get Message

The Get Message operation fetches a network-triggered message for the application from Services Gatekeeper.

To get a message, provide the appropriate message identifier in the URI for the GET method.

The actual message is returned as an attachment.

Authorization

Basic

HTTP Method

`GET`

URI

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

where:

- *host* and *port* are the host name and port name of the Services Gatekeeper access tier.
- *messageRefIdentifier* is the value for *messageIdentifier* in:
  - the response body of the "Get Received Messages" operation
  - **notifyMessageReception** data object. See "Notification Data Object for Message Reception (notifyMessageReception)"

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

There is no request body.

Response Header

The MIME-type for the Content-Type header field is **multipart/mixed**.

If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body.

Get Message Delivery Status

The Get Message Delivery Status operation retrieves the delivery status of a previously sent message.

This operation is valid only if the **correlator** callback reference was not defined in the **receiptRequest** attribute of the initial request to send the email message. In such a scenario, Services Gatekeeper stores the delivery status for a configurable period of time. (If the callback reference correlator is defined in the **receiptRequest** attribute of the initial request to send the email message, Services Gatekeeper will invoke the Notify Message Delivery Receipt operation and discard the delivery status information.)

The request header for the Get Message Delivery Status operation contains the request identifier from **result** object of the initial request.

If Get Message Delivery Status operation is successful, the response body will contain the SIP-formatted URI from the address field in the Send Message request and corresponding delivery status for the message.

Authorization

Basic

HTTP Method

`GET`

URI

`http://host:port/rest/emailmultimedia_messaging/delivery_status/${requestIdentifier}`

where:

- *host* and *port* are the host name and port name of the Services Gatekeeper access tier.
- *requestIdentifier* is a string returned in the **result** object of the initial Send Message request.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The response body contains an array of structures as the value for **result**. Each element in the array contains values for the following parameters.

- **address**. String. The SIP-formatted URI to which the initial message was sent.
- **deliveryStatus**. Enumeration value. Table 9–1 lists the possible status:

Table 9–1 Enumeration Values for Delivery Status

Value	Description
DeliveredToNetwork	Successful delivery to network. In case of concatenated messages, only when all the email-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.
DeliveryNotificationNotSupported	Unable to provide delivery receipt notification.

The response body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{"result": [{
  "address": "URI",
  "deliveryStatus":
"DeliveredToTerminal|DeliveryUncertain|DeliveryImpossible|MessageWaiting|Delivered
ToNetwork"
}]}
```

Start Message Notification

The Start Message Notification operation starts a notification for the application.

To set up an email message notification, provide the criteria which will trigger notifications and a reference object for the delivery of the notifications. The criteria can be a string which, when matched, could be the notification of an email received or of a delivery receipt. The reference object (also a JSON object) contains the correlator for the notification, the endpoint address (a specific Bayeux channel name) to which the call direction notifications must be sent and, optionally, the interface name (a string to identify the notification).

If the Start Message Notification request is successful:

- The response header will contain the URI of the publish/subscribe server.
- A data object associated with the result of the short message operation will be sent to the endpoint address specified in the request body. This data object will contain the appropriate notification (that the message was received or a delivery receipt for the call).

When the application receives such a response, it must access the endpoint address to retrieve the specific call event notifications.

Authorization

Basic

HTTP Method

`PUT`

URI

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

where *host* and *port* are the host name and port name of the Services Gatekeeper access tier.

Request Header

The request header depends on the type of message:

- The MIME-type for the Content-Type header field is **application/json**.
- The X-Parameter **Password** to indicate the credential of the email service activation number. The value should be encrypted by AES (Advanced Encryption Standard) or 3DES (Triple Data Encryption Standard) algorithm.
- The X-Parameter **SizeLimit** to indicate the maximum total size (in kilobyte) of an email message attachment accepted by Services Gatekeeper.

Request Body

The request body is a nested JSON object containing the following parameters:

- **reference**. JSON object. Required. Use this object to provide the following information about the endpoint that is to receive the notification:
  - **correlator**. String. Required. The correlator used to identify the notification.

  - **endpoint**. String. Required. The URI which represents the endpoint address to which the notification should be delivered. This string should be a Bayeux protocol channel name that begins with **/bayeux/***appInstanceID* where *appInstanceID* is the client application's application instance account ID.

    For more information on application instances, see the discussion on creating and maintaining service provider and application accounts in Oracle Communications Services Gatekeeper *Accounts and SLAs Guide*.

  - **interfaceName**. String. Required. A descriptive string to identify the type of notification.

- **messageServiceActivationNumber**. String. Required. The number associated with the invoked Message service, that is the destination address used to send the message
- **sizeLimit**. Integer. The maximum size for the email allowed in Services Gatekeeper.

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "messageServiceActivationNumber": "URI",
  "reference": {
    "correlator": "String",
    "endpoint": "URI",
    "interfaceName": "String"
  },
  "sizeLimit": "Integer"
}
```

Response Header

The Location header field contains the URI of the publish/subscribe server:

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

If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body. The appropriate notifications (described below) are sent to the **endpoint** address provided by the application in the request body of this operation.

Notification Data Object for Message Reception (notifyMessageReception)

When an email message is successfully retrieved by the called party and the value for **criteria** in the request body is **notifyMessageReception**, Services Gatekeeper sends a nested JSON data object to the **endpoint** address.

This nested JSON object contains the following as the value of the attribute name **notifyMessageReception**:

- **correlator**. String. The correlator used to identify the notification.
- **message**. Nested JSON object. It contains the following:
  - **message**. String. The contents of the email message.

- **senderAddress**. String.
- **messageServiceActivationNumber**. String. The number associated with the invoked Message service, that is the destination address used to send the message.
- **dateTime**. String. The time at which the message was sent in ISO 8601 extended format `yyyy-mm-ddThh-mm-ss`

The notification data object delivered to the **endpoint** address when the criteria is **notifyMessageReception** is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{"notifyMessageReception": {
  "correlator": "String",
  "message": {
    "messageServiceActivationNumber": "String",
    "priority": "Default|Low|Normal|High",
    "senderAddress": "URI",
    "dateTime": "Calendar",
    "message": "String",
    "messageIdentifier": "String",
    "subject": "String"
  }
}}
```

Notification Data Object for Email Delivery Receipt (notifyMessageDeliveryReceipt)

When an email message is successfully retrieved by the called party and the value for **criteria** in the request body is **notifyMessageDeliveryReceipt**, Services Gatekeeper sends a nested JSON data object to the **endpoint** address.

This nested JSON object contains the following as the value of the attribute name **notifyMessageDeliveryReceipt**:

- **correlator**. String. The correlator used to identify the notification.
- **deliveryStatus**. Enumeration value. Table 9–2 lists the possible statuses:

Table 9–2 Enumeration Values for Delivery Status

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

The notification data object delivered to the **endpoint** address when the criteria is **notifyMessageDeliveryReceipt** is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{"notifyMessageDeliveryReceipt": {
  "correlator": "String",
  "deliveryStatus": {
    "address": "URI",
    "deliveryStatus":
"DeliveredToTerminal|DeliveryUncertain|DeliveryImpossible|MessageWaiting|Delivered
ToNetwork"
  }
}}
```

Stop Message Notification

The Stop Message Notification operation terminates a previously set up email message notification for the application.

To stop a previously set up email notification, provide the correlator for the notification passed earlier in the Start Message Notification request.

There is no request or response body for the Stop Message Notification operation. If the request fails, the body of the error response will contain the identifier for the notification and the type of exception.

Authorization

Basic

HTTP Method

`DELETE`

URI

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

where:

- where *host* and *port* are the host name and port name of the Services Gatekeeper access tier.
- *correlator* is the correlator for the notification provided in the **reference** object of the initial Start Message Notification operation.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body.

Examples

Example 9–2 Stop Message Notification Request

```
DELETE /rest/multimedia_messaging/notification/6789 HTTP/1.1
X-Session-ID: app:4130997928482260925
Authorization: Basic ZG9tYWluX3VzZXI6ZG9tYWluX3VzZXI=
User-Agent: Jakarta Commons-HttpClient/3.0
Host:10.182.100.218.8002
```

Example 9–3 Stop Message Notification Response

```
HTTP/1.1 204 No Content
```

```
Connection: close
Date: Thu, 04 Nov 2101 09:59:05 GMT
Content-Length: 0
X-Plugin-Param-Keys:
X-Plugin-Param-Values:
X-Powered-By: Servlet/2.5 JSP/2.1
```

10

Terminal Status

This chapter describes the operations in the Terminal Status interface of the RESTful Web Services provided in Oracle Communications Services Gatekeeper.

About the Terminal Status Interface

Applications use the RESTful Terminal Status interface to get the status of an individual terminal or a group of terminals, or to receive notifications of changes in the status of a terminal.

REST Service Descriptions Available at Run-time

When the Administration Server for your Services Gatekeeper domain is in the running state, the REST service descriptions of these operations can be found at

`http://host:port/rest/terminal_status/index.html`

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Get Status

The Get Status operation retrieves the status of a single terminal.

To retrieve the status of a specific terminal, provide its URI as the address value of the query object in the Request-URI of the GET method. The interface returns an error if the query object contains more than one address.

If the Get Status operation is successful, the response body contains a JSON data object indicating whether the specific terminal is reachable, unreachable or busy.

Authorization

Basic

HTTP Method

`GET`

URI

`http://host:port/rest/terminal_status/status?query=${query}`

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *${query}* contains the URI address for the terminal as the string value for URI in the data object:

  ```
  {"address":"tel:123"}
  ```

For example, the operation to get the status for a terminal identified as "tel:123" is:

```
GET /rest/terminal_status/status?query="%7B%22address%22%3A%22tel%3A123%22%7D"
HTTP/1.1
```

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The status of the specific terminal is returned in the body of the response as the value of the **result** attribute:

- **Reachable**
- **unReachable**

- **Busy**

The response body for this operation is represented by the following JSON data structure, where the value part of the name/value pair indicates its data type:

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

Examples

Example 10–1 Get Status Request

```
GET /rest/terminal_status/status?query="%7B%22address%22%3A522tel%3A123%22%7D"
HTTP/1.1
X-Session-ID: app:4130997928482260925
Authorization: Basic ZG9tYWluX3VzZXI6ZG9tYWluX3VzZXI=
X-Param-Keys:
X-Param-Values:
User-Agent: Jakarta Commons-HttpClient/3.0
Host:10.182.100.218.8002
```

Example 10–2 Get Status Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Nov 2101 08:12:43 GMT
Content-Length: 22
Content-Type: application/json
X-Plugin-Param-Keys:
X-Plugin-Param-Values:
X-Powered-By: Servlet/2.5 JSP/2.1

{"result":"Reachable"}
```

Get Status For Group

The Get Status for Group operation retrieves the status for a group of terminals.

To retrieve the status of a specific terminals, provide their URIs as the address values of the query object in the Request-URI of the GET method.

If the Get Status for Group operation is successful, the response body contains a JSON data object indicating the status for each terminal (whether the specific terminal is reachable, unreachable or busy).

Authorization

Basic

HTTP Method

`GET`

URI

`http://host:port/rest/terminal_status/status?queryForGroup=${query}`

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *${query}* contains the URIs for the terminals as an array of string values in the data object:

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

For example,

```
GET /rest/terminal_
status/status?queryForGroup="%7B%22address%22%3A%5B%22tel%3A123%22%2C%22tel%3A4562
2tel%3A789%22%5D%7D" HTTP/1.1
```

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The response body contains an array of structures as the value for **result**. Each element in the array contains the status for one of the terminals whose address was specified in the Request-URI for **status?queryForGroup**. The following parameters make up the individual terminal's status:

- **address**. String. The URI of a terminal whose status is required.
- **reportStatus**. String. The status of the terminal. It can be one of the following:
  - **Retrieved**. The terminal's status is available. It is provided in this object as the current status of the terminal.
  - **Not Retrieved**. The terminal's status is not available.
  - **Error**: There was an error in the attempt to get the status for this terminal. The error data is provided in this object.
- **currentStatus**. String. This parameter will be present if the value for **reportStatus** is **Retrieved**. The current status of the terminal as one of the following:
  - **Reachable**
  - **Unreachable**
  - **Busy**
- **errorInformation**. a JSON object. This object will be present if the value for `reportStatus` is **Error**. It will contain the following error information about the terminal:
  - **messageID**. String. The error message ID.
  - **text**. String. The text for the error message.
  - **variables**. Array of string values. An Array of variables to substitute into text.

The response body for this operation is represented by the following JSON data structure, where the value part of the name/value pair indicates its data type:

```
{"result": [{
  "address": "URI",
  "reportStatus": "Retrieved|NotRetrieved|Error",
  "currentStatus": "Reachable|Unreachable|Busy",
  "errorInformation": {
    "messageId": "String",
    "text": "String",
    "variables": ["String"]
  }
}]}
```

Examples

Example 10–3 Get Status for Group Request

```
GET /rest/terminal_
status/status?queryForGroup="%7B%22address%22%3A%5B%22tel%3A123%22%2C%22tel%3A4562
2tel%3A789%22%5D%7D" HTTP/1.1
X-Session-ID: app:4130997928482260925
Authorization: Basic ZG9tYWluX3VzZXI6ZG9tYWluX3VzZXI=
X-Param-Keys:
X-Param-Values:
User-Agent: Jakarta Commons-HttpClient/3.0
Host:10.182.100.218.8002
```

Example 10–4 Get Status for Group Response

```
HTTP/1.1 200 OK
```

```
Date: Thu, 04 Nov 2101 08:12:43 GMT
Content-Length: 22
Content-Type: application/json
X-Plugin-Param-Keys:
X-Plugin-Param-Values:
X-Powered-By: Servlet/2.5 JSP/2.1

{"result":
       [{"address":"tel:123",
           "errorInformation":null,
           "reportStatus":"Retrieved",
           "currentStatus":"Reachable"
          },
          {"address":"tel:456",
           "errorInformation":null,
           "reportStatus":"Retrieved",
           "currentStatus":"Reachable"
          },
          {"address":"tel:789",
           "errorInformation":null,
           "reportStatus":"Retrieved",
           "currentStatus":"Reachable"
          }]
}
```

Start Notification

The Start Notification operation starts requesting for notifications on the status of a specified terminal according to a specified frequency and for a specified limit of notifications over a specified time duration.

To set up for such notifications, provide the SIP-formatted URI of the terminal address for which the application must receive notifications, the criteria which will trigger notifications and a reference object for the delivery of the notifications. The reference object (also a JSON object) contains the **correlator** for the notification, the **endpoint** address to which the notifications must be sent and, optionally, the interface name (a string to identify the notification). Additionally, you can specify the request frequency, the total number of notifications and the duration for the notification and whether the check must start immediately or not.

If the Start Notification request is successful, the **endpoint** address specified in the request body will receive a notification when:

- The current status for the terminal status has been retrieved.
- The notification limit or the specified duration has been reached.
- An error has been encountered in obtaining the status.

Authorization

Basic

HTTP Method

`PUT`

URI

`http://`*`host`*`:`*`port`*`/rest/terminal_status/status-notification`

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Start Notification operation is a nested JSON object. It contains an array of addresses for the terminals in whose status the application is interested, and the parameters for the status check provided by the following:

- **address**. String. Required. The URI of a terminal to monitor.
- **checkImmediate**. String. Required. Boolean value ("true" or "false"). Set to:
  - **True**: The application requires the status notification to start immediately and thereafter, as often as required.
  - **False**: The application requires the status notification to start at the end of the time period.

- **criteria**. A set of strings. Required. The required criteria about the terminal. It can be one or all of the following:
  - **Reachable**
  - **UnReachable**
  - **Busy**
- **frequency**. JSON object. Required. This object specifies the frequency for checking the status with the following:
  - **metric**. String. Required. The unit of time for the check specified as **Millisecond**, **Second**, **Minute**, **Hour**, **Day**, **Week**, **Month**, or **Year**.
  - **units**. Integer. Required. The number of times the check must be made within the specified metric.
- **reference**. a JSON object. Required. Use this object to provide the following information about the endpoint that is to receive the notification:
  - **correlator**. String. Required. The correlator used to identify the notification.
  - **endpoint**. String. Required. The URI which represents the endpoint address to which the notification should be delivered. This string should be a Bayeux protocol channel name that begins with **`/bayeux/`***`appInstanceID`* where *appInstanceID* is the client application's application instance account ID.

    For more information on application instances, see the discussion on creating and maintaining service provider and application accounts in *Oracle Communications Services Gatekeeper Accounts and SLAs Guide*.
  - **interfaceName**. String. Required. A descriptive string to identify the type of notification.
- **count**. Integer. Required. The maximum number of notifications to be sent to the application. If this number is reached the **statusEnd** notification is delivered to the **endpoint** address.
- **duration**. JSON object. Required. This object specifies the total duration before for the count with the following:
  - **metric**. String. Required. The unit of time for the notifications specified as **Millisecond**, **Second**, **Minute**, **Hour**, **Day**, **Week**, **Month**, or **Year**.
  - **units**. Integer. Required. The duration in the specified metric.

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "addresses": ["URI"],
  "checkImmediate": "Boolean",
  "criteria": ["Reachable|Unreachable|Busy"],
  "frequency": {
    "metric": "Millisecond|Second|Minute|Hour|Day|Week|Month|Year",
    "units": "Integer"
  },
  "reference": {
    "correlator": "String",
    "endpoint": "URI",
    "interfaceName": "String"
  },
  "count": "Integer",
  "duration": {
```

```
    "metric": "Millisecond|Second|Minute|Hour|Day|Week|Month|Year",
    "units": "Integer"
  }
}
```

Response Header

The Location header field contains the URI of the publish/subscribe server:

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

If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body. The appropriate notifications (described below) are sent to the **endpoint** address provided by the application in the request body of this operation.

Notification Of Current Terminal Status (statusNotification)

When there is a match for the **criteria** value specified in the request body, Services Gatekeeper sends a nested JSON data object to the **endpoint** address.

This nested JSON object contains the following as the value of the attribute name **statusNotification**:

- **correlator**. String. The correlator used to identify the notification and provided in the request body of the Start Notification operation.
- **terminalStatus**. An array of JSON objects. Each element in the array contains the current status of the terminal (based on the criteria) being monitored and defined by the following:
  - **address**. String. The URI of the terminal being monitored.
  - **currentStatus**. String. The current status which can be **Reachable**, **unReachable**, or **Busy**.

The notification data object delivered to the **endpoint** address when the criteria is **statusNotification** is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{"statusNotification": {
  "correlator": "String",
  "terminalStatus": [{
    "address": "URI",
    "currentStatus": "Reachable|Unreachable|Busy"
  }]
}}
```

Notification of Error in Retrieving Terminal Status (statusError)

When there is an error in retrieving the status of a terminal, Services Gatekeeper sends a nested JSON data object to the **endpoint** address.

This nested JSON object contains the following as the value of the attribute name **statusError**:

- **correlator**. String. The correlator used to identify the notification and provided in the request body of the Start Notification operation.
- **reason**: JSON object. The explanation of the error specified by the following:
  - **messageID**. String. The error message identifier.
  - **text**. String. The error message description.
  - **variables**. An array of strings. The text to string variables.
- **address**. String. The URI of a terminal to monitor.

The notification data object delivered to the **endpoint** address when the criteria is **statusError** is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{"statusError": {
  "correlator": "String",
  "reason": {
    "messageId": "String",
    "text": "String",
    "variables": ["String"]
  },
  "address": "URI"
}}
```

Notification Signalling End to Monitoring Terminal (statusEnd)

When the notification count specified as the value for **count** is reached or when the duration specified for **units** in the **duration** object is reached, Services Gatekeeper sends a nested JSON data object to the **endpoint** address.

This nested JSON object contains the following as the value of the attribute name **statusEnd**:

- **correlator**. String. The correlator used to identify the notification and provided in the request body of the Start Notification operation.

  This notification is not delivered in the case of an error, or if the application ended the notification using **endNotification**.

The notification data object delivered to the **endpoint** address when the criteria is **statusEnd** is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

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

Examples

Example 10–5 Start Notification Request

```
PUT /rest/terminal_status/status-notification HTTP/1.1
X-Session-ID: app:4130997928482260925
Authorization: Basic ZG9tYWluX3VzZXI6ZG9tYWluX3VzZXI=
X-Param-Keys:
X-Param-Values:
User-Agent: Jakarta Commons-HttpClient/3.0
Host:10.182.100.218.8002
```

```
Content-Length: 289
Content-Type: application/json

{
 "duration":null,
 "count":["Reachable","Busy","Unreachable"],
 "frequency":
        {
         "metric":"Second",
         "units":"5"
        },
 "checkImmediate":"true",
 "addresses":["tel:123","tel:456","tel:789"],
 "reference":
        {"interfaceName":"interfaceName",
         "correlator":"6789",
         "endpoint":"/bayeux/domain-user/ts"
        }
}
```

Example 10–6 Start Notification Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Nov 2101 09:59:05 GMT
Content-Length: 0
X-Plugin-Param-Keys:
X-Plugin-Param-Values:
TerminalStatus: http://10.182.100.218:8002/rest/terminal_status/notifications
X-Powered-By: Servlet/2.5 JSP/2.1
```

End Notification

The End Notification operation terminates an application's previously set up notification to get the status of a specified terminal.

To stop a previously set up Start Notification, provide the correlator for the notification passed earlier in the Start Notification request.

There is no request or response body for the End Notification operation. If the request fails, the body of the error response will contain the identifier for the notification and the type of exception.

Authorization

Basic

HTTP Method

`DELETE`

URI

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

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *correlator* is the correlator for the notification provided in the **reference** object of the initial Start Notification operation request.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body.

Examples

Example 10–7 End Notification Request

```
DELETE /rest/terminal_status/status-notification/6789 HTTP/1.1
X-Session-ID: app:4130997928482260925
Authorization: Basic ZG9tYWluX3VzZXI6ZG9tYWluX3VzZXI=
X-Param-Keys:
X-Param-Values:
User-Agent: Jakarta Commons-HttpClient/3.0
Host:10.182.100.218.8002
```

Example 10–8 End Notification Response

```
HTTP/1.1 204 No Content
Connection: close
Date: Thu, 04 Nov 2101 09:59:05 GMT
Content-Length: 0
X-Plugin-Param-Keys:
X-Plugin-Param-Values:
X-Powered-By: Servlet/2.5 JSP/2.1
```

11

Terminal Location

This chapter describes the operations in the Terminal Location interface of the RESTful Web Services provided in Oracle Communications Services Gatekeeper.

About the Terminal Location Interface

Applications use the RESTful Terminal Location interface 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.

REST Service Descriptions Available at Run-time

When the Administration Server for your Services Gatekeeper domain is in the running state, the REST service descriptions of these operations can be found at

`http://host:port/rest/terminal_location/index.html`

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Get Location

The Get Location operation retrieves the location of a single terminal. The interface will return an error if the query object contains more than one address.

To retrieve the location of a specific terminal, provide its URI as the address value of the query object in the Request-URI of the GET method.

If the Get Location operation is successful, the response body contains a nested JSON data object containing the physical co-ordinates of each of the specific terminal and the date and time for when such data was last collected.

Authorization

Basic

HTTP Method

`GET`

URI

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

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *${query}* is a nested JSON data object.

The parameters accepted by *query* are:

- **acceptableAccuracy**: Integer. Required. The range that the application considers useful. If the location cannot be determined within this range, the application would prefer not to receive this information.
- **address**. String. Required. The address of the terminal whose location is required, as a URI.
- **requestedAccuracy:** Integer. Required. 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:** String. Required. Enumerated value denoting the priority of response time versus accuracy. Table 11–1 lists the possible values:

Table 11–1 Enumeration Values for Tolerance Attribute

Value	Description
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 (for example, to be available for a later request.

Table 11–1 (Cont.) Enumeration Values for Tolerance Attribute

Value	Description
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.

- **maximumAge**: JSON object. Optional. The maximum acceptable age of the location information. This object is defined by the following:
  - **metric**. String. The unit of time. Required if the **maximumAge** object is present in the body of the query. Possible entries are **Millisecond**, **Second**, **Minute**, **Hour**, **Day**, **Week**, **Month**, or **Year**.
  - **units**: Integer. Required if the **maximumAge** object is present in the body of the query. The number of units for the metric.
- **responseTime**: JSON object. Optional. The maximum response time that the application will accept. This object is defined by the following:
  - **metric**. String. The unit of time. Required if the **responseTime** object is present in the body of the query. Possible entries are **Millisecond**, **Second**, **Minute**, **Hour**, **Day**, **Week**, **Month**, or **Year**.
  - **units**: Integer. Required if the **maximumAge** object is present in the body of the query. The number of units for the metric.

The *${query}* object in the URI is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "acceptableAccuracy": "Integer",
  "address": "URI",
  "requestedAccuracy": "Integer",
  "tolerance": "NoDelay|LowDelay|DelayTolerant",
  "maximumAge": {
    "metric": "Millisecond|Second|Minute|Hour|Day|Week|Month|Year",
    "units": "Integer"
  },
  "responseTime": {
    "metric": "Millisecond|Second|Minute|Hour|Day|Week|Month|Year",
    "units": "Integer"
  }
}
```

Example 11–1 Request URI for Get Location

```
GET /rest/terminal_
location/location?query=%7B%22aceptableAccuracy%22%3A%22100%%22%2C%22address%22%3A
%22tel%3A1234%22%2C%22requestAccuracy%22%3A%22100%%22%2C
%22tolerance%22%3A%22NoDelay%22%2C%22maximumAge%22%3Anull%2C%22responseTime%22%3An
ull%7D HTTP/1.1
```

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The location of the specific terminal is returned in the body of the response as the value of the **result** JSON object. The parameters in this object are:

- **accuracy**: Integer.
- **latitude**: Number (floating point).
- **longitude**: Number (floating point).
- **timestamp**. String. The date and time when the terminal's geographical co-ordinates were collected, given in ISO 8601 extended format `yyyy-mm-ddThh-mm-ss`.
- **altitude**: Number (floating point).

The response body for this operation is represented by the following JSON data structure, where the value part of the name/value pair indicates its data type:

```
{"result": {
  "accuracy": "Integer",
  "latitude": "Float",
  "longitude": "Float",
  "timestamp": "Calendar",
  "altitude": "Float"
}}
```

Example 11–2 Response Body for Get Location

```
{
   "result":
       {
        "accuracy":"10",
        "latitude":"37.78843",
        "longitude":"-122.4374",
        "timestamp":"2010-11-05T21:56:21+08:00"
        }
}
```

Examples

Example 11–3 Get Location Request

```
GET /rest/terminal_
location/location?query=%7B%22aceptableAccuracy%22%3A%22100%%22%2C%22address%22%3A
%22tel%3A1234%22%2C%22requestAccuracy%22%3A%22100%%22%2C
%22tolerance%22%3A%22NoDelay%22%2C%22maximumAge%22%3Anull%2C%22responseTime%22%3An
ull%7D HTTP/1.1
X-Session-ID: app:5198750923966743997
Authorization: Basic ZG9tYWluX3VzZXI6ZG9tYWluX3VzZXI=
X-Param-Keys:
```

```
X-Param-Values:
User-Agent: Jakarta Commons-HttpClient/3.0
Host:10.182.100.218.8002
```

Example 11–4 Get Location Response

```
HTTP/1.1 200 OK
Date: Thu, 05 Nov 2101 05:52:41 GMT
Content-Length: 131
Content-Type: application/json
X-Plugin-Param-Keys:
X-Plugin-Param-Values:
X-Powered-By: Servlet/2.5 JSP/2.1

{
   "result":
       {
        "accuracy":"10",
        "latitude":"37.78843",
        "longitude":"-122.4374",
        "timestamp":"2010-11-05T21:56:21+08:00"
        }
}
```

Get Location for Group

The Get Location for Group operation retrieves the location information for a group of terminals.

To retrieve the location information of a specific terminals, provide their URIs as the address values of the query object in the Request-URI of the GET method.

If the Get Location for Group operation is successful, the response body contains a JSON data object indicating the physical location for each terminal (whether the specific terminal is reachable, unreachable or busy).

Authorization

Basic

HTTP Method

`GET`

URI

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

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *${queryForGroup}* is a nested JSON data object.

The parameters accepted by *queryForGroup* are:

- **acceptableAccuracy**: Integer. Required. The range that the application considers useful. If the location cannot be determined within this range, the application would prefer not to receive this information.
- **addresses**: Array of Strings. Required. The addresses of the terminals whose location is required, as URI.
- **requestedAccuracy:** Integer. Required. 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:** String. Required. Enumerated value denoting the priority of response time versus accuracy. Table 11–2 lists the possible values:

Table 11–2 Enumeration Values for Tolerance Attribute

Value	Description
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 (for example, to be available for a later request.

Table 11–2 (Cont.) Enumeration Values for Tolerance Attribute

Value	Description
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.

- **maximumAge**: JSON object. Optional. The maximum acceptable age of the location information. This object is defined by the following:
  - **metric**. String. The unit of time. Required if the `maximumAge` object is present in the body of the query. Possible entries are **Millisecond**, **Second**, **Minute**, **Hour**, **Day**, **Week**, **Month**, or **Year**.
  - **units**: Integer. Required if the **maximumAge** object is present in the body of the query. The number of units for the metric.
- **responseTime**: JSON object. Optional. The maximum response time that the application will accept. This object is defined by the following:
  - **metric**. String. The unit of time. Required if the **responseTime** object is present in the body of the query. Possible entries are **Millisecond**, **Second**, **Minute**, **Hour**, **Day**, **Week**, **Month**, or **Year**.
  - **units**: Integer. Required if the **maximumAge** object is present in the body of the query. The number of units for the metric.

The *${queryForGroup}* object in the URI is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "acceptableAccuracy": "Integer",
  "addresses": ["URI"],
  "requestedAccuracy": "Integer",
  "tolerance": "NoDelay|LowDelay|DelayTolerant",
  "maximumAge": {
    "metric": "Millisecond|Second|Minute|Hour|Day|Week|Month|Year",
    "units": "Integer"
  },
  "responseTime": {
    "metric": "Millisecond|Second|Minute|Hour|Day|Week|Month|Year",
    "units": "Integer"
  }
}
```

Example 11–5 Request URI for Get Location for Group

```
GET /rest/terminal_
location/location?queryForGroup=%7B%22aceptableAccuracy%22%3A%22100%%22%2C%22addre
sses%22%3A%%5B22tel%3A1234%22%2C%22tel%3A123%22%5D%2C%22requestAccuracy%22%3A%2210
0%%22%2C
%22tolerance%22%3A%22NoDelay%22%2C%22maximumAge%22%3Anull%2C%22responseTime%22%3An
ull%7D HTTP/1.1
```

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The response body is the attribute **result** whose value consists of an array of JSON objects. Each element in the array contain the report status, the location information (if any) and the error information (if any) for the terminals whose addresses were specified in the Request-URI for **status?queryForGroup**.

The following parameters make up the information associated with an individual terminal's location:

- **address**. String. The address of a terminal whose location is required, as a URI.
- **reportStatus**. String. The status of the terminal. It can be one of the following:
  - **Retrieved**: The terminal's location information is available. It is provided in this object as the current location of the terminal.
  - **Not Retrieved**: The terminal's location information is not available.
  - **Error**: There was an error in the attempt to get the location information for this terminal. The error data is provided in this object.
- **currentLocation**. String. This parameter will be present if the value for **reportStatus** is **Retrieved**. The current location of the terminal as one of the following:
  - **accuracy**: Integer.
  - **latitude**: Number (floating point).
  - **longitude**: Number (floating point).
  - **altitude**: Number (floating point).
  - **timestamp**. String. The date and time when the terminal's geographical coordinates were collected, in ISO 8601 extended format `yyyy-mm-ddThh-mm-ss`.
- **errorInformation**: JSON object. This object will be present if the value for `reportStatus` is `Error`. It will contain the following error information about the terminal:
  - **messageID**. String. The error message ID.
  - **text**. String. The text for the error message.
  - **variables**: Array of string values. Variables to substitute into text strings.

The response body for this operation is represented by the following JSON data structure, where the value part of the name/value pair indicates its data type:

```
{"result": [{
  "address": "URI",
```

```
  "reportStatus": "Retrieved|NotRetrieved|Error",
  "currentLocation": {
    "accuracy": "Integer",
    "latitude": "Float",
    "longitude": "Float",
    "timestamp": "Calendar",
    "altitude": "Float"
  },
  "errorInformation": {
    "messageId": "String",
    "text": "String",
    "variables": ["String"]
  }
}]}
```

Examples

Example 11–6 Get Location for Group Request

```
GET /rest/terminal_
location/location?queryForGroup=%7B%22aceptableAccuracy%22%3A%22100%%22%2C%22addre
sses%22%3A%%5B22tel%3A1234%22%2C%22tel%3A123%22%5D%2C%22requestAccuracy%22%3A%2210
0%%22%2C
%22tolerance%22%3A%22NoDelay%22%2C%22maximumAge%22%3Anull%2C%22responseTime%22%3An
ull%7D HTTP/1.1
X-Session-ID: app:4130997928482260925
Authorization: Basic ZG9tYWluX3VzZXI6ZG9tYWluX3VzZXI=
X-Param-Keys:
X-Param-Values:
User-Agent: Jakarta Commons-HttpClient/3.0
Host:10.182.100.218.8002
```

Example 11–7 Get Location for Group Response

```
HTTP/1.1 200 OK
Date: Thu, 04 Nov 2101 08:12:43 GMT
Content-Length: 438
Content-Type: application/json
X-Plugin-Param-Keys:
X-Plugin-Param-Values:
X-Powered-By: Servlet/2.5 JSP/2.1

{"result":
       [{"address":"tel:1234",
           "errorInformation":null,
           "reportStatus":"Retrieved",
           "currentLocation":
                    {
                     "accuracy":"10",
                     "latitude":"37.78843",
                     "longitude":"-122.4374",
                     "altitude":"0.0"
                     "timestamp":"2010-11-05T22:13:28+08:00"
          },
          {"address":"tel:123",
           "errorInformation":null,
           "reportStatus":"Retrieved",
           "currentLocation":
```

```
                                {
                                 "accuracy":"10",
                                 "latitude":"55.2776",
                                 "longitude":"7.012778",
                                 "altitude":"20.0"
                                 "timestamp":"2010-11-05T22:13:28+08:00"
                        }
                }
```

Get Terminal Distance

The Get Terminal Distance operation retrieves the distance between a specified terminal and a required location. The terminal distance is calculated in meters.

To retrieve the distance, provide the URI for the terminal and the geographic co-ordinates of the required location in the Request-URI of the GET method.

If the Get Terminal Distance operation is successful, the response body contains a JSON data object indicating the distance between a specified terminal and a required location, in meters.

Authorization

Basic

HTTP Method

`GET`

URI

`http://host:port/rest/terminal_distance/distance?query=${query}`

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *${query}* is a nested JSON data object.

The parameters accepted by the *query* attribute are:

- **address**. String. Required. The address of the terminal, as a URI.
- **latitude:** Number (floating point). Required. The latitude of the location.
- **longitude:** Number (floating point). Required. The longitude of the location

The *${query}* object in the URI is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "address": "URI",
  "latitude": "Float",
  "longitude": "Float"
}
```

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The distance between the specified terminal and the required location is returned as the value for **result** in a data object. The unit for the distance is meters and the value is an integer.

The response body for this operation is represented by the following JSON data structure, where the value part of the name/value pair indicates its data type:

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

Examples

Example 11–8 Get Terminal DIstance Request

```
GET /rest/terminal_
distance/distance?query=%7B%22address%22%3A%22tel%3A1234%22%2C%22longitude%22%3A%2
237.7707%%22%2C %22latittude%22%3A%22122.4177%227D HTTP/1.1
X-Session-ID: app:5198750923966743997
Authorization: Basic ZG9tYWluX3VzZXI6ZG9tYWluX3VzZXI=
X-Param-Keys:
X-Param-Values:
User-Agent: Jakarta Commons-HttpClient/3.0
Host:10.182.100.218.8002
```

Example 11–9 Get Terminal Distance Response

```
HTTP/1.1 200 OK
Date: Thu, 05 Nov 2101 05:52:41 GMT
Content-Length: 131
Content-Type: application/json
X-Plugin-Param-Keys:
X-Plugin-Param-Values:
X-Powered-By: Servlet/2.5 JSP/2.1

{
   "result":"9316351"
}
```

Start Geographical Notification

The Start Geographical Notification operation provides notifications based on whether terminals enter or leave a specified geographic area. The area to be monitored is a circle with specified latitude and longitude as the center and having a specified radius.

To set up for such notifications, provide the SIP-formatted URI of the terminal addresses for which the application must receive notifications, the criteria which will trigger notifications and a reference object for the delivery of the notifications. Additionally, you can specify the request frequency, the total number of notifications and the duration for the notification and whether the check must start immediately or not. The reference object (also a JSON object) contains the correlator for the notification, the endpoint address (a specific Bayeux channel name) to which the notifications must be sent and, optionally, the interface name (a string to identify the notification).

If the request for the Start Geographical Notification operation is successful, the endpoint address specified in the request body will receive a notification when:

- The terminal location has been successfully retrieved.
- The notification limit or the specified duration has been reached.
- An error has been encountered in obtaining the location of the terminal.

Authorization

Basic

HTTP Method

`PUT`

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Start Geographical Notification operation is a nested JSON object. It accepts an array of addresses for the terminals in whose geographical information the application is interested, and the parameters for the check:

- **addresses**: Array of string values. Required. Each element in the array is a SIP-formatted address for a terminal, as a URI.
- **checkImmediate**. String. Required. Set to:
  - **True**: The application requires the geographical notification to start immediately and thereafter, as often as required.
  - **False**: The application requires the geographical notification to start at the end of the time period.

- **criteria**. String. Required. The status of the terminal. It can be one of the following:
  - **Entering**: The notification should be made when the terminal enters the area being monitored.
  - **Leaving**: The notification should be made when the terminal leaves the area being monitored.
- **frequency**: JSON object. Required. This object specifies the frequency for checking the status with the following:
  - **metric**. String. Required. The unit of time for the check specified as **Millisecond**, **Second**, **Minute**, **Hour**, **Day**, **Week**, **Month**, or **Year**.
  - **units**: Integer. Required. The number of times the check must be made within the specified metric.
- **latitude**: Number (floating point). Required. The latitude of the location which will be the center of the area under surveillance.
- **longitude**: Number (floating point). Required. The longitude of the location which will be the center of the area under surveillance.
- **radius**: Number (floating point) in meters. Required. The radius of the circle around the location (center point), in meters.
- **reference**: JSON object. Required. Use this object to provide the following information about the endpoint that is to receive the notification:
  - **correlator**. String. Required. The correlator used to identify the notification.
  - **endpoint**. String. Required. The URI which represents the endpoint address to which the notification should be delivered. This string should be a Bayeux protocol channel name that begins with **`/bayeux/`***`appInstanceID`* where *appInstanceID* is the client application's application instance account ID.

    For more information on application instances, see the discussion on creating and maintaining service provider and application accounts in *Oracle Communications Services Gatekeeper Accounts and SLAs Guide*.
  - **interfaceName**. String. Required. A descriptive string to identify the type of notification.
- **trackingAccuracy**: Number (floating point). Required. The acceptable error in the tracking, in meters.
- **count**: Integer. Optional. The maximum number of notifications to be sent to the application. If this number is reached (and the application has not ended the notification operation), the `locationEnd` notification is delivered to the `endpoint` address.
- **duration**: JSON object. Optional. This object specifies the total duration before for the count with the following:
  - **metric**. String. Required. The unit of time for the notifications specified as **Millisecond**, **Second**, **Minute**, **Hour**, **Day**, **Week**, **Month**, or **Year**.
  - **units**: Integer. Required. The duration in the specified metric. If this number is reached (and the application has not ended the notification operation), the **locationEnd** notification is delivered to the **endpoint** address.

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
```

```
  "addresses": ["URI"],
  "checkImmediate": "Boolean",
  "criteria": "Entering|Leaving",
  "frequency": {
    "metric": "Millisecond|Second|Minute|Hour|Day|Week|Month|Year",
    "units": "Integer"
  },
  "latitude": "Float",
  "longitude": "Float",
  "radius": "Float",
  "reference": {
    "correlator": "String",
    "endpoint": "URI",
    "interfaceName": "String"
  },
  "trackingAccuracy": "Float",
  "count": "Integer",
  "duration": {
    "metric": "Millisecond|Second|Minute|Hour|Day|Week|Month|Year",
    "units": "Integer"
  }
}
```

Response Header

The Location header field contains the URI of the publish/subscribe server:

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

If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body. The appropriate notifications (described below) are sent to the **endpoint** address provided by the application in the request body of this operation.

Notification When Terminal Location is Successfully Retrieved (locationNotification)

When there is a match for the **criteria** value (specified in the request body) and the terminal's location is successfully retrieved, Services Gatekeeper sends a nested JSON data object to the **endpoint** address.

This nested JSON object contains the following as the value of the attribute name **locationNotification**:

- **correlator**. String. The correlator used to identify the notification and provided in the request body of the Start Geographical Notification operation.
- **criteria**. String. The state of the terminal. It can be one of the following:
  - **Entering**: The terminal entered the area being monitored.
  - **Leaving**: The terminal left the area being monitored.
- **data**: Array of JSON objects. Each element in the array contains the current location information of a terminal being monitored and contains the following parameters:
  - **address**. String. The URI of the terminal being monitored.

- **reportStatus**. String. Retrieval status for this terminal which can be **Retrieved**, **NotRetrieved**, or **Error**. This allows for partial reports to avoid timeouts, and so on.
- **currentLocation**. String. This object will be present only if the **reportStatus** is **Retrieved**. It contains the location information for the terminal, as of the date and time specified in this object. Table 11–3 lists the attributes in this object:

Table 11–3 Attributes in the currentLocation JSON object

Attribute	Description
accuracy	Number (floating point). The accuracy error in arriving at the terminal's location, in meters.
latitude	Number (floating point). The latitude for the terminal.
longitude	Number (floating point). The longitude for the terminal.
timestamp	String. The date and time when the terminal's geographical coordinates were collected, given in ISO 9601's extended format.
altitude	Number (floating point). The altitude for the terminal.

- **errorInformation**: JSON object. This object will be present if the value for **reportStatus** is **Error**. Table 11–4 lists the attributes in this object.

Table 11–4 Attributes in the errorInformation object

Attribute	Description
messageID	String. Message identifier for the fault
text	String. The text of the message. If this string contains replacement variables, the variables entry hold the
variables	Array of string values. Optional. An array of variables to substitute into text strings.

The location notification data object delivered to the **endpoint** address is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{"locationNotification": {
  "correlator": "String",
  "data": [{
    "address": "URI",
    "reportStatus": "Retrieved|NotRetrieved|Error",
    "currentLocation": {
      "accuracy": "Integer",
      "latitude": "Float",
      "longitude": "Float",
      "timestamp": "Calendar",
      "altitude": "Float"
    },
    "errorInformation": {
      "messageId": "String",
      "text": "String",
      "variables": ["String"]
    }
  }],
  "criteria": "Entering|Leaving"
}}
```

Example 11–10 locationNotification Object Delivered to the Application

```
{
  "locationNotification":
            {
              "correlator":"6789",
              "data":
                  [{
                     "address":"tel:123",
                     "reportStatus":"Retrieved",
                     "errorInformation":null,
                     "currentLocation":
                            {
                              "accuracy":"8",
                              "latitude":"37.80",
                              "longitude":"-122.56",
                              "altitude":"90",
                              "timestamp":"2010-11-08T10:29:38"
                            }
                  }],
              "criteria":"Entering"
            }
}
```

Notification of Error in Retrieving Terminal Location (locationError)

When there is an error in retrieving the location for the specified terminal, Services Gatekeeper sends a nested JSON data object to the **endpoint** address.

The nested JSON object contains the following as the value of the attribute name **locationError**:

- **correlator**. String. The correlator used to identify the notification and provided in the request body of the Start Notification operation.
- **reason**: JSON object. The explanation of the error specified by the following:
  - **messageID**. String. The error message identifier.
  - **text**. String. The error message description.
  - **variables**. An array of string values. The array of variables to substitute into text strings.
- **address**. String. The URI of a terminal to monitor.

The location error notification data object delivered to the **endpoint** address is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{"locationError": {
  "correlator": "String",
  "reason": {
    "messageId": "String",
    "text": "String",
    "variables": ["String"]
  },
  "address": "URI"
}}
```

Notification Signalling End to Monitoring Terminal's Location (locationEnd)

When the notification count specified as the value for **count** is reached or when the duration specified for `units` in the **duration** object is reached, Services Gatekeeper sends a nested JSON data object to the **endpoint** address.

The JSON object contains the following as the value of the attribute name **locationEnd**:

- **correlator**. String. The correlator used to identify the notification and provided in the request body of the Start Notification operation.

Here is the structure of this notification:

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

> **Note:** This notification is not delivered in the case of an error, or if the application ended the notification using **endNotification**.

Examples

Example 11–11 Start Geographic Notification Request

```
PUT /rest/terminal_location/geographical-notification HTTP/1.1
X-Session-ID: app:4130997928482260925
Authorization: Basic ZG9tYWluX3VzZXI6ZG9tYWluX3VzZXI=
X-Param-Keys:
X-Param-Values:
User-Agent: Jakarta Commons-HttpClient/3.0
Host:10.182.100.218.8002
Content-Length: 366
Content-Type: application/json

{
   "addresses":["tel:123"],
   "checkImmediate:"true",
   "criteria":"Entering",
   "frequency":
        {
          "metric": "Second",
          "units": "5"
        },
   "latitude":"37.7707",
   "longitude":"-122.4177",
   "radius":"5000.0",
   "reference":
        {
          "correlator":"6789",
          "endpoint":"bayeux/app_instance_1/tl",
          "interfaceName":"interfaceName"
        },
   "trackingAccuracy":"10.0",
   "count":"5",
   "duration":
        {
          "metric":"Minutes",
          "units":"30"
        }
}
```

Example 11–12 Start Notification Response

```
HTTP/1.1 200 Created
Date: Fri, 05 Nov 2101 09:59:05 GMT
Location: http://10.182.100.218:8002/rest/terminal_location/notifications
Content-Length: 0
X-Plugin-Param-Keys:
X-Plugin-Param-Values:
X-Powered-By: Servlet/2.5 JSP/2.1
```

Start Periodic Notification

The Start Periodic Notification operation provides notifications for the locations of a set of terminals at a defined interval.

To set up for such location notifications, provide the SIP-formatted URI of the terminal addresses for which the application must receive notifications, the request frequency, the total number of notifications and the duration for the notification and whether the check must start immediately or not. The reference object (also a JSON object) contains the correlator for the notification, the endpoint address (a specific Bayeux channel name) to which the notifications must be sent and, optionally, the interface name (a string to identify the notification).

If the request for the Start Periodic Notification operation is successful, the endpoint address specified in the request body will receive a notification when:

- The terminal location has been successfully retrieved.
- The notification limit or the specified duration has been reached.
- An error has been encountered in obtaining the location of the terminal.

Authorization

Basic

HTTP Method

`PUT`

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Start Periodic Notification operation is a nested JSON object. It contains an array of addresses for the terminals in whose information the application is interested, and the parameters for the check provided by the following

- **addresses**: Array of string values. Required. Each element in the array is a SIP-formatted address for a terminal, as a URI.
- **frequency**: JSON object. Required. The following parameters define the frequency for checking the location of the terminal:
  - **metric**. String. Required. The unit of time for the check specified as **Millisecond**, **Second**, **Minute**, **Hour**, **Day**, **Week**, **Month**, or **Year**.
  - **units**: Integer. Required. The number of times the check must be made within the specified metric.

- **reference**: JSON object. Required. Use this object to provide the following information about the endpoint that is to receive the notification:
  - **correlator**. String. Required. The correlator used to identify the notification.
  - **endpoint**. String. Required. The URI which represents the endpoint address to which the notification should be delivered. This string should be a Bayeux protocol channel name that begins with **`/bayeux/`***`appInstanceID`* where *appInstanceID* is the client application's application instance account ID.

    For more information on application instances, see the discussion on creating and maintaining service provider and application accounts in *Oracle Communications Services Gatekeeper Accounts and SLAs Guide*.
  - **interfaceName**. String. Required. A descriptive string to identify the type of notification.
- **requestedAccuracy**: Number (floating point). Required. The acceptable error in the tracking, in meters.
- **duration**: JSON object. Required. This object specifies the total duration before for the count with the following:
  - **metric**. String. Required. The unit of time for the notifications specified as **Millisecond**, **Second**, **Minute**, **Hour**, **Day**, **Week**, **Month**, or **Year**.
  - **units**: Integer. Required. The duration in the specified metric. If this number is reached (and the application has not ended the notification operation), the **locationEnd** notification is delivered to the **endpoint** address.

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "addresses": ["URI"],
  "frequency": {
    "metric": "Millisecond|Second|Minute|Hour|Day|Week|Month|Year",
    "units": "Integer"
  },
  "reference": {
    "correlator": "String",
    "endpoint": "URI",
    "interfaceName": "String"
  },
  "requestedAccuracy": "Integer",
  "duration": {
    "metric": "Millisecond|Second|Minute|Hour|Day|Week|Month|Year",
    "units": "Integer"
  }
}
```

Response Header

The Location header field contains the URI of the publish/subscribe server:

`http://`*`host`*`:`*`port`*`/rest/terminal_location/notifications`

If the request fails, the Status-Line header field will contain the status code and the reason for the failure.

Response Body

There is no response body. The appropriate notifications (described below) are sent to the **endpoint** address provided by the application in the request body of this operation.

Examples

Example 11–13 Start Periodic Notification Request

```
PUT /rest/terminal_location/periodic-notification HTTP/1.1
X-Session-ID: app:4130997928482260925
Authorization: Basic ZG9tYWluX3VzZXI6ZG9tYWluX3VzZXI=
X-Param-Keys:
X-Param-Values:
User-Agent: Jakarta Commons-HttpClient/3.0
Host:10.182.100.218.8002
Content-Length: 366
Content-Type: application/json

{
   "addresses":["tel:123"],
   "frequency":
        {
          "metric": "Second",
          "units": "5"
        },
   "reference":
        {
          "correlator":"6789",
          "endpoint":"bayeux/app_instance_1/tl",
          "interfaceName":"interfaceName"
        },
   "requestedAccuracy":"10.0",
   "duration":
        {
          "metric":"Minutes",
          "units":"30"
        }
}
```

Example 11–14 Start Periodic Notification Response

```
HTTP/1.1 200 Created
Date: Fri, 05 Nov 2101 09:59:05 GMT
Location: http://10.182.100.218:8002/rest/terminal_location/notifications
Content-Length: 0
X-Plugin-Param-Keys:
X-Plugin-Param-Values:
X-Powered-By: Servlet/2.5 JSP/2.1
```

Notification When Terminal Location is Successfully Retrieved (locationNotification)

See "Notification When Terminal Location is Successfully Retrieved (locationNotification)".

Notification of Error in Retrieving Terminal Location (locationError)

See "Notification of Error in Retrieving Terminal Location (locationError)".

Notification Signalling End to Monitoring Terminal (locationEnd)

See "Notification Signalling End to Monitoring Terminal's Location (locationEnd)".

End Notification

The End Notification operation terminates an application's previously set up notification to get the geographical and periodic information for a specified terminal.

To stop a previously set up Start Geographical Notification or Start Periodic Notification, provide the correlator for the notification passed earlier in the appropriate start request.

There is no request or response body for the End Notification operation. If the request fails, the body of the error response will contain the identifier for the notification and the type of exception.

Authorization

Basic

HTTP Method

`PUT`

URI

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

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *correlator* is the correlator for the notification provided in the **reference** object of the initial request for the Start Geographical Notification or Start Periodic Notification operation.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure.

Response Body

There is no response body.

Examples

Example 11–15 End Notification Request

```
DELETE /rest/terminal_location/status-notification/6789 HTTP/1.1
X-Session-ID: app:4130997928482260925
Authorization: Basic ZG9tYWluX3VzZXI6ZG9tYWluX3VzZXI=
X-Param-Keys:
X-Param-Values:
User-Agent: Jakarta Commons-HttpClient/3.0
Host:10.182.100.218.8002
```

Example 11–16 End Notification Response

```
HTTP/1.1 204 No Content
Connection: close
Date: Thu, 04 Nov 2101 09:59:05 GMT
Content-Length: 0
X-Plugin-Param-Keys:
X-Plugin-Param-Values:
X-Powered-By: Servlet/2.5 JSP/2.1
```

12

Payment

This chapter describes the operations in the Payment interface of the RESTful Web Services provided in Oracle Communications Services Gatekeeper.

About the Payment Interface

Applications use the RESTful Payment interface 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. Applications can also reserve amounts, reserve additional amounts, charge against the reservation or release the reservation.

REST Service Descriptions Available at Run-time

When the Administration Server for your Services Gatekeeper domain is in the running state, the REST service descriptions of these operations can be found at

`http://host:port/rest/payment/index.html`

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Charge Amount

The Charge Amount operation charges an amount directly to an end-user's application using the Diameter protocol.

To charge an amount for a call, provide the SIP-formatted URI of the address of the end-user, a reference code in case there is any dispute regarding the charges, and the billing information to charge for the call.

There is no response body for the Charge Amount operation.

Authorization

Basic

HTTP Method

`POST`

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Charge Amount operation accepts the following parameters:

- **charging**. a JSON object. Optional. This object defines the cost-charging properties for the call. A call with no charging parameters can be entered as `"charging": null`. If a charge is to be applied, provide values for the following in the **charging** object:
  - **description**. String. Required if the **charging** object is present in the body of the request. The text to be used for information and billing.
  - **amount**. Number (integer, or decimal). Optional. The amount to be charged.
  - **code**. String. Optional. The charging code, from an existing contractual description.
  - **currency**. String. Optional. The currency identifier as defined in ISO 4217 [9].
- **endUserIdentifier**. String. Required. The address of the end-user's application that is to be charged.
- **referenceCode**. String. Required. A unique identifier in case of disputes with respect to the charges.

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "charge": {
    "description": "String",
```

```
    "amount": "BigDecimal",
    "code": "String",
    "currency": "String"
  },
  "endUserIdentifier": "URI",
  "referenceCode": "String"
}
```

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body.

Refund Amount

The Refund Amount operation refunds an amount directly to an end-user's application using Diameter.

To refund an amount for a call, provide the SIP-formatted URI of the address of the end-use, a reference code in case there is any dispute regarding the charges, and the billing information to charge for the call. receive the message in the request body.

There is no response body for the Refund Amount operation.

Authorization

Basic

HTTP Method

`POST`

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Refund Amount operation accepts the following parameters:

- **charging**. a JSON object. Optional. This object defines the cost-charging properties for the call. A call with no charging parameters can be entered as `"charging": null`. If a charge is to be applied, provide values for the following in the **charging** object:
  - **description**. String. Required if the `charging` object is present in the body of the request. The text to be used for information and billing.
  - **amount**. Number (integer, or decimal). Optional. The amount to be refunded.
  - **code**. String. Optional. The charging code, from an existing contractual description.
  - **currency**. String. Optional. The currency identifier as defined in ISO 4217 [9].
- **endUserIdentifier**. String. Required. The address of the end-user's application that is to receive the refund.
- **referenceCode**. String. Required. A unique identifier in case of disputes with respect to the refund.

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "charge": {
    "description": "String",
```

```
    "amount": "BigDecimal",
    "code": "String",
    "currency": "String"
  },
  "endUserIdentifier": "URI",
  "referenceCode": "String"
}
```

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body.

Charge Split Amount

The Charge Split Amount operation charges an amount directly to multiple end-users concurrently (for example, for charging multiple participants in a conference.

To split the charge an amount for a call, provide the billing information to charge for the call, a reference code in case there is any dispute regarding the charges, the address of the end-user, and the percentage of the charges for which the end-user is liable.

There is no response body for the Charge Split Amount operation.

Authorization

Basic

HTTP Method

`POST`

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Charge Split Amount operation accepts the following parameters:

- **charge**. a JSON object. Required. Use this object to define the cost-charging properties for the call. A call with no charging parameters can be entered as `"charging": null`. If a charge is to be applied, provide values for the following in the **charging** object:
  - **description**. String. Required if the **charging** object is present in the body of the request. The text to be used for information and billing.
  - **amount**. Number (integer, or decimal). Optional. The amount to be charged.
  - **code**. String. Optional. The charging code, from an existing contractual description.
  - **currency**. String. Optional. The currency identifier as defined in ISO 4217 [9].
- **referenceCode**. String. Required. A unique identifier in case of disputes with respect to the charges.
- **splitInfo**. An array of JSON objects. Required. For each entry, the end-user identifier and the method by which the charges must be split.
  - **endUserIdentifier**. String. Required. The address of the end-user.
  - **percent**. Integer. Required. The percentage this end-user should be charged.

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "charge": {
    "description": "String",
    "amount": "BigDecimal",
    "code": "String",
    "currency": "String"
  },
  "referenceCode": "String",
  "splitInfo": [{
    "endUserIdentifier": "URI",
    "percent": "Integer"
  }]
}
```

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body.

Reserve Amount

The Reserve Amount operation reserves an amount for an account specified by the end-user identifier.

To reserve an amount for a call, provide the address of the end-user and the billing information for the call.

If the Reserve Amount operation successful, the response body will contain the string identifier for the reservation.

Authorization

Basic

HTTP Method

`POST`

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Reserve Amount operation accepts the following parameters:

- **charge**. a JSON object. Optional. Use this object to define the cost-charging properties for the call. A call with no charging parameters can be entered as `"charging": null`. If a charge is to be applied, provide values for the following in the **charging** object:
  - **description**. String. Required if the **charging** object is present in the body of the request. The text to be used for information and billing.
  - **amount**. Number (integer, or decimal). Optional. The amount to be charged.
  - **code**. String. Optional. The charging code, from an existing contractual description.
  - **currency**. String. Optional. The currency identifier as defined in ISO 4217 [9].
- **endUserIdentifier**. String. Required. The address of the end-user against whose account the reservation is made.

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "charge": {
    "description": "String",
    "amount": "BigDecimal",
    "code": "String",
    "currency": "String"
```

```
  },
  "endUserIdentifier": "URI"
}
```

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The response body contains the **result** attribute whose value is a String-formatted identifier for the reservation (used as **reservationIdentifier** in subsequent related operations).

The response body for this operation is represented by the following JSON data structure, where the value part of the name/value pair indicates its data type:

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

Reserve Additional Amount

The Reserve Additional Amount operation reserves an additional amount for an account specified by the end-user identifier.

To reserve an additional amount for a call, provide the reservation identifier obtained from the initial request to reserve an amount for the end-user and the billing information for the call.

If the Reserve Additional Amount operation is successful, the response body will contain the string identifier for the reservation.

Authorization

Basic

HTTP Method

`POST`

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Reserve Additional Amount operation accepts the following parameters:

- **charge**. a JSON object. Optional. Use this object to define the cost-charging properties for the call. A call with no charging parameters can be entered as `"charging":null`. If a charge is to be applied, provide values for the following in the **charging** object:
  - **description**. String. Required if the **charging** object is present in the body of the request. The text to be used for information and billing.
  - **amount**. Number (integer, or decimal). Optional. The amount to be charged.
  - **code**. String. Optional. The charging code, from an existing contractual description.
  - **currency**. String. Optional. The currency identifier as defined in ISO 4217 [9].
- **reservationIdentifier**. String. Required. The string identifier **result** obtained from the initial "Reserve Amount" operation for this account.

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "charge": {
    "description": "String",
```

```
    "amount": "BigDecimal",
    "code": "String",
    "currency": "String"
  },
  "reservationIdentifier": "String"
}
```

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body.

Charge Volume

The Charge Volume operation charges the specified volume to the account specified by the end-user identifier.

Authorization

Basic

HTTP Method

`POST`

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Charge Volume operation accepts the following parameters:

- **endUserIdentifier**. String. Required. Identifies the end-user account to be charged.
- **volume**. Long. Required. Identifies the volume amount to be charged. (This is not a currency amount.)
- **billingText**. String. Required. Textual information to appear on the bill.
- **referenceCode**. String. Required. Code to uniquely identify the request.
- **parameters[0...unbounded]**. String. Optional. Additional name/value pairs to use to perform rating.

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "endUserIdentifier": "URI"
  "volume": "BigDecimal"
  "billingText": "String"
  "referenceCode": "String"
  "parameters": {
    "name": "String",
    "value": "String",
  },
}
```

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body.

Refund Volume

The Refund Volume operation directly applies a refund volume to the account specified by the end-user identifier.

Authorization

Basic

HTTP Method

`POST`

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Refund Volume operation accepts the following parameters:

- **endUserIdentifier**. String. Required. Identifies the end-user account to be refunded.
- **volume**. Long. Required. Identifies the volume amount to be refunded.
- **billingText**. String. Required. Textual information to appear on the bill.
- **referenceCode**. String. Required. Code to uniquely identify the request.
- **parameters[0...unbounded]**. String. Optional. Additional name/value pairs to use to perform rating.

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "endUserIdentifier": "URI"
  "volume": "BigDecimal"
  "billingText": "String"
  "referenceCode": "String"
  "parameters": {
    "name": "String",
    "value": "String",
  },
}
```

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body.

Charge Split Volume

The Charge Split Volume operation applies a split volume charge to multiple end-user accounts.

The portion of the volume charge applied to each account is expressed as a percentage.

Authorization

Basic

HTTP Method

`POST`

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Charge Split Volume operation accepts the following parameters:

- **splitInfo[1...unbounded]**. a JSON object. Required. Use this object to define the end-user accounts and the percentage of the volume for the request to be billed to each account. Percentages must total 100:
  - **endUserIdentifier**. String. Required. Identifies an end-user account to be charged.
  - **percent**. Number (integer, or decimal). Required. The percentage of the transaction to be charged to the end-user account.
- **volume**. Long. Required. Identifies the volume amount to be refunded. (This is not a currency amount.)
- **billingText**. String. Required. Textual information to appear on the bill.
- **referenceCode**. String. Required. Code to uniquely identify the request.
- **parameters[0...unbounded]**. String. Optional. Additional name/value pairs to use to perform rating.

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "splitInfo": {
    "endUserIdentifier": "URI"
    "percent": "BigDecimal",
  },
  "volume": "BigDecimal"
  "billingText": "String"
```

```
  "referenceCode": "String"
  "parameters": {
    "name": "String",
    "value": "String",
  },
}
```

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body.

Get Amount

The Get Amount operation converts a volume to a currency amount for the end-user account.

Authorization

Basic

HTTP Method

`POST`

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Get Amount operation accepts the following parameters:

- **endUserIdentifier**. String. Required. Identifies the end-user account for the currency calculation.
- **volume**. Long. Required. Identifies the volume to be converted to a currency amount.
- **parameters[0...unbounded]**. String. Optional. Additional name/value pairs to use to perform rating.

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "endUserIdentifier": "URI"
  "volume": "BigDecimal"
  "parameters": {
    "name": "String",
    "value": "String",
  },
}
```

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

A **charging** object is returned, with the **currency** element containing the converted currency amount.

Charge Reservation

The Charge Reservation operation charges a previously reserved amount against an end-user account.

To charge a previously reserved amount to an end-user account, provide the information for billing, the reservation identifier obtained from the initial request to reserve an amount for the end-user, and the reference code for any possible disputes.

Authorization

Basic

HTTP Method

POST

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Charge Reservation operation accepts the following parameters:

- **charge**. a JSON object. Required. Use this object to define the cost-charging properties for the call. A call with no charging parameters can be entered as `"charging":null`. If a charge is to be applied, provide values for the following in the **charge** object:
  - **description**. String. Required if the **charging** object is present in the body of the request. The text to be used for information and billing.
  - **amount**. Number (integer, or decimal). Optional. The amount to be charged.
  - **code**. String. Optional. The charging code, from an existing contractual description.
  - **currency**. String. Optional. The currency identifier as defined in ISO 4217 [9].
- **referenceCode**. String. Required. A unique identifier in case of disputes with respect to the charges.
- **reservationIdentifier**. String. Required. The string identifier **result** obtained from the initial Reserve Amount operation for this account.

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "charge": {
    "description": "String",
```

```
    "amount": "BigDecimal",
    "code": "String",
    "currency": "String"
  },
  "referenceCode": "String",
  "reservationIdentifier": "String"
}
```

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body.

Release Reservation

The Release Reservation operation returns funds left in a reservation to the account against which this reservation was made.

To returns funds left in a reservation to an account, provide the reservation identifier obtained from the initial request to reserve an amount for the end-user.

Authorization

Basic

HTTP Method

`POST`

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Release Reservation operation accepts the following parameter:

- **reservationIdentifier**. String. Required. The string identifier `result` obtained from the initial Reserve Amount operation for this account.

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

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

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body.

Reserve Volume

The Reserve Volume operation reserves a volume for the account specified by the end-user identifier.

If the Reserve Volume operation is successful, the response body contains the string identifier for the reservation.

Authorization

Basic

HTTP Method

`POST`

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Reserve Volume operation accepts the following parameters:

- **endUserIdentifier**. String. Required. Identifies the end-user account for which the reservation should be placed.
- **volume**. Long. Required. Identifies the volume amount to be reserved.
- **billingText**. String. Required. Textual information to appear on the bill.
- **parameters[0...unbounded]**. String. Optional. Additional name/value pairs to use to perform rating.

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "endUserIdentifier": "URI"
  "volume": "BigDecimal"
  "billingText": "String"
  "parameters": {
    "name": "String",
    "value": "String",
  },
}
```

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The response body consists of a String containing an identifier for the newly created reservation.

Reserve Additional Volume

The Reserve Additional Volume operation adds or subtracts a volume to or from an existing volume reservation for the account specified by the end-user identifier.

To reserve an additional volume for a call, provide the reservation identifier obtained from the initial request to reserve an amount for the end-user and the billing information for the call.

If the Reserve Additional Volume operation is successful, the response body will contain the string identifier for the reservation.

Authorization

Basic

HTTP Method

`POST`

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Reserve Additional Volume operation accepts the following parameters:

- **reservationIdentifier**. String. Required. Identifies the reservation to be amended.
- **volume**. Long. Required. Identifies the volume amount to be added to or subtracted from to the existing reservation.
- **billingText**. String. Required. Textual information to appear on the bill.

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "reservationIdentifier": "String"
  "volume": "BigDecimal"
  "billingText": "String"
  },
}
```

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body.

Get Amount Reserve Charging

The Get Amount Reserve Charging operation converts a reserved volume to a currency amount for the end-user account.

Authorization

Basic

HTTP Method

`POST`

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Get Amount Reserve Charging operation accepts the following parameters:

- **endUserIdentifier**. String. Required. Identifies the end-user account for the currency calculation.
- **volume**. Long. Required. Identifies the volume amount to be converted to a currency amount.
- **parameters[0...unbounded]**. String. Optional. Additional name/value pairs to use to perform rating.

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "endUserIdentifier": "URI"
  "volume": "BigDecimal"
  "parameters": {
    "name": "String",
    "value": "String",
  },
}
```

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

A **charging** object is returned, with the **currency** element containing the converted reserved currency amount.

13

Audio Call

This chapter describes the operations in the Audio Call interface of the RESTful Web Services provided in Oracle Communications Services Gatekeeper.

About the Audio Call Interface

Applications use the RESTful Audio Call interface to get an audio file from Text to Speech file, Voice XML file from an independent location and send it to a terminal.

REST Service Descriptions Available at Run-time

When the Administration Server for your Services Gatekeeper domain is in the running state, the REST service descriptions of these operations can be found at

`http://host:port/rest/audio_call/index.html`

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Play Audio Message

The Play Audio Message operation sends an audio file (such as .WAV) to a single terminal. If the request contains more than one URI, the operation will fail.

To play the audio message at a specific terminal, provide the address of the terminal that is to receive the message, the URL of the audio (.wav) file, and, optionally, the information for billing the call. In effect, this is a request to set up a call to the user (the terminal) and play the audio file.

The call to play the audio file at the terminal is done asynchronously and can be monitored by retrieving its status. If the Play Audio Message operation is successful, the response body will contain the call identifier for the actual call when the audio file is played at the terminal. Use this call identifier to check on the request status.

Authorization

Basic

HTTP Method

`POST`

URI

`http://`*`host`*`:`*`port`*`/rest/audio_call/audiocalls`

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Play Audio Message operation accepts the following parameters:

- **address**. String. Required. The SIP-formatted URI of the party who must receive the audio message.
- **audioUrl**. String. Required. The URI of the audio file that is to be played.
- **charging**: a JSON object. Optional. This object defines the cost charging properties for the call. A call with no charging parameters can be entered as `"charging": null`. If a charge is to be applied, provide values for the following in the **charging** object:
  - **description**. String. Required if the `charging` object is present in the body of the request. The text to be used for information and billing.
  - **amount**: Number (integer, or decimal). Optional. The amount to be charged.
  - **code**. String. Optional. The charging code, from an existing contractual description.
  - **currency**. String. Optional. The currency identifier as defined in ISO 4217 [9].

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "address": "URI",
  "audioUrl": "URI",
  "charging": {
    "description": "String",
    "amount": "BigDecimal",
    "code": "String",
    "currency": "String"
  }
}
```

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The call identifier for the actual call when the audio file is played at the terminal is returned in the body of the response as the value of the **result** attribute.

The response body for this operation is represented by the following JSON data structure, where the value part of the name/value pair indicates its data type:

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

Examples

Example 13–1 Play Audio Message Request

```
GET /rest/audio_call/audiocalls HTTP/1.1
X-Session-ID: app:65671150306510708
Authorization: Basic YxBwXzFfdToxMjM0NTY3oA=
X-Param-Keys:
X-Param-Values:
User-Agent: Jakarta Commons-HttpClient/3.0
Host:10.182.100.142:8001
Content-Length: 125
Content-Type: application/json

{
  "audioUrl":"file://opt/OC/share/jsr309-TCK-media/dtmfs-1-9.wav",
  "address":"sip:alice@10.182.100.142.10072",
  "charging":
       {
        "description":"Message for Alice",
        "amount":"11",
        "code":"1111",
        "currency":"rmb"
        }
}
```

Example 13–2 Play Audio Message Response

```
HTTP/1.1 201 Created
Date: Fri, 05 Nov 2010 02:03:09 GMT
Content-Length: 95
Content-Type: application/json
X-Plugin-Param-Keys:
X-Plugin-Param-Values:
X-Powered-By: Servlet/2.5 JSP/2.1

{"result":"app-6nbs6twem41p|5624bd08b079138534263491b78d49e9@10.182.100.142|-40d94
a935cd12af0"}
```

Play Text Message

The Play Text Message operation sends a Text-to-Speech to a single terminal. The text is read through a text to speech engine and based on a specified format. If the request contains more than one URI, the operation will fail.

To play the text message at a specific terminal, provide the address of the terminal that is to receive the message, the text message and its language format, and, optionally, the information for billing the call. In effect, this is a request to set up a call to the user (the terminal) and play the text file.

The call to play the text message at the terminal is done asynchronously and can be monitored by retrieving its status. If the Play Text Message operation is successful, the response body will contain the call identifier for the actual call that is to play the text message at the terminal. Use this call identifier to check on the request status.

Authorization

Basic

HTTP Method

`POST`

URI

`http://`*`host`*`:`*`port`*`/rest/audio_call/textcalls`

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Play Audio Message operation accepts the following parameters:

- **address**. String. Required. The SIP-formatted URI of the party who must receive the text message.
- **text**. String. Required. The text to be read as text-to-speech (the "voice" structure in Speech Synthesis Markup Language (SSML) format.
- **language**. String. Required. The language in which the message is to be played, using ISO 639 format.
- **charging**. a JSON object. Optional. This object defines the cost charging properties for the call. A call with no charging parameters can be entered as `"charging": null`. If a charge is to be applied, provide values for the following in the **charging** object:
  - **description**. String. Required if the **charging** object is present in the body of the request. The text to be used for information and billing.
  - **amount**. Number (integer, or decimal). Optional. The amount to be charged.

- **code**. String. Optional. The charging code, from an existing contractual description.
- **currency**. String. Optional. The currency identifier as defined in ISO 4217 [9].

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "address": "URI",
  "language": "String",
  "text": "String",
  "charging": {
    "description": "String",
    "amount": "BigDecimal",
    "code": "String",
    "currency": "String"
  }
}
```

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The call identifier for the actual call when the text-to-speech file is played at the terminal is returned in the body of the response as the value of the **result** attribute.

The response body for this operation is represented by the following JSON data structure, where the value part of the name/value pair indicates its data type:

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

Examples

Example 13–3 Play Text Message Request

```
GET /rest/audio_call/textcalls HTTP/1.1
X-Session-ID: app:65671150306510708
Authorization: Basic YxBwXzFfdToxMjM0NTY3oA=
X-Param-Keys:
X-Param-Values:
User-Agent: Jakarta Commons-HttpClient/3.0
Host:10.182.100.142:8001
Content-Length: 125
Content-Type: application/json

{
  "address":"sip:alice@10.182.100.142.10072",
  "text":"WHATSTHIS",
  "language":"WHATSTHIS",
  "charging":
       {
        "description":"Message for Alice",
        "amount":"11",
        "code":"1111",
        "currency":"rmb"
```

```
        }
}
```

Example 13–4 Play Text Message Response

```
HTTP/1.1 201 Created
Date: Fri, 05 Nov 2010 02:03:09 GMT
Content-Length: 95
Content-Type: application/json
X-Plugin-Param-Keys:
X-Plugin-Param-Values:
X-Powered-By: Servlet/2.5 JSP/2.1

{"result":"app-6nbs6twem41p|5624bd08b079138534263491b78d49e9@10.182.100.142|-40d94
a935cd12af0"}
```

Play VoiceXML Message

The Play Text Message operation sets up a request to send a specified VoiceXML content located at a specific voice URL to a single terminal. If the request contains more than one URI, the operation will fail.

To play the voice xml message at a specific terminal, provide the address of the terminal that is to receive the message, the location of the VoiceXML file, and, optionally, the information for billing the call. In effect, this is a request to set up a call to the user (the terminal) and play the text file.

The call to play the text message at the terminal is done asynchronously and can be monitored by retrieving its status. If the Play Text Message operation is successful, the response body will contain the call identifier for the actual call that is to play the text message at the terminal. Use this call identifier to check on the request status.

Authorization

Basic

HTTP Method

`POST`

URI

`http://`*`host`*`:`*`port`*`/rest/audio_call/voicexmlcalls`

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Play Audio Message operation accepts the following parameters:

- **address**. String. Required. The SIP-formatted URI of the party who must receive the text message.
- **voiceXmlUrl**. String. Required. The location of the VoiceXML file.
- **charging**. a JSON object. Optional. This object defines the cost charging properties for the call. A call with no charging parameters can be entered as `"charging": null`. If a charge is to be applied, provide values for the following in the `charging` object:
  - **description**. String. Required if the **charging** object is present in the body of the request. The text to be used for information and billing.
  - **amount**: Number (integer, or decimal). Optional. The amount to be charged.
  - **code**. String. Optional. The charging code, from an existing contractual description.
  - **currency**. String. Optional. The currency identifier as defined in ISO 4217 [9].

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "address": "URI",
  "voiceXmlUrl": "URI",
  "charging": {
    "description": "String",
    "amount": "BigDecimal",
    "code": "String",
    "currency": "String"
  }
}
```

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The call identifier for the actual call when the VoiceXML is played at the terminal is returned in the body of the response as the value of the **result** attribute.

The response body for this operation is represented by the following JSON data structure, where the value part of the name/value pair indicates its data type:

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

Examples

Example 13–5 Play VoiceXML Message Request

```
GET /rest/audio_call/voicexmlcalls HTTP/1.1
X-Session-ID: app:65671150306510708
Authorization: Basic YxBwXzFfdToxMjM0NTY3oA=
X-Param-Keys:
X-Param-Values:
User-Agent: Jakarta Commons-HttpClient/3.0
Host:10.182.100.142:8001
Content-Length: 125
Content-Type: application/json

{
  "address":"sip:alice@10.182.100.142.10072",
  "voiceXmlUrl":"http://some_url/mesg_recorded.vxml",
  "charging":
       {
        "description":"Message for Alice",
        "amount":"11",
        "code":"1111",
        "currency":"rmb"
        }
}
```

Example 13–6 Play VoiceXML Message Response

```
HTTP/1.1 201 Created
Date: Fri, 05 Nov 2010 02:03:09 GMT
Content-Length: 95
Content-Type: application/json
X-Plugin-Param-Keys:
X-Plugin-Param-Values:
X-Powered-By: Servlet/2.5 JSP/2.1

{"result":"app-6nbs6twem41p|5624bd08b079138534263491b78d49e9@10.182.100.142|-40d94
a935cd12af0"}
```

Get Message Status

The Get Message Status operation retrieves the status of a previously set up request, which in this case is the status of an audio file bound for a single terminal.

To retrieve the status of the audio file, provide the appropriate correlator. This correlator is the string value of the `result` attribute from the response body for the appropriate Play (Audio/Text/VoiceXML) Message operation.

If the Get Message Status operation is successful, the response body contains a JSON data object indicating whether the audio file is pending, playing, has played or an error has occurred.

Authorization

Basic

HTTP Method

`GET`

URI

`http://`*`host`*`:`*`port`*`/rest/audio_call/call?getMessageStatus=`*`${getMessageStatus}`*

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *${getMessageStatus}* contains the call identifier from the response body for the appropriate Play (Audio/Text/VoiceXML) Message operation as the string value for the `correlator` attribute.

The *${getMessageStatus}* object in the URI is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

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

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The status of the specific audio file is returned in the body of the response as the value of the **result** attribute:

- **Played**
- **Playing**

- **Pending**
- **Error**

The response body for this operation is represented by the following JSON data structure, where the value part of the name/value pair indicates its data type:

```
{"result": "Played|Playing|Pending|Error"}
```

Examples

Example 13–7 Get Message Status Request

```
GET /rest/audio_
call/call?getMessageStatus=%7B%22correlator%22%3A%22app-6nbs6twem41p%7c5624bd08b07
9138534263491b78d49e9%4010.182.100.142%7c-40d94a935cd12af0 HTTP/1.1
X-Session-ID: app:6567114150306510708
Authorization: Basic YXBwXzFfdToxMjM0NTY30A=
X-Param-Keys:
X-Param-Values:
User-Agent: Jakarta Commons-HttpClient/3.0
Host:10.182.100.142.8001
```

Example 13–8 Get Message Status Response

```
HTTP/1.1 200 OK
Date: Thu, 05 Nov 2101 02:18:18 GMT
Content-Length: 19
Content-Type: application/json
X-Plugin-Param-Keys:
X-Plugin-Param-Values:
X-Powered-By: Servlet/2.5 JSP/2.1

{"result":"Played"}
```

End Message

The End Message operation terminates an application's previously set up request to play the specified audio file.

To stop a previously set up request to play an audio file, provide the appropriate correlator. This correlator is the string value of the `result` attribute from the response body for the appropriate Play (Audio/Text/VoiceXML) Message operation.

Authorization

Basic

HTTP Method

`DELETE`

URI

`http://host:port/rest/audio_call/call?endMessage=${endMessage}`

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *endMessage* contains the call identifier from the response body for the appropriate Play (Audio/Text/VoiceXML) Message operation as the string value for the `correlator` attribute.

  The *${endMessage}* object in the URI is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

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

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The response body contains a data object with the current status of the audio file as the value of the **result** attribute. The value can be one of the following:

- **Played**
- **Playing**
- **Pending**
- **Error**

The response body for this operation is represented by the following JSON data structure, where the value part of the name/value pair indicates its data type:

```
{"result": "Played|Playing|Pending|Error"}
```

Examples

Example 13–9 End Message Request

```
DELETE /rest/audio_
call/end-call?endMessage=%7B%22correlator%22%3A%22app-gfim8nqt869v%7cb4142a2f38d1d
1e6341c897734c93a91%4010.182.100.142%7c-40d94a935cd12af0%22%7D HTTP/1.1
X-Session-ID: app:6567114150306510708
Authorization: Basic YXBwXzFfdToxMjM0NTY30A=
X-Param-Keys:
X-Param-Values:
User-Agent: Jakarta Commons-HttpClient/3.0
Host:10.182.100.132.8001
```

Example 13–10 End Message Response

```
HTTP/1.1 200 OK
Connection: close
DATE: Fri, 05 Nov 2010 02:31:31 GMT
Content-Length: 20
Content-Type: application/json
X-Plugin-Param-Keys:
X-Plugin-Param-Values:
X-Powered-By: Servlet/2.5 JSP/2.1

{"result":"Playing"}
```

14

Quality of Service

This chapter describes the RESTful interface for the Services Gatekeeper QoS communication service.

About the QoS Interface

An application can use the QoS RESTful interface to apply a QoS policy, query, modify and remove that policy and register as well as unregister for QoS-related notifications. A Policy and Charging Rule Function provider (PCRF) can also return QoS events to registered applications.

REST Service Descriptions Available at Run-time

When the Administration Server for your Services Gatekeeper domain is in the running state, the REST service descriptions of QoS operations can be found at

`http://host:port/application.wadl`

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

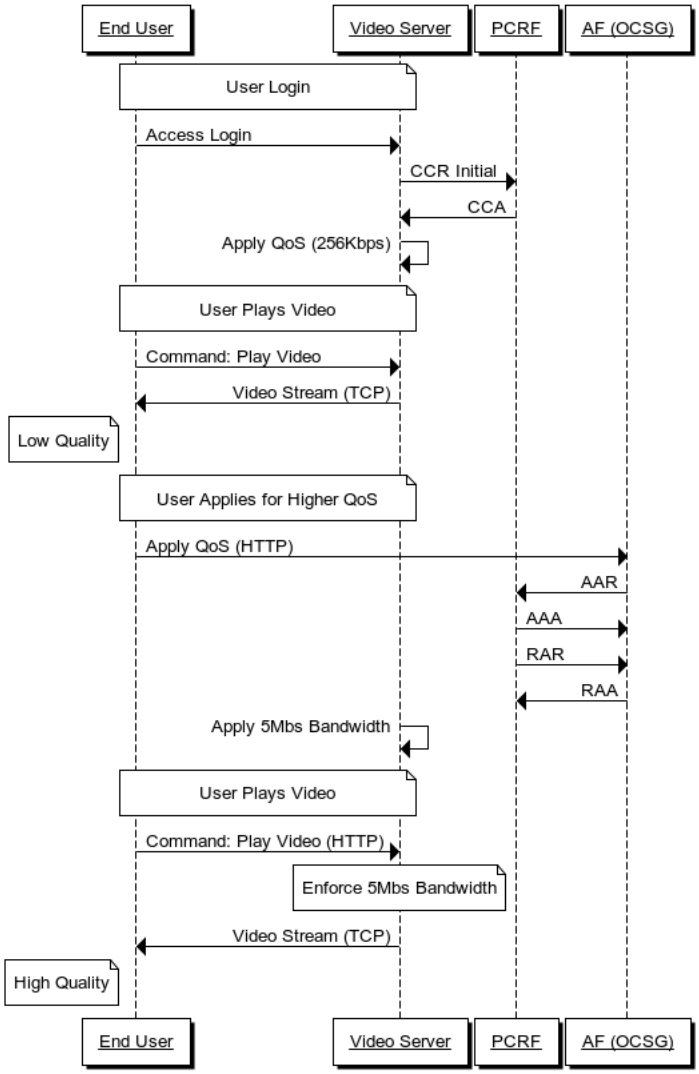
Example QoS Scenario

A typical QoS scenario involves a subscriber using a handset to access a video feed using a video application installed on the handset. Initially, because the default QoS is set to a low bandwidth, the video stops and stutters frequently as it is buffered repeatedly over the low speed connection. The subscriber requests a faster QoS via the application, presumably with a corresponding billing charge. Services Gatekeeper forwards that request to a PCRF which then applies the upgraded QoS. The subscriber's video now streams at the upgraded speed, without stuttering.

> **Note:** While QoS frequently refers to raw bandwidth speed, it can apply to any factors that affect network performance, for example, connection latency and time-out.

Figure 14–1 shows a detailed QoS call flow sequence.

Figure 14–1 Example QoS Call Sequence



In Figure 14–1:

1. A user logs into the video server.
2. The video server initiates an initial Credit Check Request (CCR) to the PCRF.
3. The PCRF returns a Credit Check Authorization (CCA) to the video server and the low bandwidth, 256Kbps QoS plan is applied.
4. The user plays the video using the low bandwidth QoS plan; video playback is low quality with stuttering and continual buffering requests.
5. The user requests a better QoS plan using the applyQos RESTful request from the handset's host application.
6. Upon receiving the applyQoS request, Services Gatekeeper issues an Authorization and Authentication Request (AAR) to the PCRF which then returns an Authorization and Authentication Answer (AAA).

7. The PCRF issues a Re-Authorization Request (RAR) to Services Gatekeeper, which then returns a Re-Authorization Answer, and the high bandwidth, 5Mbps QoS plan is applied.
8. The user plays the video, and the new 5Mbps QoS plan is enforced. The video plays smoothly, without stuttering or continued buffering.

Configuring QoS for Services Gatekeeper

Before you can implement QoS functionality, a QoS plug-in must be deployed and configured in Services Gatekeeper. For information on deploying and configuring QoS plug-ins, see "Managing the QoS Communication Service" in the *Oracle Communications Services Gatekeeper Communication Service Guide*, another document in this document set.

Using OAuth with QoS

The Services Gatekeeper QoS communication service fully supports OAuth 2.0 authentication between the QoS communication service itself and an AT application.

To establish OAuth authentication between the QoS communication service and your application, do the following:

1. From your application, contact the QoS communication service and request an OAuth token.
2. The QoS communication service will return an OAuth token to your application.
3. Add the access_token (UUID) to your application's request header:

   ```
   access_token: 3ddc24b2-5b17-4d46-8818-6e14726b217c
   ```

4. In addition, add the Authorization parameter to your application's http header:

   ```
   Authorization: Bearer 3ddc24b2-5b17-4d46-8818-6e14726b217c
   ```

For more information on using OAuth authentication, see the *Oracle Communications Services Gatekeeper OAuth Guide*.

Apply QoS

The Apply QoS operation requests that a QoS plan be applied to end user IDs as specified in a Services Gatekeeper QoS plug-in regular expression matching rule.

Authorization

Basic or OAuth 2.0

HTTP Method

`POST`

URI

`http://host:port/ApplicationQoSService/${endUserId}/qos`

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *endUserId* is a valid end user identifier.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Apply QoS operation accepts the following parameters:

- **duration**. Unsigned Long. Optional. The duration of the applied QoS in seconds. If no duration is specified, the QoS session will not time out until it is explicitly removed.
- **applicationIdentifier**. String. Optional. Identifies the service to which the application-facing service session belongs.
- **mediaComponentDescription**. Complex. Optional. Service information for a single media component within an application-facing session.

  See "mediaComponentDescription" for details on each of the mediaComponentDescription parameters.
- **serviceInfoStatus**. Complex. Optional. Indicates the status of the service information that the application facing interface is providing to the PCRF. If this parameter is not provided, the value, **FINAL_SERVICE_INFORMATION**, is assumed. The service info status is specified as one of the following:
  - **FINAL_SERVICE_INFORMATION**
  - **PRELIMINARY_SERVICE_INFORMATION**
- **chargingIdentifier**. String. Optional. Application facing charging identifier.
- **sIPForkingIndication**. Enumerated. Optional. Indicates if multiple SIP dialogs are related to a single Diameter session. If this parameter is not provided, the value, **SINGLE_DIALOGUE**, is assumed. The SIP forking indication is specified as one of the following:

  - **SINGLE_DIALOGUE**
  - **SEVERAL_DIALOGUES**

- **subscriptionId**. Complex. Optional. An end user's subscription ID.

  See "subscriptionID" for details on each of the subscriptionId parameters.

- **supportedFeatures**. Complex. Optional. If present, informs the destination host about the features that the origin host requires to successfully complete the command exchange.

  See "supportedFeatures" for details on each of the supportedFeatures parameters.

- **reservationPriority**. Enumerated. Optional. Applies to all those IP flows within the media component and describes the relative importance of the IP flow as compared to other IP flows. If this parameter is not specified, the default value is **0**. The reservation priority is specified as one of the following:
  - **0**
  - **7**
- **framedIPAddress**. String. Optional. The valid routable IPv4 address that is applicable for the IP Flows towards the user equipment at the PCEF.
- **framedIPv6Prefix**. String. Optional. A valid full IPv6 address that is applicable to an IP flow or IP flows towards the user equipment at the PCEF.
- **calledStationId**. String. Optional. If a private IP address is being used, the ID of the packet data network.
- **serviceURN**. Enumerated. Optional. Indicates that the AF session is used for emergency traffic. The service URN is specified as one of the following:
  - **counseling**
  - **counseling.children**
  - **counseling.mental-health**
  - **counseling.suicide**
  - **sos**
  - **sos.ambulance**
  - **sos.animal-control**
  - **sos.fire**
  - **sos.gas**
  - **sos.marine**
  - **sos.mountain**
  - **sos.physician**
  - **sos.poison**
  - **sos.police**
- **sponsoredConnectivityData**. Complex. Optional. Indicates the data associated with the sponsored data connectivity that the AF is providing to the PCRF.

  See "sponsoredConnectivityData" for details on each of the sponsoredConnectivityData parameters.

- **mPSIdentifier**. String. Optional. Indicates that an AF session relates to an MPS session. It contains the national variant for the MPS service name, for example, NGN GETS.
- **applicationIdentifier**. String. Optional. Identifies the particular service to which the media component belongs. If the parameter is not present, the applicationIdentifier from the main body of the request is used.

mediaComponentDescription

These are the parameters for the optional MediaComponentDescription.

- **mediaComponentNumber**. Unsigned Integer. Required. Ordinal number of the media component.
- **mediaSubComponent**. Complex. Optional. The requested bitrate and filters for the set of IP flows identified by their common flow identifier.

  See "mediaSubComponent" for details on each of the mediaSubComponent parameters.
- **applicationIdentifier**. String. Optional. Identifies the particular service to which the media component belongs. If the parameter is not present, the applicationIdentifier from the main body of the request is used.
- **mediaType**. Enumerated. Optional. Determines the media type of the session component. The media type is specified as one of the following:
  - **AUDIO**
  - **VIDEO**
  - **DATA**
  - **APPLICATION**
  - **CONTROL**
  - **TEXT**
  - **MESSAGE**
  - **OTHER**
- **maxRequestedBandwidthUL**. Unsigned Integer. Optional. The maximum requested bandwidth in bits per second for an uplink IP flow.
- **maxRequestedBandwidthDL**. Unsigned Integer. Optional. The maximum requested bandwidth in bits per second for a downlink IP flow.
- **minRequestedBandwidthUL**. Unsigned Integer. Optional. The minimum requested bandwidth in bits per second for an uplink IP flow.
- **minRequestedBandwidthDL**. Unsigned Integer. Optional. The minimum requested bandwidth in bits per second for a downlink IP flow.
- **flowStatus**. Enumerated. Optional. Describes whether the IP flow(s) are enabled or disabled. The flow status is specified as one of the following:
  - **ENABLED-UPLINK**
  - **ENABLED-DOWNLINK**
  - **ENABLED**
  - **DISABLED**
  - **REMOVED**

- **reservationPriority**. Enumerated. Optional. Applies to all those IP flows within the media component and describes the relative importance of the IP flow as compared to other IP flows. If this parameter is not specified, the value is **0**. The reservation priority is specified as one of the following:
  - **0**
  - **7**
- **rSBandwidth**. Unsigned Integer. Optional. Indicates the maximum required bandwidth in bits per second for RTCP sender reports within the session component.
- **rBBandwidth**. Unsigned Integer. Optional. Indicates the maximum required bandwidth in bits per second for RTCP receiver reports within the session component.
- **codecData**. String. Optional. Codec-related information known at the AF. The encoding rule should comply with 3gpp TS 29.214 [5.3.7].

mediaSubComponent

These are the parameters for the optional mediaSubComponent parameter.

- **flowNumber**. Unsigned Integer. Required. Ordinal number of the IP flow.
- **flowDescription**. String. Optional. Filters for an IP flow. The format must comply with RFC3588 [4.3] IPFilterRule and 3gpp TS 29.214 [5.3.8].
- **flowStatus**. Enumerated. Optional. Describes whether the IP flow(s) are enabled or disabled. The flow status is specified as one of the following:
  - **ENABLED-UPLINK**
  - **ENABLED-DOWNLINK**
  - **ENABLED**
  - **DISABLED**
  - **REMOVED**
- **flowUsage**. Enumerated Optional. Provides information about the usage of IP flows. The flow usage is specified as one of the following:
  - **NO_INFORMATION**
  - **RTCP**
  - **AF_SIGNALLING**
- **maxRequestedBandwidthUL**. Unsigned Integer. Optional. The maximum requested bandwidth in bits per second for an uplink IP flow.
- **maxRequestedBandwidthDL**. Unsigned Integer. Optional. The maximum requested bandwidth in bits per second for a downlink IP flow
- **signallingProtocol**. Enumerated. Optional. Indicates the protocol used for signalling between the UE and the AF. If this parameter is absent, the value **NO_INFORMATION** is assumed. The signalling protocol is specified as one of the following:
  - **NO_INFORMATION**
  - **SIP**

subscriptionID

These are the parameters for the optional subscriptionID parameter.

- **subscriptionIdType**. Enumerated. Required. Type of the end user's subscription ID. The subscription ID type is specified as one of the following:
  - **END_USER_E164**
  - **END_USER_IMSI**
  - **END_USER_SIP_URI**
  - **END_USER_NAI**
  - **END_USER_PRIVATE**
- **subscriptionIdData**. String. Required. Value of the end user's subscription ID.

supportedFeatures

These are the parameters for the optional supportedFeatures parameter.

- **vendorId**. Unsigned Integer. Required. The vendor ID.
- **featureListID**. Unsigned Integer. Required. The feature list ID.
- **featureList**. Unsigned Integer. Required. A list of the application's supported features.

sponsoredConnectivityData

These are the parameters for the optional sponsoredConnectivityData parameter.

- **sponsorIdentity**. String. Optional. String identifying the sponsor.
- **applicationServiceProviderIdentity**. String. Optional. String identifying the application service provider.
- **grantedServiceUnit**. Complex. Optional. Provides a usage threshold level to the PCRF if the volume of traffic allowed during the sponsored data connectivity is monitored.

  See "grantedServiceUnit" for details on each of the grantedServiceUnit parameters.
- **usedServiceUnit**. Complex. Optional. Provides the number of used units from the point at which the service became active, or, if interim measurements are used during the session, the point at which the previous session ended.

  See "grantedServiceUnit" for details on each of the usedServiceUnit parameters. Note that the parameters are identical to those of grantedServiceUnit.

grantedServiceUnit

These are the parameters for the optional grantedServiceUnit parameter.

- **tariffTimeChange**. Enumerated. Optional. Determines the timing of the unit relative to a tariff time change. The tariff time change parameter can take the following values:
  - **UNIT_BEFORE_TARIFF_CHANGE**
  - **UNIT_AFTER_TARIFF_CHANGE**
  - **UNIT_INDETERMINATE**
- **cCTime**. Unsigned Integer. Optional. Indicates the length of requested, granted or used time in seconds.

- **cCMoney**. Complex. Optional. Specifies the monetary amount in a given currency. See "cCMoney" for details.
- **cCTotalOctets**. Unsigned 64-bit Integer. Optional. Specifies the total number of the granted, requested or used octets, regardless of the flow direction.
- **cCInputOctets**. Unsigned 64-bit Integer. Optional. Specifies the total number of the granted, requested or used octets, that either can be or have been received from an end user.
- **cCOutputOctets**. Unsigned 64-bit Integer. Optional. Specifies the total number of the granted, requested or used octets, that either can be or have been sent to an end user.
- **cCServiceSpecificUnits**. Unsigned 64-bit Integer. Optional. Specifies the number of service specific units provided in a particular service.

cCMoney

These are the parameters for the optional ccMoney parameter.

- **unitValue**. Decimal. Required. Specifies a multiplier that converts between units of a particular unit type and abstract units within the service credit pool.
- **currencyCode**. Unsigned Integer. Optional. Specifies in which currency the cost was given. Must comply with the numeric values defined in the ISO 4217 standard.

Custom AVPs in QoS Requests

In addition to the preset elements, Services Gatekeeper QoS requests can accommodate custom AVP definitions as long as they are supported by the Diameter server. Such custom AVP definitions can be added to the following elements in any number:

- **mediaSubComponent**
- **supportedFeatures**
- **sponsoredConnectivityData**
- **grantedServiceUnit**
- **usedServiceUnit**

Example 14–1 shows a portion of a JSON request specifying a parameter with the name **myCustomDiameterParameter** and the value **My diameter parameter value**.

Example 14–1 Custom JSON AVP Request

```
"parameter": {
  "name": "myCustomDiameterParameter
  "value": "My diameter parameter value"
}
```

Request Example

Example 14–2 shows an example of an apply QoS request body.

Example 14–2 Apply QoS Request Body

```
{
  "qoSFeatureProperties": {
    "duration": 3600,
    "applicationIdentifier": "test_app_id",
```

```
"mediaComponentDescription": [
  {
    "mediaComponentNumber": 1,
    "mediaSubComponent": [
      {
        "flowNumber": 1,
        "flowDescription": [
          "test_flow"
        ],
        "flowStatus": "ENABLED-UPLINK",
        "flowUsage": "NO_INFORMATION",
        "maxRequestedBandwidthUL": 3300,
        "maxRequestedBandwidthDL": 2200,
        "signallingProtocol": "SIP"
      }
    ],
    "applicationIdentifier": "test",
    "mediaType": "AUDIO",
    "maxRequestedBandwidthUL": 1000,
    "maxRequestedBandwidthDL": 1000,
    "minRequestedBandwidthUL": 10,
    "minRequestedBandwidthDL": 10,
    "flowStatus": "ENABLED-UPLINK",
    "reservationPriority": 1,
    "rSBandwidth": 10,
    "rRBandwidth": 10,
    "codecData": [
      "codec"
    ]
  }
],
"serviceInfoStatus": "FINAL_SERVICE_INFORMATION",
"chargingIdentifier": "charging_id",
"sIPForkingIndication": "SINGLE_DIALOGUE",
"subscriptionId": [
  {
    "subscriptionIdType": "END_USER_E164",
    "subscriptionIdData": "861013388991111"
  }
],
"supportedFeatures": [
  {
    "vendorId": 1,
    "featureListID": 1,
    "featureList": 333
  }
],
"reservationPriority": 1,
"framedIPAddress": "0A987898",
"calledStationId": "adsf",
"serviceURN": "counseling",
"sponsoredConnectivityData": {
  "sponsorIdentity": "1adsf",
  "applicationServiceProviderIdentity": "1ss",
  "grantedServiceUnit": {
    "tariffTimeChange": 122,
    "cCTime": 444,
    "cCMoney": {
      "unitValue": 1,
      "currencyCode": 11
```

```
        },
        "cCTotalOctets": 1,
        "cCInputOctets": 1,
        "cCOutputOctets": 1,
        "cCServiceSpecificUnits": 1
      }
    },
    "mPSIdentifier": "mps_id"
  }
}
```

Response Header

In addition to the standard header fields, two additional fields are returned:

- **X-Plugin-Param-Keys**. Comma-separated keys that map to the values returned in X-Plugin-Param-Values. Two values are returned:
  - **AVP_LIST**. Key matching the Avp-List XML structure returned in X-Plugin-Param-Values.
  - **session-id**. Key matching the session ID returned in X-Plugin-Param-Values.
- **X-Plugin-Param-Values**. Comma-separated values that are mapped to their respective keys returned in X-Plugin-Param-Keys.

For the **Avp-List** XML structure, different Diameter servers may return different elements and values, the only required element being the **Result-Code**. For detailed information on the possible elements and values, see the *UMTS Policy and charging control over Rx reference point (ETSI TS 129 214 V10.6.0)* available at http://www.etsi.org/deliver/etsi_ts/129200_129299/129214/10.06.00_60/ts_129214v100600p.pdf.

Likewise, the **session-id** format is dependent upon your Diameter server.

Example 14–3 shows a possible response header.

Example 14–3 Response Header

```
HTTP/1.1 201 Created
Date: Mon, 11 Mar 2013 03:29:11 GMT
Transfer-Encoding: chunked
Location:
http://localhost:8001/ApplicationQoSService/tel%3A88888888/qos/localhost%3B1362972
174%3B0-1362972554169
Content-Type: application/json
X-Plugin-Param-Keys: AVP_LIST,session-id
X-Plugin-Param-Values:
<Avp-List><Session-Id>localhost;1362972174;0</Session-Id><Origin-Host>MINFXU-CN</O
rigin-Host><Origin-Realm>oracle.com</Origin-Realm><Result-Code>2001</Result-Code><
IP-CAN-Type>0</IP-CAN-Type><RAT-Type>0</RAT-Type></Avp-List>,localhost;1362972174;
0
X-Powered-By: Servlet/2.5 JSP/2.1
```

If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The response body contains an array of structures as the value for **applyQoSFeatureResponse**. Each element in the array contains values for the following parameters.

- **requestId**. String. The request ID. Used to uniquely identify the QoS session.
- **actualProperties**. Complex. Additional properties resulting from the request.

  See "actualProperties" for details on each of the actualProperties parameters.

actualProperties

These are the values returned for the actualProperties parameter.

- **accessNetworkChargingIdentifier**. Complex. Contains a charging identifier within the accessNetworkChargingIdentifierValue AVP along with information about the flows transported within the corresponding bearer within the flows AVP. If no flows AVP is provided, the accessNetworkChargingIdentifierValue applies to all flows within the AF session.
- **accessNetworkChargingAddress**. String. Indicates the IP address of the network entity that handles charging within the access network.
- **acceptableServiceInfo**. Complex. Contains the maximum bandwidth for an AF session and/or for specific media components that will be authorized by the PCRF.
- **iPCANType**. Enumerated. Indicates the type of Connectivity Access Network (CAN) to which a user is connected. The CAN type is specified as one of the following:
  - **_3GPP-GPRS**
  - **DOCSIS**
  - **xDSL**
  - **WiMAX**
  - **_3GPP2**
  - **_3GPP-EPS**
  - **Non-3GPP-EPS**
- **rATType**. Identifies the Radio Access Technology (RAT) that is servicing the user equipment. The RAT type is specified as one of the following:
  - **WLAN**
  - **VIRTUAL**
  - **UTRAN**
  - **GERAN**
  - **GAN**
  - **HSPA_EVOLUTION**
  - **EUTRAN**
  - **CDMA2000_1X**
  - **HRPD**
  - **UMB**
  - **EHRPD**
- **flows**. Complex. Indicates IP flows via their flow identifiers.

  See "flows" for details on each of the flows parameters.

- **supportedFeatures**. Complex. See "supportedFeatures" for detailed information.

accessNetworkChargingIdentifier

These are the values returned for the accessNetworkChargingIdentifier parameter.

- **accessNetworkChargingIdentifierValue**. String. Includes the charging identifier.
- **flows**. Complex. Indicates IP flows via their flow identifiers. See "flows" for details on each of the flows parameters.

flows

These are the values returned for the flows parameter.

- **mediaComponentNumber**. Unsigned Integer. Ordinal number of the media component.
- **flowNumber**. Integer. Indicates the number of the flow. If no flowNumber AVP(s) are supplied, this refers to all flows matching the media component number.
- **finalUnitAction**. Enumerated. When reporting an out of credit condition, the finalUnitAction indicates the termination action applied to the impacted flows. Indicates to the credit-control client the action to be taken when a user's account cannot cover the service cost. The final unit action is specified as one of the following:
  - **TERMINATE**
  - **REDIRECT**
  - **RESTRICT_ACCESS**

acceptableServiceInfo

These are the values returned for the acceptableServiceInfo parameter.

- **mediaComponentDescription**. Complex. See "mediaComponentDescription" for detailed information.
- **maxRequestedBandwidthUL**. Unsigned Integer. The maximum requested bandwidth in bits per second for an uplink IP flow.
- **maxRequestedBandwidthDL**. Unsigned Integer. The maximum requested bandwidth in bits per second for a downlink IP flow

Response Body Example

Example 14–4 shows an example of an apply QoS response body.

Example 14–4 Apply QoS Response Body

```
{"applyQoSFeatureResponse": {
    "requestId": "localhost;1362972174;0-1362972554169",
    "actualProperties": {
      "iPCANType": "_3GPP-GPRS",
      "rATType": "WLAN"
    }
  }
}
```

Template-based Apply QoS

The Template-based Apply QoS operation requests that a QoS plan based upon a template stored in Services Gatekeeper be applied to end user IDs as specified in a Services Gatekeeper QoS plug-in regular expression matching rule.

QoS Templates

QoS templates must be formatted according to the XSD found in the **xsd** subdirectory in the **plugin_qos_diameter.jar** file which itself is contained within the **wlng_nt_qos.ear** archive located in *Middleware_Home*/ocsg_5.1/applications directory. Example 14–10 shows a reference template containing all of the possible elements and attributes for a request.

Following is a sample QoS template:

Example 14–5 QoS Template Example

```
<QoSTemplate xmlns="http://oracle/ocsg/rest/qos/template"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://oracle/ocsg/rest/qos/template ../xsd/qosTemplate.xsd"
  templateId="default_template">
  <applicationIdentifier parameterName="$APP_
ID">4d6f62696c655456</applicationIdentifier>
  <!--Zero or more repetitions: -->
  <mediaComponentDescription>
    <mediaComponentNumber>0</mediaComponentNumber>
    <!--Zero or more repetitions: -->
    <mediaSubComponent>
      <flowNumber parameterName="$FLOW_NUMBER_0">1</flowNumber>
      <!--0 to 2 repetitions: -->
      <flowDescription parameterName="$FLOW_DESCRIPTION_0">
 <![CDATA[permit out 8001 from assigned 34 to 24.2.1.6/18
8000]]></flowDescription>
    </mediaSubComponent>
                <mediaType parameterName="$MED_TYPE">VIDEO</mediaType>
    <flowStatus parameterName="$FLOW_STATUS">ENABLED</flowStatus>
  </mediaComponentDescription>
  <!--Optional: -->M
  <serviceInfoStatus parameterName="$SERV_INFO_STATUS">PRELIMINARY_SERVICE_
INFORMATION</serviceInfoStatus>
  <!--Optional: -->
  <chargingIdentifier parameterName="$CHG_ID">charging-id-555</chargingIdentifier>
  <!--Optional: -->
  <sIPForkingIndication parameterName="$SIP_FORK_IND">SINGLE_
DIALOGUE</sIPForkingIndication>
  <subscriptionId>
    <subscriptionIdType parameterName="$SUB_ID_TYPE">END_USER_
E164</subscriptionIdType>
    <subscriptionIdData parameterName="$SUB_ID_
DATA">14128771501</subscriptionIdData>
  </subscriptionId>
  <serviceURN parameterName="$SERV_URN">sos.police</serviceURN>
</QoSTemplate>
```

In Example 14–5, each element has a *parameterName* attribute whose value maps to the request's parameter name. The *parameterName* attribute identifier must be unique throughout the entire template. For example, if you have two instances of the element

applicationIdentifier (one for the whole template and one for a sub-component), you can use the following names for each instance: *$APP_ID_0* and *$APP_ID_1*.

If a parameter is set dynamically in a request, its value replaces the default value configured in the template. For example, if a request sets the *$FLOW_DESCRIPTION_0* parameter value to "modified flow description", the PCRF will receive that value rather than the one defined in the template.

Custom AVPs in QoS Templates

In addition to the preset elements QoS templates can accommodate custom AVP definitions, both simple and enumerated, as long as they are supported by the Diameter server. Such custom AVP definitions can be added to the following elements in any number:

- **mediaSubComponent**
- **supportedFeatures**
- **sponsoredConnectivityData**
- **grantedServiceUnit**
- **usedServiceUnit**

Example 14–6 shows a simple custom AVP template in which the type of the custom parameter is set to **String**, the parameterName is set to **$MY_CUSTOM_DIAMETER_PARAMETER,** and the value of the custom parameter is set to **My Diameter value**.

Example 14–6 Custom AVP Template Element

```
<avp name="myCustomDiameterParameter" description="This is a sample AVP"
code="234567" may-encrypt="true" mandatory-flag="optional" vendor-id="Oracle
Corporation" constrained="false">
   <type type-name="Integer32"/>
   <value parameterName="$MY_CUSTOM_DIAMETER_PARAMETER">"My Diameter
Value"</value>
</avp>
```

Example 14–7 shows a custom enumerated AVP element where the AVP name is **myCustomDiameterParameter**, and two possible enumerated values are defined:

- **ENUM_1**, the logical name associated with the enumerated type, **0**.
- **ENUM_2**, the logical name associated with the enumerated type, **1**.

The parameterName, **$ENUM_VAL**, can be replaced like any standard template parameter with a value of either **ENUM_1** or **ENUM_2**.

> **Note:** For custom enumerated AVPs, the type element's type-name attribute is always **Integer32**.

Example 14–7 Custom Enumerated AVP Template Element

```
<avp name="myCustomEnumDiameterParameter" description="This is a sample AVP"
code="12345" may-encrypt="true" mandatory-flag="required" vendor-id="Oracle
Corporation" constrained="false">
   <type type-name="Integer32"/>
   <enum name="ENUM_1" code="0"/>
   <enum name="ENUM_2" code="1"/>
   <value parameterName="$ENUM_VAL">ENUM_1</value>
</avp>
```

Managing QoS Templates in Services Gatekeeper

You use the Services Gatekeeper administration console or the Platform Test Environment to load, modify and query QoS templates using MBeans. Table 14–1 lists the available MBean operations and their descriptions:

Table 14–1 QoS Template Management MBean Operations

Operation	Description
listQoSRequestTemplateMatchRules	Lists all of the match rules that have been defined for the QoS plug-in.
loadQoSRequestTemplate	Loads a QoS template.
retrieveQoSRequestTemplate	Retrieves a QoS template associated with a particular subscriber ID or a range of subscriber IDs.
deleteQoSRequestTemplate	Deletes a QoS template associated with a particular subscriber ID or a range of subscriber IDs.

For more information on managing QoS templates, see "Managing QoS Templates" in the *Oracle Communications Services Gatekeeper Communication Service Guide*, another document in this document set.

Authorization

Basic or OAuth 2.0

HTTP Method

`POST`

URI

`http://host:port/ApplicationQoSService/${endUserId}/qos/templatebased`

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *endUserId* is a valid end user identifier.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Apply Template-based QoS operation accepts the following parameters:

- **duration**. Unsigned Long. Required. The duration of the applied QoS in seconds.
- **applicationIdentifier**. String. Optional. Identifies the service to which the application-facing service session belongs.
- **framedIPAddress**. Hexadecimal Binary. Optional. The valid routable IPv4 address that is applicable for the IP Flows towards the user equipment at the PCEF.

- **framedIPv6Prefix**. Hexadecimal Binary. Optional. A valid full IPv6 address that is applicable to an IP flow or IP flows towards the user equipment at the PCEF.
- **calledStationId**. String. Optional. If a private IP address is being used, the ID of the packet data network.
- **parameter**. Complex. Optional. An array of JSON objects that define which template parameters will be replaced and what the replacement values will be.

  The array of JSON objects are a collection of one or more AVPs labeled **name** and **value** that determine which parameters in the QoS template will be replaced and what the replacement values will be. In Example 14–8, the values for the template parameters **$CHG_ID** and **$MAX_REQ_BAND_DL** will be replaced with the values **"charging_id_test"** and **2048** respectively.

Request Body Example

Example 14–8 shows an example of an Apply Template-based QoS request body.

Example 14–8 Apply Template-based QoS Request Body

```
{
  "templateQoSFeatureProperties": {
    "duration": 3600,
    "applicationIdentifier": "app_id",
    "framedIPAddress": "0A0B9899",
    "calledStationId": "called_station_id",
    "parameter": [
      {
        "name": "$CHG_ID",
        "value": "charging_id_test"
      },
      {
        "name": "$MAX_REQ_BAND_DL",
        "value": 2048
      }
    ]
  }
}
```

Response Header

For details on the response header, see the Apply QoS "Response Header" section.

Response Body

The response body parameters for the Template-based Apply QoS request are the same as those for the Apply QoS operation. See the Apply QoS "Response Body" section for details.

Response Body Example

Example 14–9 shows an example of Template-based Apply QoS response body.

Example 14–9 Template-based Apply QoS Response Body

```
{
  "applyQoSFeatureResponse": {
    "requestId": "localhost;1362972174;1-1362973261091",
    "actualProperties": {
      "iPCANType": "_3GPP-GPRS",
```

```
      "rATType": "WLAN"
    }
  }
}
```

Reference: Complete QoS Template

Example 14–10 is a QoS template containing all of the available elements and attributes from the reference XSD.

Example 14–10 A Complete QoS Template

```
<QoSTemplate xmlns="http://oracle/ocsg/rest/qos/template"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://oracle/ocsg/rest/qos/template ../xsd/qosTemplate.xsd"
  templateId="default_template">
  <applicationIdentifier parameterName="$APP_ID">test_
appId</applicationIdentifier>
  <!--Zero or more repetitions: -->
  <mediaComponentDescription>
    <mediaComponentNumber>0</mediaComponentNumber>
    <!--Zero or more repetitions: -->
    <mediaSubComponent>
      <flowNumber parameterName="$FLOW_NUMBER_0">0</flowNumber>
      <!--0 to 2 repetitions: -->
      <flowDescription parameterName="$FLOW_DESCRIPTION_0">flow0_
Description</flowDescription>
      <!--Optional: -->
      <flowStatus parameterName="$FLOW_STATUS_0">ENABLED</flowStatus>
      <!--Optional: -->
      <flowUsage parameterName="$FLOW_USAGE_0">NO_INFORMATION</flowUsage>
      <!--Optional: -->
      <maxRequestedBandwidthUL parameterName="$MAX_REQ_BAND_UL_
0">1000</maxRequestedBandwidthUL>
      <!--Optional: -->
      <maxRequestedBandwidthDL parameterName="$MAX_REQ_BAND_DL_
0">1000</maxRequestedBandwidthDL>
      <!--Optional: -->
      <signallingProtocol parameterName="$SIG_PROTOCOL_0">SIP</signallingProtocol>
    </mediaSubComponent>
    <!--Optional: -->
    <applicationIdentifier parameterName="$MED_DES_APP_ID">test_
appId</applicationIdentifier>
    <!--Optional: -->
    <mediaType parameterName="$MED_TYPE">AUDIO</mediaType>
    <!--Optional: -->
    <maxRequestedBandwidthUL parameterName="$MAX_REQ_BAND_
UL">1000</maxRequestedBandwidthUL>
    <!--Optional: -->
    <maxRequestedBandwidthDL parameterName="$MAX_REQ_BAND_
DL">1000</maxRequestedBandwidthDL>
    <!--Optional: -->
    <minRequestedBandwidthUL parameterName="$MIN_REQ_BAND_
UL">10</minRequestedBandwidthUL>
    <!--Optional: -->
    <minRequestedBandwidthDL parameterName="$MIN_REQ_BAND_
UL">10</minRequestedBandwidthDL>
    <!--Optional: -->
    <flowStatus parameterName="$FLOW_STATUS">ENABLED</flowStatus>
    <!--Optional: -->
```

```
    <reservationPriority parameterName="$RES_PRI">0</reservationPriority>
    <!--Optional: -->
    <rSBandwidth parameterName="$RS_BAND">1000</rSBandwidth>
    <!--Optional: -->
    <rRBandwidth parameterName="$RR_BAND">1000</rRBandwidth>
    <!--0 to 2 repetitions: -->
    <codecData parameterName="$CODEC_DATA">CODEC</codecData>
  </mediaComponentDescription>
  <!--Optional: -->
  <serviceInfoStatus parameterName="$SERV_INFO_STATUS">FINAL_SERVICE_
INFORMATION</serviceInfoStatus>
  <!--Optional: -->
  <chargingIdentifier parameterName="$CHG_ID">test_charging</chargingIdentifier>
  <!--Optional: -->
  <sIPForkingIndication parameterName="$SIP_FORK_IND">SINGLE_
DIALOGUE</sIPForkingIndication>
  <!--Zero or more repetitions: -->
  <subscriptionId>
    <subscriptionIdType parameterName="$SUB_ID_TYPE">END_USER_
E164</subscriptionIdType>
    <subscriptionIdData parameterName="$SUB_ID_
DATA">13693312888</subscriptionIdData>
  </subscriptionId>
  <!--Zero or more repetitions: -->
  <supportedFeatures>
    <vendorId parameterName="$VENDOR_ID">654321</vendorId>
    <featureListID parameterName="$FEATURE_LIST_ID">654320</featureListID>
    <featureList parameterName="$FEATURE_LIST">654322</featureList>
    <!--Zero or more repetitions: -->
    <avp name="test_supported_feature" description="test supported feature avp"
      code="688788" may-encrypt="true" mandatory-flag="required" vendor-id="87349"
      constrained="false">
      <grouped>
        <!--1 or more repetitions: -->
        <gavp name="grouped_avp" />
      </grouped>
      <avps name="avps_name" code="96785">
        <type type-name="String"/>
        <value parameterName="$customer_avps_name">xmf</value>
      </avps>
    </avp>
  </supportedFeatures>
  <!--Optional: -->
  <reservationPriority parameterName="$RESV_PRI">0</reservationPriority>
  <!--Optional: -->
  <framedIPAddress></framedIPAddress>
  <!--Optional: -->
  <framedIPv6Prefix></framedIPv6Prefix>
  <!--Optional: -->
  <calledStationId></calledStationId>
  <!--Optional: -->
  <serviceURN parameterName="$SERV_URN">sos.fire</serviceURN>
  <!--Optional: -->
  <sponsoredConnectivityData>
    <!--Optional: -->
    <sponsorIdentity parameterName="$SPON_ID">spon_id</sponsorIdentity>
    <!--Optional: -->
    <applicationServiceProviderIdentity
      parameterName="$SPON_APP_SERV_PROV_ID">spon_serv_prov_
id</applicationServiceProviderIdentity>
```

```
    <!--Optional: -->
    <grantedServiceUnit>
      <!--Optional: -->
      <tariffTimeChange parameterName="$TARIF_TIME_CHG">1</tariffTimeChange>
      <!--Optional: -->
      <cCTime parameterName="$CC_TIME">60</cCTime>
      <!--Optional: -->
      <cCMoney>
        <unitValue parameterName="$UNIT_VAL">6.28</unitValue>
        <!--Optional: -->
        <currencyCode parameterName="$CUR_CODE">80</currencyCode>
      </cCMoney>
      <!--Optional: -->
      <cCTotalOctets parameterName="$CC_TOTAL_OCTS">1000000</cCTotalOctets>
      <!--Optional: -->
      <cCInputOctets parameterName="$CC_INPUT_OCTS">500000</cCInputOctets>
      <!--Optional: -->
      <cCOutputOctets parameterName="$CC_OUTPUT_OCTS">500000</cCOutputOctets>
      <!--Optional: -->
      <cCServiceSpecificUnits parameterName="$CC_SERV_SPEC_
UNIT">1</cCServiceSpecificUnits>
    </grantedServiceUnit>
  </sponsoredConnectivityData>
  <!--Optional: -->
  <mPSIdentifier parameterName="$MPS_ID">mps_id</mPSIdentifier>
  <!--Zero or more repetitions: -->
  <avp name="myRandomAVP" description="This is a sample AVP" code="93222"
may-encrypt="true" mandatory-flag="required" vendor-id="Oracle Corporation"
constrained="false">
    <type type-name="Integer32"/>
    <enum name="ENUM_1" code="0"/>
    <enum name="ENUM_2" code="1"/>
    <value parameterName="$ENUM_VAL">ENUM_1</value>
  </avp>
</QoSTemplate>
```

Modify QoS

The Modify QoS operation lets you modify the parameters of an existing QoS plan.

Authorization

Basic or OAuth 2.0

HTTP Method

`PUT`

URI

`http://host:port/ApplicationQoSService/qos/${requestId}`

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *requestId* is a valid QoS request ID.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

To modify a QoS session, create a request body with the parameters you want changed. See the Apply QoS "Request Body" section for a complete listing of request body parameters.

Request Body Example

Example 14–11 shows an example of a modify QoS request body.

Example 14–11 Modify QoS Request Body

```
{
  "qoSFeatureProperties": {
    "applicationIdentifier": "654321"
  }
}
```

Response Header

For details on the response header, see the Apply QoS "Response Header" section.

Response Body

The response body contains an array of structures as the value for **actualProperties**. See "actualProperties" for complete details.

Response Body Example

Example 14–12 shows an example of a modify QoS response body.

Example 14–12 Modify QoS Response Body

```
{
  "actualProperties": {
    "iPCANType": "_3GPP-GPRS",
    "rATType": "WLAN"
  }
}
```

Modify Template-based QoS

The Modify Template-based QoS operation requests a modification to an existing template-based QoS plan.

Authorization

Basic or OAuth 2.0

HTTP Method

`POST`

URI

`http://host:port/ApplicationQoSService/qos/${requestId}/templatebased`

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *requestId* is a valid QoS request ID.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

To modify a template-based QoS session, create a request body with the parameters you want changed. See the Apply Template-based QoS "Request Body" section for a complete listing of request body parameters.

Request Body Example

Example 14–13 shows an example of a modify template-based QoS request body.

Example 14–13 Modify Template-based QoS Request Body

```
{
  "templateQoSFeatureProperties": {
    "parameter": [
      {
        "name": "$FLOW_DESCRIPTION_0",
        "value": "permit out 8001 from assigned 34 to 24.2.1.6/18 8000"
      }
    ]
  }
}
```

Response Header

For details on the response header, see the Apply QoS "Response Header" section.

Response Body

See "actualProperties" for details on the response body parameters for a Modify Template-based QoS requests.

Response Body Example

Example 14–14 shows an example of a Modify Template-based QoS response body.

Example 14–14 Modify Template-based QoS Response Body

```
{
  "actualProperties": {
    "iPCANType": "_3GPP-GPRS",
    "rATType": "WLAN"
  }
}
```

Get QoS Status

The Get QoS operation returns detailed information on the currently applied QoS plan.

Authorization

Basic or OAuth 2.0

HTTP Method

`GET`

URI

`http://host:port/ApplicationQoSService/qos/${requestId}`

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *requestId* is a valid QoS session identifier.

Request Header

Standard header fields.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The response body contains an array of structures as the value for **applyQoSFeatureResponse**. Each element in the array contains values for the following parameters.

- **userIsConnected**. Boolean. True indicates a user is connected. False indicates a user is disconnected.
- **actualProperties**. Complex. Additional properties resulting from the request.

  See "actualProperties" for details on each of the actualProperties parameters.
- **qosFeatureProperties**. Complex. Additional properties resulting from the request.

  See the "Request Body" section of the "Apply QoS" operation for details on each of the qosFeatureProperties parameters.

Response Body Example

Example 14–15 shows an example of a QoS status response body.

Example 14–15 QoS Status Response Body

```
{
  "qoSStatus": {
    "userIsConnected": true,
    "actualProperties": {
      "iPCANType": "_3GPP-GPRS",
      "rATType": "WLAN"
    },
    "qoSFeatureProperties": {
      "duration": 3600,
      "applicationIdentifier": "654321",
      "mediaComponentDescription": [
        {
          "mediaComponentNumber": 0,
          "mediaSubComponent": [
            {
              "flowNumber": 1,
              "flowDescription": [
                "flow_1"
              ],
              "flowStatus": "ENABLED-UPLINK",
              "flowUsage": "NO_INFORMATION",
              "maxRequestedBandwidthUL": 100,
              "maxRequestedBandwidthDL": 500,
              "signallingProtocol": "NO_INFORMATION"
            }
          ],
          "applicationIdentifier": "test_app_id",
          "mediaType": "VIDEO",
          "maxRequestedBandwidthUL": 200,
          "maxRequestedBandwidthDL": 1000,
          "minRequestedBandwidthUL": 10,
          "minRequestedBandwidthDL": 100,
          "flowStatus": "ENABLED",
          "reservationPriority": 1
        }
      ],
      "serviceInfoStatus": "FINAL_SERVICE_INFORMATION",
      "chargingIdentifier": "charging_id",
      "subscriptionId": [
        {
          "subscriptionIdType": "END_USER_E164",
          "subscriptionIdData": "88888888"
        }
      ],
      "framedIPAddress": "0A999899",
      "calledStationId": "EE-AA-CD-AF-09"
    }
  }
}
```

Remove QoS

The Remove QoS operation removes a current QoS plan identified by a requestId.

Authorization

Basic or OAuth 2.0

HTTP Method

`DELETE`

URI

`http://host:port/ApplicationQoSService/qos/${requestId}`

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *requestId* is a valid QoS session identifier.

Request Header

The MIME-type for the Content-Type header field is **application/x-www-form-urlencoded**.

Request Body

There is no request body.

Response Header

For details on the response header, see the Apply QoS "Response Header" section.

Response Body

There is no response body.

Register for QoS Notifications

The Register for QoS Notifications operation lets an application register to receive QoS events.

Table 14–2 shows the events an application can register to receive.

Table 14–2 QoS Event Types

Event Type	Trigger Condition
QOS_FEATURE_RELEASED	Triggered when the AF session times out. Controlled by the duration parameter in the QoS session request.
CHARGING_CORRELATION_EXCHANGE	If different Access Network Charging Information is applicable to the IP-CAN session, the PCRF will notify the AF about the Access Network Charging Information that applies to each authorized flow.
INDICATION_OF_LOSS_OF_BEARER	The server reports a loss of a bearer to the AF.
INDICATION_OF_RECOVERY_OF_ BEARER	The server reports a recovery of a bearer to the AF.
INDICATION_OF_RELEASE_OF_ BEARER	The server reports the release of a bearer to the AF.
IP_CAN_CHANGE	The server indicates a change in the IP-CAN type or RAT type (if the IP-CAN type is GPRS).
INDICATION_OF_OUT_OF_CREDIT	The PCRF reports to the AF that the SDFs have run out of credit and that the termination action determined by the finalUnitAction AVP applies.
INDICATION_OF_SUCCESSFUL_ RESOURCES_ALLOCATION	The PCRF reports that the requested resources have been successfully allocated.
INDICATION_OF_FAILED_ RESOURCES_ALLOCATION	The PCRF reports that the requested resources could not be successfully allocated.
INDICATION_OF_LIMITED_PCC_ DEPLOYMENT	The server reports limited PCC deployment—dynamically allocated resources are not available.
USAGE_REPORT	The PCRF reports accumulated usage volume when the usage threshold provided by the AF has been reached.

Authorization

Basic or OAuth 2.0

HTTP Method

POST

URI

`http://`*host*`:`*port*`/ApplicationQoSNotification/registration`

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Apply QoS operation accepts the following parameters:

- **reference**. Complex. Required. Defines the application endpoint, interfaceName and correlator used to notify the application.
- **endUserIdentities**. Array. Required. Network end user(s) to be monitored for events. An array of one or more end user IDs.
- **eventCriteria**. Array. Required. Event(s) to be monitored. Events are specified by their numeric event ID. Table 14–3 shows the mapping of numeric event IDs to logical event IDs. For definitions of these events, see Table 14–2.

Table 14–3 Numeric to Logical QoS Event Mapping

Numeric Event	Logical Event Name
1	CHARGING_CORRELATION_EXCHANGE
2	INDICATION_OF_LOSS_OF_BEARER
3	INDICATION_OF_RECOVERY_OF_BEARER
4	INDICATION_OF_RELEASE_OF_BEARER
6	IP-CAN_CHANGE
7	INDICATION_OF_OUT_OF_CREDIT
8	INDICATION_OF_SUCCESSFUL_RESOURCES_ALLOCATION
9	INDICATION_OF_FAILED_RESOURCES_ALLOCATION
10	INDICATION_OF_LIMITED_PCC_DEPLOYMENT
11	USAGE_REPORT
12	QOS_FEATURE_RELEASED

reference

These are the parameters for the reference parameter.

- **endpoint**. String. Required. The notification end point expected by the AF.
- **interfaceName**. String. Required. The interface name. Rarely used.
- **correlator**. String. Required. A unique ID for the message.

Request Body Example

Example 14–16 shows an example of a Register for QoS Notifications request body.

Example 14–16 Register for QoS Notifications Request Body

```
{
  "startQoSNotification": {
    "reference": {
      "endpoint": "http://10.182.105.66:13444/jaxrs/QoSNotification",
      "interfaceName": "interfaceName",
      "correlator": "987654321"
    },
    "endUserIdentities": [
```

```
        "tel:88888888", "tel:123456", "tel:234567"
      ],
      "eventCriteria": [
        "1", "2", "6", "7"
      ]
    }
}
```

Response header

For details on the response header, see the Apply QoS "Response Header" section.

Response Body

There is no response body.

Unregister for QoS Notifications

The Unregister for QoS Notifications operation requests that the server cease sending an application QoS notifications.

Authorization

Basic or OAuth 2.0

HTTP Method

`DELETE`

URI

`http://host:port/ApplicationQoSNotification/registration/${correlator}`

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *correlator* is the unique identifier of the original notification request.

Request Header

The MIME-type for the Content-Type header field is **application/x-www-form-urlencoded**.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body.

QoS Event Notification

A QoS event notification is generated by a PCRF when an event is triggered for which an application has registered to receive notifications.

Authorization

Basic or OAuth 2.0

HTTP Method

`POST`

URI

`http://host:port/QoSNotification`

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the QoS Event Notification operation accepts the following parameters:

- **correlator**. String. Required. The duration of the applied QoS in seconds.
- **endUserIdentities**. URI. Required. Identifies the service to which the application-facing service session belongs.
- **eventType**. Complex. Required. The valid routable IPv4 address that is applicable for the IP Flows towards the user equipment at the PCEF.
- **subscriptionId**. Complex. Optional. A valid full IPv6 address that is applicable to an IP flow or IP flows towards the user equipment at the PCEF.
- **accessNetworkChargingIdentifier**. Complex. Optional. Contains a charging identifier within the accessNetworkChargingIdentifierValue AVP along with information about the flows transported within the corresponding bearer within the flows AVP. If no flows AVP is provided, the accessNetworkChargingIdentifierValue applies to all flows within the AF session.

  See "accessNetworkChargingIdentifier" for details on each of the accessNetworkChargingIdentifier parameters.
- **accessNetworkChargingAddress**. String. Optional. Indicates the IP address of the network entity that handles charging within the access network.
- **iPCANType**. Enumerated. Optional. Indicates the type of Connectivity Access Network (CAN) to which a user is connected. The CAN type is specified as one of the following:
  - **_3GPP-GPRS**
  - **DOCSIS**

  - **xDSL**
  - **WiMAX**
  - **_3GPP2**
  - **_3GPP-EPS**
  - **Non-3GPP-EPS**
- **rATType**. Enumerated. Optional. Identifies the Radio Access Technology (RAT) that is servicing the user equipment. The RAT type is specified as one of the following:
  - **WLAN**
  - **VIRTUAL**
  - **UTRAN**
  - **GERAN**
  - **GAN**
  - **HSPA_EVOLUTION**
  - **EUTRAN**
  - **CDMA2000_1X**
  - **HRPD**
  - **UMB**
  - **EHRPD**
- **flows**. Complex. Indicates IP flows via their flow identifiers.

  See "flows" for details on each of the flows parameters.
- **abortCause**. Enumerated. Optional. Determines the cause of an Abort Session Request (ASR) or of a Resource Access Restriction (RAR) indicating a bearer release. The abort cause is specified as one of the following:
  - **BEARER_RELEASED**
  - **INSUFFICIENT_SERVER_RESOURCES**
  - **INSUFFICIENT_BEARER_RESOURCES**
  - **PS_TO_CS_HANDOVER**
  - **SPONSORED_DATA_CONNECTIVITY_DISALLOWED**
- **sponsoredConnectivityData**. Complex. Optional. A set of AVPs that define which template parameters will be replaced and what the replacement values will be.

  See "sponsoredConnectivityData" for details on each of the sponsoredConnectivityData parameters.

Request Example

Example 14–17 shows an example of a QoS Event Notification request body.

Example 14–17 Notify QoS Event Request Body

```
{
  "notifyQoSEvent": {
    "correlator": "987654321",
    "eventType": 6,
```

```
    "accessNetworkChargingAddress": "127.0.0.1",
    "iPCANType": "WiMAX",
    "rATType": "WLAN"
  }
}
```

Response header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body.

List QoS Event Notifications

The List QoS Event operation returns a list of QoS events that are registered either to a single correlator, if a correlator is specified in the request, or a complete list of registered events for all correlators if no correlator is specified.

Authorization

Basic or OAuth 2.0

HTTP Method

`GET`

URI

`http://host:port/ApplicationQoSNotification/registration/[${correlator}]`

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *correlator* is the optional unique identifier of the original notification request.

Request Header

Standard request headers.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body with No Correlator

The response body for the List QoS Event operation when an optional correlator parameter is specified, returns the following parameters:

- **correlators**. Array. An array of correlators that are registered to receive QoS events.

Response Body Example

Example 14–18 shows an example of a List QoS Notifications response body when no correlator is specified in the request.

Example 14–18 List QoS Notifications Response Body without a Correlator

```
{
  "correlators": {
    "correlator": [
      "987654321", "12345676", "1272638"
    ]
  }
```

```
}
```

Response Body with Optional Correlator

The response body for the List QoS Event operation when an optional correlator parameter is specified, returns the following parameters:

- **reference**. Complex. Defines the application endpoint, interfaceName and correlator used to notify the application. See "reference" for detailed information on the reference parameters.
- **endUserIdentities**. Array. Network end user(s) to be monitored for events. An array of one or more end user IDs.
- **eventCriteria**. Array. Event(s) to be monitored. Events are specified by their numeric event ID. Table 14–3 shows the mapping of numeric event IDs to logical event IDs. Table 14–2 lists these event IDs.

Table 14–4 Numeric to Logical QoS Event Mapping

Numeric Event	Logical Event Name
1	CHARGING_CORRELATION_EXCHANGE
2	INDICATION_OF_LOSS_OF_BEARER
3	INDICATION_OF_RECOVERY_OF_BEARER
4	INDICATION_OF_RELEASE_OF_BEARER
6	IP-CAN_CHANGE
7	INDICATION_OF_OUT_OF_CREDIT
8	INDICATION_OF_SUCCESSFUL_RESOURCES_ALLOCATION
9	INDICATION_OF_FAILED_RESOURCES_ALLOCATION
10	INDICATION_OF_LIMITED_PCC_DEPLOYMENT
11	USAGE_REPORT
12	QOS_FEATURE_RELEASED

reference

These are the parameters for the reference parameter.

- **endpoint**. String. The notification end point expected by the AF.
- **interfaceName**. String. The interface name. Rarely used.
- **correlator**. String. A unique ID for the message.

Response Body Example

Example 14–19 shows an example of a List QoS Notifications response body when a correlator is specified in the request.

Example 14–19 List QoS Notifications Response Body with Correlator

```
{
  "startQoSNotification": {
    "reference": {
      "endpoint": "http://10.182.105.66:13444/jaxrs/QoSNotification",
      "correlator": "987654321"
    },
    "endUserIdentities": [
```

```
      "tel:88888888"
    ],
    "eventCriteria": [
      "6", "7", "1", "2"
    ]
  }
}
```

15

Presence

This chapter describes the operations in the Presence interface of the RESTful Web Services provided in Oracle Communications Services Gatekeeper.

About the Presence Interface

Applications use the RESTful Presence interface to act as either of two different parties to a presence interaction: as a presentity or as a watcher.

REST Service Descriptions Available at Run-time

When the Administration Server for your Services Gatekeeper domain is in the running state, the REST service descriptions of these operations can be found at

`http://host:port/rest/presence/index.html`

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

About Presentities and Watchers

A presentity agrees to have certain data (called *attributes*) such as current activity, available communication means, and contact addresses made available to others. 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.

A watcher is the consumer of current activity, available communication means, and contact addresses made available by a presentity. 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.

Data Common to Operations in RESTful Presence Interface

This section describes the following data objects that are commonly used in the JSON data structures in RESTful Presence interface operations. They are described in detail here to avoid repeating the information.

attributes

The term **attributes** is used to describe the current attributes of a presentity. It specifies the single Presence attribute or the set of Presence attributes that a watcher might wish

to view for a specified presentity. It is used in the request body for Subscribe Presence and Start Presence Notifications, the Request-URI query for Get User Presence and the response body for Get Open Subscriptions,

The following structure is used to specify the array of values for **attributes**:

```
"attributes": ["Activity|Place|Privacy|Sphere|Communication|Other"
```

Each element of the array denotes a single Presence attribute and in turn, represented by a JSON attribute-value pair (or a JSON data object). An empty array indicates that the watcher is interested in all of the Presence attributes associated with a presentity.

PresenceAttribute

The term **presenceAttribute** indicates the type of presentity data that must be updated or changed for the watcher. It is used in the request body for Update Subscription Authorization and in subscription notifications.

The values **presenceAttribute** accepts are specified in the following name/value pair structure:

```
"presenceAttribute": ["Activity|Place|Privacy|Sphere|Communication|Other"
```

unionElement

The term **unionElement** indicates the type of presentity data that is presented. It is used in the request body for Publish, the response body for Get User Presence and in notifications for a change in status.

The values **unionElement** accepts are specified in the following name/value pair structure:

```
"unionElement": ["Activity|Place|Privacy|Sphere|Communication|Other"
```

Activity

The term **activity** specifies the presentity's current activity. It is one of the values accepted by **attributes**. The values **activity** accepts are specified in the following name/value pair structure:

```
"activity":
"ActivityNone|Available|Busy|DoNotDisturb|OnThePhone|Steering|Meeting|Away|Meal|Pe
rmanentAbsence|Holiday|Performance|InTransit|Travel|Sleeping|ActivityOther
```

where:

- **ActivityNone** indicates that the value has not been set.
- **ActivityOther** refers to any non-listed activity type.

Place

The **place** attribute specifies the current type of location for a presentity. It is one of the values accepted by the **attributes** attribute. The values **place** accepts are specified in the following name/value pair structure:

```
"place":
"PlaceNone|Home|Office|PublicTransport|Street|Outdoors|PublicPlace|Hotel|Theatre|R
estaurant|School|Industrial|Quiet|Noisy|Aircraft|Ship|Bus|Station|Mall|Airport|Tra
in|PlaceOther"
```

where:

- **PlaceNone** indicates that the value has not been set.
- **PlaceOther** refers to any other type of place not listed here.

Privacy

The **privacy** attribute specifies the level of privacy in the presentity's current environment. It is one of the values accepted by the **attributes** attribute. The values **privacy** accepts are specified in the following name/value pair structure:

```
"privacy": "PrivacyNone|PrivacyPublic|PrivacyPrivate|PrivacyQuiet|PrivacyOther"
```

where:

- **PrivacyNone** indicates that the value has not been set.
- **PrivacyOther** refers to any other level of privacy not listed here.

Sphere

The **sphere** attribute specifies the sphere within which the presentity is currently acting. It is one of the values accepted by the **attributes** attribute. The values **sphere** accepts are specified in the following name/value pair structure:

```
"sphere": "SphereNone|SphereWork|SphereHome|SphereOther"
```

where:

- **SphereNone** indicates that the sphere has not been set.)
- **SphereOther** is used to refer to any other type of sphere not listed here.

Type

The **type** attribute specifies the type of contact client for a specified or preferred form of communication. It is one of the values accepted by the **means** JSON data object. The values **type** accepts are specified in the following name/value pair structure:

```
"type": "Phone|Chat|Sms|Video|Web|Email|Mms|MeansOther"
```

Communication

The **communication** attribute specifies the connection information for a presentity's preferred form of communication. It is one of the values accepted by **attributes**, **presenceAttribute** and **unionElement**. It accepts only one value, **means**, specified in the following structure:

```
"communication": {"means": [{
      "contact": "URI",
      "priority": "Float",
      "type": "Phone|Chat|Sms|Video|Web|Email|Mms|MeansOther"
    }]
}
```

Means

The **means** object specifies the parameters for the preferred form of communication. It accepts the following parameters:

- **contact**. String. Required. The contact address for this particular means or form of communications, as a URI.
- **priority**. Number (floating point). required parameter. The priority of this particular means. Valid entries start at 0.0 for the lowest priority and range in increasing order to 1.0 as the value for highest priority.
- **type**. Described earlier. See "Type".

The following structure is used to specify the array of values for **means**:

```
"means": [{
      "contact": "URI",
      "priority": "Float",
      "type": "Phone|Chat|Sms|Video|Web|Email|Mms|MeansOther"
    }]
```

TypeandValue

The **typeAndValue** JSON object contains the presence information for a presentity. It is used in **Get User Presence** and Publish operations and in the Presence notifications. It is made up of the following parameters (each of which has been described earlier):

- **unionElement**. String. Required. See "unionElement".
- **activity**. String. Optional. Present if the value for **unionElement** is **Activity**. See "Activity".
- **place**. String. Optional. Present if the value for **unionElement** is **Place**. See "Place".
- **privacy**. String. Optional. Present if the value for **unionElement** is **Privacy**. See "Privacy".
- **sphere**. String. Optional. Present if the value for **unionElement** is **Sphere**. See "Sphere".
- **communication**: JSON object. Optional. Present if the value for **unionElement** is **Communication**. See "Communication".
- **other**: JSON object. Optional. See "Other".

The following structure is used to specify the values for **typeAndValue**:

```
"typeAndValue": {
    "unionElement": "Activity|Place|Privacy|Sphere|Communication|Other",
    "activity":
"ActivityNone|Available|Busy|DoNotDisturb|OnThePhone|Steering|Meeting|Away|Meal|Pe
rmanentAbsence|Holiday|Performance|InTransit|Travel|Sleeping|ActivityOther",
    "communication": {"means": [{
      "contact": "URI",
      "priority": "Float",
      "type": "Phone|Chat|Sms|Video|Web|Email|Mms|MeansOther"
    }]
}
```

Other

The **other** JSON object specifies the parameters for arbitrary information. It has the following parameters:

- **name**. String. Required. The name for this specific information.
- **value**. String. Required. The specific value to be matched for the `name`.

The following structure is used to specify the values for **typeAndValue**:

```
"other": {
      "name": "String",
      "value": "String"
    }
```

Get Open Subscriptions

The Get Open Subscription operation polls for any watchers who wish to subscribe to this presentity's data.

To retrieve the information on the applications, the **subscription?status=open** query string is included in the Request-URI of the GET method.

If the Get Open Subscription operation is successful, the response body contains the **result** JSON data object with the information on the watchers.

Authorization

Basic

HTTP Method

`GET`

URI

`http://host:port/rest/presence/subscription?status=open`

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- the `subscription?status=open` query string is used to retrieve any watchers who wish to subscribe to this presentity's data.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The information on the watchers is returned as an array of values for the **result** attribute. Each element in the array contains the following attribute-value pairs.

- **application**. String. Information only. A descriptive name for the application that operates on behalf of the watcher.
- **attributes**. String. The array of Presence attributes that the watcher wishes to see. See "attributes".
- **watcher**. String. The address (as a URI) of the watcher making the request.

The response body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{"result": [{
  "application": "String",
  "attributes": ["Activity|Place|Privacy|Sphere|Communication|Other"],
  "watcher": "URI"
}]}
```

Get My Watchers

The Get My Watchers operation retrieves an array of current watchers.

To retrieve the information on the current watchers, include the `subscription?filter=watcher` query string in the Request-URI of the GET method.

If the Get My Watchers operation is successful, the response body contains the `result` JSON data object with the list of current watchers.

Authorization

Basic

HTTP Method

`GET`

URI

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

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- the `subscription?filter=watcher` query string is used to retrieve the array of current watchers.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The response body contains a JSON data object as the value for the **result** entry. The value for **result** is an array of URIs which represent the addresses of the set of current watchers:

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

Update Subscription Authorization

The Update Subscription Authorization operation is used to add watchers who have recently asked for subscriptions or to change permissions for any current watchers.

To do so, provide the address of the watcher along with the decision on whether permission must be granted, and the Presence attribute for which such permission must be granted (or revoked).

If the request fails, the Status-Line header field will contain the status code and the reason for the failure.

Authorization

Basic

HTTP Method

`PUT`

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Update Subscription Authorization operation accepts the following parameters:

- **watcher**. String. Required. The URI of the watcher.
- **decisions**: Array of JSON objects. Required. Each JSON object in the array contains the following parameters:
  - **presenceAttribute**. String. Required. The specific Presence attribute for which permission is to be granted (to a new watcher) or permission must be changed (for the current watcher). See "PresenceAttribute".
  - **decision**: Boolean. Required. If the value is `true`, permission is granted (to a new watcher) or permission is changed (for the current watcher).

The request body is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "decisions": [{
    "decision": "Boolean",
    "presenceAttribute": "Activity|Place|Privacy|Sphere|Communication|Other"
  }],
  "watcher": "URI"
}
```

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body.

Subscribe Presence

The Subscribe Presence operation requests a subscription to a presentity's data. No data will be available until the presentity authorizes the watcher using Update Subscription Authorization.

To request a subscription to a presentity's data, provide the address of the presentity or group whose data is being requested, the Presence attributes this watcher wishes to see. Because the actual data depends on whether the subscription is authorized, a reference data object must also be provided in the request body. The reference object (also a JSON object) contains the correlator for the notification, the **endpoint** address (a specific Bayeux channel name) to which the notifications must be sent and, optionally, the interface name (a string to identify the notification).

If the request fails, the Status-Line header field will contain the status code and the reason for the failure.

Authorization

Basic

HTTP Method

`PUT`

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Subscribe Presence operation accepts the following parameters:

- **application**. String. Required. A descriptive name for the application whose data the watcher wishes to access. Informational only.
- **presentity**: a JSON object. Required. The address of the presentity whose data is requested (as a URI).
- **attributes**. An array of string values. Optional. The set of Presence attributes the watcher wishes to see. An empty array (or if this attribute is not present) indicates the watcher wishes to view all the attributes. For a description of the attributes, see "attributes".
- **reference**: JSON object. Required. Use this object to provide the following information about the endpoint that is to receive the notification:
  - **correlator**. String. Required. The correlator used to identify the notification.
  - **endpoint**. String. Required. The URI which represents the endpoint address to which the notification should be delivered. This string should be a Bayeux

protocol channel name that begins with **/bayeux/***appInstanceID* where *appInstanceID* is the client application's application instance account ID.

For more information on application instances, see the discussion on creating and maintaining service provider and application accounts in *Accounts and SLAs Guide*.

– **interfaceName**. String. Required. A descriptive string to identify the type of notification.

The request body is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "application": "String",
  "presentity": "URI",
  "reference": {
    "correlator": "String",
    "endpoint": "URI",
    "interfaceName": "String"
  },
  "attributes": ["Activity|Place|Privacy|Sphere|Communication|Other"]
}
```

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body.

Block Subscription

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

To block a watcher's subscription to a presentity's data, provide the address of the watcher as a URI in the Request-URI of the DELETE method.

If the request fails, the Status-Line header field will contain the status code and the reason for the failure.

Authorization

Basic

HTTP Method

`DELETE`

URI

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

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- the *${watcher}* is the URI of the watcher who must be blocked from viewing this presentity's data.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body.

Start Presence Notification

The Start Presence Notification operation begins delivering presence data to the endpoint defined in the reference attribute. This operation is only functional if the presentity has approved the watcher.

To set up for such notifications, provide the SIP-formatted URI of the presentity for which the watcher must receive notifications, the Presence attributes which will trigger the notifications and a reference object for the delivery of the notifications. The reference object (also a JSON object) contains the correlator for the notification, the **endpoint** address (a specific Bayeux channel name) to which the notifications must be sent and, optionally, the interface name (a string to identify the notification). Additionally, you can specify the request frequency, the total number of notifications and the duration for the notification and whether the check must start immediately.

If the Start Presence Notification request is successful, the endpoint address specified in the request body will receive a notification when:

- the presentity has authorized the watcher. See "Notification of Subscription to a Presentity's Data".
- the presentity's status changes. See "Notification of a Status Change for a Presentity".
- the notification limit or the specified duration has been reached. See "Notification Signalling End to Monitoring of Presentity".
- the subscription has been terminated or blocked by the presentity. See "Notification of End to a Subscription".

The application accesses the **endpoint** address to retrieve the specific notifications.

If there is any issue with subscriptions, the response body will contain the address(es) of the presentity (or presentities) to whose attributes the watcher did not successfully subscribe.

Authorization

Basic

HTTP Method

`PUT`

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Start Presence Notification operation accepts following parameters:

- **presentity**. String. Required. The URI of a presentity or group whose data is being accessed.
- **checkImmediate**. String. Required. Boolean value ("true" or "false"). Set to:
  - **True**: The application requires the status notification to start immediately and thereafter, as often as required.
  - **False**: The application requires the status notification to start at the end of the time period.
- **criteria**. String. Required. The status of the terminal. It can be one or all of the following:
  - **Reachable**
  - **UnReachable**
  - **Busy**
- **frequency**: JSON object. Required. This object specifies the frequency for checking the status (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**. It takes the following parameters:
  - **metric**. String. Required. The unit of time for the check specified as **Millisecond**, **Second**, **Minute**, **Hour**, **Day**, **Week**, **Month**, or **Year**.
  - **units**. Integer. Required. The number of times the check must be made within the specified metric.
- **reference**. A JSON object. Required. Use this object to provide the following information about the endpoint that is to receive the notification:
  - **correlator**. String. Required. The correlator used to identify the notification.
  - **endpoint**. String. Required. The URI which represents the endpoint address to which the notification should be delivered. This string should be a Bayeux protocol channel name that begins with **`/bayeux/`***`appInstanceID`* where *appInstanceID* is the client application's application instance account ID.

    For more information on application instances, see the discussion on creating and maintaining service provider and application accounts in *Accounts and SLAs Guide*.
  - **interfaceName**. String. Required. A descriptive string to identify the type of notification.
- **count**. Integer. Required. The maximum number of notifications to be sent to the watcher. If this number is reached the `statusEnd` notification is delivered to the `endpoint` address.
- **duration**. A JSON object. Required. This object specifies the total duration before for the count with the following:
  - **metric**. String. Required. The unit of time for the notifications specified as **Millisecond**, **Second**, **Minute**, **Hour**, **Day**, **Week**, **Month**, or **Year**.
  - **units**. Integer. Required. The duration in the specified metric.

The request body is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "checkImmediate": "Boolean",
```

```
  "frequency": {
    "metric": "Millisecond|Second|Minute|Hour|Day|Week|Month|Year",
    "units": "Integer"
  },
  "presentity": "URI",
  "reference": {
    "correlator": "String",
    "endpoint": "URI",
    "interfaceName": "String"
  },
  "attributes": ["Activity|Place|Privacy|Sphere|Communication|Other"],
  "count": "Integer",
  "duration": {
    "metric": "Millisecond|Second|Minute|Hour|Day|Week|Month|Year",
    "units": "Integer"
  }
}
```

Response Header

The Location header field contains the URI of the publish/subscribe server:

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

If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The requested information is returned in a JSON object as the value for the **result** attribute. It is an array of URIs which represent the addresses of the presentities to whose attributes the watcher did not successfully subscribe:

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

The actual notifications delivered to the **endpoint** address are described below.

Notification of Subscription to a Presentity's Data

The **notifySubscription** notification indicates whether the presentity has (or has not) authorized the watcher and the attributes that are permitted to be viewed.

The **notifySubscription** is a JSON object containing the following:

- **presentity**. String. The URI of the presentity who has (or has not) authorized the watcher to whom the notification is sent.
- **decisions**: Array of JSON objects. Required. Each JSON object in the array contains the following parameters:
  - **presenceAttribute**. String. Required. The specific Presence attribute for which permission is to be granted. See "PresenceAttribute".
  - **decision**. Boolean. Required. If the value is **true**, permission is granted or permission is denied.

The subscription notification is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{"notifySubscription": {
  "presentity": "URI",
  "decisions": [{
```

```
    "decision": "Boolean",
    "presenceAttribute": "Activity|Place|Privacy|Sphere|Communication|Other"
  }]
}}
```

Notification of a Status Change for a Presentity

The **statusChanged** JSON object is delivered when there is a change to a specified attribute.

This object is sent to the Bayeux channel name associated with the application instance and specified in the request body as the `endpoint` attribute value.

The **statusChanged** is a nested JSON object containing the following:

- **correlator**. String. The correlator used to identify the notification and provided in the request body of the Start Presence Notification operation.
- **presentity**. String. The URI of the presentity associated with this data.
- **changedAttributes**. JSON object. Its value is an array of nested JSON objects and attributes which represent the aggregated presence data of the presentity. It contains the following:
  - **lastChange**. String. The date and time when this attribute was last changed in ISO 8601 extended format.
  - **typeAndValue**. JSON object. The presence information of the presentity. See "TypeandValue".
  - **note**. String. This optional entry is an explanatory note.

The notification for a status change is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{"statusChanged": {
  "changedAttributes": [{
    "lastChange": "Calendar",
    "typeAndValue": {
      "unionElement": "Activity|Place|Privacy|Sphere|Communication|Other",
      "activity":
"ActivityNone|Available|Busy|DoNotDisturb|OnThePhone|Steering|Meeting|Away|Meal|Pe
rmanentAbsence|Holiday|Performance|InTransit|Travel|Sleeping|ActivityOther",
      "communication": {"means": [{
        "contact": "URI",
        "priority": "Float",
        "type": "Phone|Chat|Sms|Video|Web|Email|Mms|MeansOther"
      }]},
      "other": {
        "name": "String",
        "value": "String"
      },
      "place":
"PlaceNone|Home|Office|PublicTransport|Street|Outdoors|PublicPlace|Hotel|Theatre|R
estaurant|School|Industrial|Quiet|Noisy|Aircraft|Ship|Bus|Station|Mall|Airport|Tra
in|PlaceOther",
      "privacy":
"PrivacyNone|PrivacyPublic|PrivacyPrivate|PrivacyQuiet|PrivacyOther",
      "sphere": "SphereNone|SphereWork|SphereHome|SphereOther"
    },
    "note": "String"
  }],
```

```
  "correlator": "String",
  "presentity": "URI"
}}
```

Notification Signalling End to Monitoring of Presentity

The **statusEnd** JSON object is delivered when the notification count specified as the value for **count** is reached or when the duration specified for **units** in the **duration** object is reached.

It is sent to the Bayeux channel name associated with the application instance and specified in the request body as the **endpoint** attribute value.

The **statusEnd** is a JSON object containing the following:

- **correlator**. String. The correlator used to identify the notification and provided in the request body of the Start Presence Notification operation.

  This notification is not delivered in the case of an error, or if the application ended the notification using **endNotification**.

The notification signalling the end of the monitoring of a presentity is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

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

Notification of End to a Subscription

The **subscriptionEnded** JSON object is delivered when the subscription has been terminated, either blocked by the presentity or because of a timeout or connection failure.

It is sent to the Bayeux channel name associated with the application instance and specified in the request body as the **endpoint** attribute value.

The **subscriptionEnded** is a JSON object containing the following:

- **presentity**. String. The URI of the presentity.
- **reason**. String. Required. The string indicates whether there was a timeout or the presentity blocked the watcher.

The notification signalling the end of the subscription to a presentity is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{"subscriptionEnded": {
  "presentity": "URI",
  "reason": "String"
}}
```

End Presence Notification

The End Presence Notification operation stops a Presence notification.

To stop a previously set up Presence notification, provide the correlator for the notification passed earlier in the Start Presence Notification request.

There is no request or response body for the End Presence Notification operation. If the request fails, the body of the error response will contain the identifier for the notification and the type of exception.

Authorization

Basic

HTTP Method

`DELETE`

URI

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

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *correlator* is a string. It is the correlator for the notification provided in the **reference** object within the request body of the initial Start Presence Notification operation.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body.

Get User Presence

The Get User Presence operation retrieves 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.

To retrieve the information, include the URI of the presentity and the required attributes as a data object in the query string of the Request-URI of the GET method.

If the Get User Presence operation is successful, the response body contains the `result` JSON data object with the aggregated presence data of the specified presentity.

Authorization

Basic

HTTP Method

`GET`

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed. *${query}* is a data object with the following parameters:

- **presentity**. String. Required. The URI of the presentity to whose data the watcher has previously successfully subscribed.
- **attributes**: Array of string values. Optional. The set of Presence attributes the watcher wishes to see. An empty array (or if this attribute is not present) indicates the watcher wishes to view all the attributes. For a description of the attributes, see "attributes".

The following JSON data structure represents this data object in the URI. The value part of each name/value pair indicates its data type:

```
{
  "presentity": "URI",
  "attributes": ["Activity|Place|Privacy|Sphere|Communication|Other"]
}
```

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The requested information is returned in a JSON object as the value for the **result** attribute. It is an array of nested JSON objects and attributes which represent the aggregated presence data of a presentity. It contains the following:

- **lastChange**. String. The time and date the data last changed in ISO 8601 extended format.
- **typeAndValue**: JSON object. The presence information for a presentity (described earlier). See "TypeandValue".
- **note**. String. This optional entry is an explanatory note.

The response body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{"result": [{
  "lastChange": "Calendar",
  "typeAndValue": {
    "unionElement": "Activity|Place|Privacy|Sphere|Communication|Other",
    "activity":
"ActivityNone|Available|Busy|DoNotDisturb|OnThePhone|Steering|Meeting|Away|Meal|Pe
rmanentAbsence|Holiday|Performance|InTransit|Travel|Sleeping|ActivityOther",
    "communication": {"means": [{
      "contact": "URI",
      "priority": "Float",
      "type": "Phone|Chat|Sms|Video|Web|Email|Mms|MeansOther"
    }]},
    "other": {
      "name": "String",
      "value": "String"
    },
    "place":
"PlaceNone|Home|Office|PublicTransport|Street|Outdoors|PublicPlace|Hotel|Theatre|R
estaurant|School|Industrial|Quiet|Noisy|Aircraft|Ship|Bus|Station|Mall|Airport|Tra
in|PlaceOther",
    "privacy":
"PrivacyNone|PrivacyPublic|PrivacyPrivate|PrivacyQuiet|PrivacyOther",
    "sphere": "SphereNone|SphereWork|SphereHome|SphereOther"
  },
  "note": "String"
}]}
```

Publish

The Publish operation allows the presentity to publish presence information.

To publish the information, the request body contains the `presence` JSON data object with the aggregated presence data of the specified presentity.

If the request fails, the body of the error response will contain the identifier for the notification and the type of exception.

Authorization

Basic

HTTP Method

`PUT`

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Publish operation is a nested data object. The value for **presence** attribute is an array of nested JSON structures which represent the aggregated presence data of a presentity. It accepts the following parameters:

- **lastChange**. String. The time and date the data last changed in ISO 8601 extended format.
- **typeAndValue**. JSON object. The presence information for a presentity (described earlier). See "TypeandValue".
- **note**. String. This optional entry is an explanatory note.

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{"presence": [{
  "lastChange": "Calendar",
  "typeAndValue": {
    "unionElement": "Activity|Place|Privacy|Sphere|Communication|Other",
    "activity":
"ActivityNone|Available|Busy|DoNotDisturb|OnThePhone|Steering|Meeting|Away|Meal|Pe
rmanentAbsence|Holiday|Performance|InTransit|Travel|Sleeping|ActivityOther",
    "communication": {"means": [{
      "contact": "URI",
      "priority": "Float",
      "type": "Phone|Chat|Sms|Video|Web|Email|Mms|MeansOther"
    }]},
    "other": {
      "name": "String",
```

```
      "value": "String"
    },
    "place":
"PlaceNone|Home|Office|PublicTransport|Street|Outdoors|PublicPlace|Hotel|Theatre|R
estaurant|School|Industrial|Quiet|Noisy|Aircraft|Ship|Bus|Station|Mall|Airport|Tra
in|PlaceOther",
    "privacy":
"PrivacyNone|PrivacyPublic|PrivacyPrivate|PrivacyQuiet|PrivacyOther",
    "sphere": "SphereNone|SphereWork|SphereHome|SphereOther"
  },
  "note": "String"
}]}
```

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body.

16

Device Capabilities

This chapter describes the operations in the Device Capabilities interface of the RESTful Web Services provided in Oracle Communications Services Gatekeeper.

About the Device Capabilities Interface

Applications use the RESTful Device Capabilities interface to request and receive the a terminal's device ID (such as the IMEI) using **getDeviceID** or receive the devices device ID type, name of the device/model, and a link to the User Agent Profile XML file using **getCapabilities**.

REST Service Descriptions Available at Run-time

When the Administration Server for your Services Gatekeeper domain is in the running state, the REST service descriptions of these operations can be found at

`http://host:port/rest/device_capabilities/index.html`

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Get Capabilities

The Get Capabilities operation retrieves the unique ID for the device type, name of the device/model, and a link to the User Agent Profile XML file.

The request includes the device ID of the device, usually a phone number.

Authorization

Basic

HTTP Method

`GET`

URI

```
http://host:port/rest/device_capabilities/device_
capabilities?capabilities=${capabilities}
```

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *${capabilities}* is a data object which contains the URI (usually, phone number) as the value of the **address** attribute.

  The following JSON data structure represents this data object in the URI. The value part of each name/value pair indicates its data type:

  ```
  {"address": "URI"}
  ```

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The requested device information is returned in a JSON object as the value for the **result** attribute. It contains the following name-value pairs.

- **deviceID**. String. The device/model number for the URI provided in the request.
- **name**. String. The name of the device.
- **userAgentProfileReference**. String. The link to the User Agent Profile XML file.

The response body for this operation is represented by the following JSON data structure, where the value part of the name/value pair indicates its data type:

```
{"result": {
  "deviceId": "String",
  "name": "String",
  "userAgentProfileReference": "URI"
}}
```

Examples

Example 16–1 Get Capabilities Request

```
GET rest/device_capabilities/device_
capabilities?capabilities=%7B%22address%22%3A%22tel%3A%221234%22%7D HTTP/1.1
X-Session-ID: app:5198750923966743997
Authorization: Basic YXBwX2luc3RhbmNlXzE6d2VibG9naWM=
User-Agent: Jakarta Commons-HttpClient/3.0
Host: 10.182.100.245:9001
```

Example 16–2 Get Capabilities Response

```
HTTP/1.1 200 OK
Date: Fri, Nov 05 2010 05:23:27 GMT
Content-Length=124
Content-Type=application/json
X-Powered-By: Servlet/2.5 JSP/2.1
Host: 10.182.100.245:9001

{"result":
      {
        "name":"alice.smith.uiii",
        "deviceId":"0998",
        "userAgentProfileReference":"#1234897897987867gihuyuijlkjlkjllk"
       }
}
```

Example 16–3 Error Response

```
HTTP/1.1 500 Internal Server Error
Date: Fri, Nov 05 2010 05:32:27 GMT
Content-Length=131
Content-Type=application/json
X-Powered-By: Servlet/2.5 JSP/2.1

{"error":
      {
        "message":"Invalid input for message part Address",
        "type":"org.csapi.schema.parlayx.common.v3_1.ServiceException",
       }
}
```

Get Device Id

The Get Device Id operation retrieves the equipment identifier device/name (for example the IMEI number) for a given device. The request includes the device ID, usually a phone number.

The request includes the device ID of the device.

Authorization

Basic

HTTP Method

`GET`

URI

`http://host:port/rest/device_capabilities/device_capabilities?deviceId=${deviceId}`

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *${deviceId}* is a data object which contains the URI (usually, phone number) as the value of the **address** attribute.

  The following JSON data structure represents this data object in the URI. The value part of each name/value pair indicates its data type:

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

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The required equipment identifier device/name is returned as the value for the **result** attribute. The structure for the response body is

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

Examples

Example 16–4 Get Device Id Request

```
GET rest/device_capabilities/device_
capabilities?deviceId=%7B%22address%22%3A%22tel%3A%221234%22%7D HTTP/1.1
X-Session-ID: app:5198750923966743997
Authorization: Basic YXBwX2luc3RhbmNlXzE6d2VibG9naWM=
User-Agent: Jakarta Commons-HttpClient/3.0
Host: 10.182.100.245:9001
```

Example 16–5 Get Device Id Response

```
HTTP/1.1 200 OK
Date: Fri, Nov 05 2010 05:34:51 GMT
Content-Length=17
Content-Type=application/json
X-Powered-By: Servlet/2.5 JSP/2.1
Host: 10.182.100.245:9001

{"result":"0998"}
```

Example 16–6 Error Response

```
HTTP/1.1 500 Internal Server Error
Date: Fri, Nov 05 2010 05:37:08 GMT
Content-Length=131
Content-Type=application/json
X-Powered-By: Servlet/2.5 JSP/2.1

{"error":
      {
        "message":"Invalid input for message part Address",
        "type":"org.csapi.schema.parlayx.common.v3_1.ServiceException"
       }
}
```

17

Binary Short Messaging

This chapter describes the operations in the Binary Short Messaging interface of the RESTful Web Services provided in Oracle Communications Services Gatekeeper.

About the Binary Short Messaging Interface

Applications use the RESTful Binary Short Messaging interface allows an application to send any generic binary object attachments to the network using SMS. The supported binary content is broader than the logos and ringtones specified by the Parlay X SMS Web service, extending to other types of binary content such as vCards (a file format standard for electronic business cards).

These interfaces also provide operations to start and stop notifications for SMSes with binary content.

Send Binary Sms

The Send Binary Sms operation sends an SMS that includes content in binary format.

Authorization

Basic

HTTP Method

POST

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Send Binary Sms operation accepts the following parameters:

- **addresses**. String. Required. The set of end-user terminal addresses of the recipients as an array of SIP-formatted URIs.
- **binaryMessage**. An array of JSON objects. Required. The message to be sent as an array of User Data Header (UDH) elements and message elements. Note that the entire array must be less than of equal to 141 bytes.

  Each element in the array contains:

  - **message**. String in base64Binary. Binary Message data formatted as TP-User Data (TP-UD) excluding the TP-User-Data-Indicator (TP UDHI).
  - **udh**. String in base64Binary. Specifies if the TP-User Data (TP-UD) field contains only the short message, or if it also contains the header formatted as the TP-User-Data-Indicator (TP UDHI).
- **dcs**: Byte. Required. The data-encoding scheme for the binaryMessage parameter.
- **protocolId**. Byte. Required. TP-Protocol-Identifier according to 3GPP 23.040 6.5.0
- **charging**. A JSON object. Optional. This object defines the cost charging properties for the operation. The entry `"charging": null` indicates no charge. If a charge is to be applied, provide values for the following in the **charging** object:
  - **description**. String. Required if the **charging** object is present in the body of the request. The text to be used for information and billing.
  - **amount**. Number (integer, or decimal). Optional. The amount to be charged.
  - **code**. String. Optional. The charging code, from an existing contractual description.
  - **currency**. String. Optional. The currency identifier as defined in ISO 4217 [9].

- **receiptRequest**. a JSON object. Optional. Used to notify the application that the message has been delivered to the terminal, or that delivery is impossible.

  If a delivery receipt is required, provide values for each of the following parameters which define this object:

  - **correlator**. String. Used to correlate the receipt with the initial message.
  - **endpoint**. String. The endpoint address (URI) to which the receipt must be delivered.
  - **interfaceName**. String. A description provided to identify the type of receipt.
- **senderName**. String. Optional. The sender's name.
- **validityPeriod**. String. The validity period of the short message, formatted as a validity-period parameter as described in SMPP v3.4.

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "addresses":["URI"],
  "binaryMessage":[{
    "message":"base64Binary"
    "udh":"base64Binary"
  }],
  "dcs":"Byte",
  "charging": {
    "description": "String",
    "amount": "BigDecimal",
    "code": "String",
    "currency": "String"
  }
  "protocolId":"Byte",
  "receiptRequest": {
    "correlator": "String",
    "endpoint": "URI",
    "interfaceName": "String"
  },
  "senderName":"String",
  "validityPeriod":"String",
}
```

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The response body is a JSON object which contains the **result** attribute. The identifier for the delivery request is the string value for **result**.

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

Start Binary Sms Notification

The Start Binary SMS Notification operation starts a notification for short messages that contain binary content.

Authorization

Basic

HTTP Method

`PUT`

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

The request body for the Start Binary SMS Notification operation accepts the following parameters:

- **smsServiceActivationNumber**. String. Required. The destination address, as URI, of the short message.
- **reference**. JSON object. Required. Use this object to provide the following information about the endpoint that is to receive the notification:
  - **correlator**. String. Required. The correlator used to identify the notification.
  - **endpoint**. String. Required. The URI which represents the endpoint address to which the notification should be delivered. This string should be a Bayeux protocol channel name that begins with **`/bayeux/`***`appInstanceID`* where *appInstanceID* is the client application's application instance account ID.

    For more information on application instances, see the discussion on creating and maintaining service provider and application accounts in *Oracle Communications Services Gatekeeper Accounts and SLAs Guide*.
  - **interfaceName**. String. Required. A descriptive string to identify the type of notification.

The request body for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "reference": {
    "correlator": "String",
    "endpoint": "URI",
    "interfaceName": "String"
  },
  "smsServiceActivationNumber":"String",
}
```

Response Header

The Location header field contains the URI of the publish/subscribe server:

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

If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body. The appropriate notifications (described below) are sent to the **endpoint** address provided by the application in the request body of this operation.

Notification of Binary SMS Reception

When a Binary SMS has been received for an application, its designated **endpoint** address on the pub-sub server receives a nested JSON object.

This nested JSON object contains the following as the value for the attribute **notifyBinarySmsReception**:

- **correlator**. String. The correlator used in the request body for this operation.
- **message**. a nested JSON object. It contains the data-encoding scheme for the set of messages in the notification, and the messages as an array of User Data Header (UDH) elements and message elements.
  - **dcs**. Byte. The data-encoding scheme for the binary messages included in this object.
  - **message**. Array of JSON objects. Table 17–1 lists the contents of each message object:

Table 17–1 Attributes of Message Object

Attribute	Description
message	Message data as a string base64Binary format
udh	Information specifying the message data format

- **senderAddress**. String. The sender's address, as a URI.
- **smsServiceActivationNumber**. String. The destination address for the binary message, as a URI.
- **dateTime**. String. The date and time the message was received in ISO 8601 extended format `yyyy-mm-ddThh-mm-ss`.
- **protocolId**. Byte. The TP-Protocol-Identifier according to 3GPP 23.040 6.5.0.

The notification data object delivered to the **endpoint** address is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{"notifyBinarySmsReception": {
  "correlator":"String",
  "message":{
    "dcs":"Byte",
```

```
    "message":[{
        "message":"base64Binary"
        "udh":"base64Binary"
      }],
  "senderAddress":"URI",
  "smsServiceActivationNumber": "URI",
  "dateTime":"Calendar"
  "protocolId":"Byte"
  }
}}
```

Stop Binary Sms Notification

The Stop Binary Sms Notification operation terminates a previously-started notification for messages that contain binary content.

To stop a previously set up binary Sms notification, provide the correlator for the notification passed earlier in the Start Binary Sms Notification request.

There is no request or response body for Stop Binary Sms Notification. If the request fails, the body of the error response will contain the identifier for the notification and the type of exception.

Authorization

Basic

HTTP Method

`DELETE`

URI

The Request-URI used in the DELETE method for Stop Sms Notification is:

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

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *correlator* is the correlator for the notification provided in the **reference** object of the initial Start Binary Sms Notification request.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

There is no response body.

18

Session Manager

This chapter describes the operations in the Session Manager interface of the RESTful Web Services provided in Oracle Communications Services Gatekeeper.

About the Session Manager Service

Applications use the RESTful Session Manager interface to access the Session Manager Web Service to get a unique session ID. Each application then add this session ID to the header of all its requests. Services Gatekeeper uses this value to keep track of all the traffic that an application sends for the duration of the session, and to destroy a session.

The `GeneralException` error will be thrown when any operation in the RESTful Session Manager interface fails.

REST Service Descriptions Available at Run-time

When the Administration Server for your Services Gatekeeper domain is in the running state, the REST service descriptions of these operations can be found at

`http://`*`host`*`:`*`port`*`/rest/session_manager/index.html`

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Get Session

The Get Session operation creates a session with an ID.

If the Get Session operation is successful, the response body will contain the session ID. This string value is used by the application in the X-Session-ID header of all subsequent traffic requests.

Authorization

Basic

HTTP Method

`POST`

URI

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

where *host* and *port* are the hostname and port of the machine on which Services Gatekeeper is installed.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The response body is a JSON object containing the **getSession Return** attribute whose value is the session Id, returned as a string.

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

Get Session Remaining Lifetime

The Get Session Remaining Lifetime operation retrieves the time remaining in this session, in milliseconds.

The Request-URI for the GET method contains the session ID.

If the Get Session Remaining Lifetime operation is successful, the response body will contain the time remaining in this session, in milliseconds.

Authorization

Basic

HTTP Method

`GET`

URI

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

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *${sessionID}* is the session ID obtained from the response to the Get Session request.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

There is no request body.

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The response body is a JSON object containing the **getSessionRemainingLifeTimeReturn** attribute whose value is an integer representing the time remaining in this session, in milliseconds.

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

Destroy Session

The Destroy Session operation destroys this session.

To do so, provide the appropriate session Id in the Request-URI for this operation. This identifier should have been obtained by the initial setup for this session.

There is no request or response body for Destroy Session. If the request fails, the body of the error response will contain the call identifier and the type of exception.

Authorization

Basic

HTTP Method

`DELETE`

URI

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

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *sessionId* is the session ID obtained from the response to the Get Session request.

Request Header

The MIME-type for the Content-Type header field is **application/json**.

Request Body

There is no request body.

Response Body

The response body is a JSON object containing the **destroySessionReturn** attribute whose value is a boolean.

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

If the value for **destroySessionReturn** is **true**, the session was destroyed.

19

Subscriber Profile

This chapter describes the operations in the Subscriber Profile interface of the RESTful Web Services provided in Oracle Communications Services Gatekeeper.

About the Subscriber Profile Interface

Applications use the RESTful Subscriber Profile interface to query an operator's database for individual subscriber profile attributes (such as a user's terminal type) or entire subscriber profiles.

REST Service Descriptions Available at Run-time

When the Administration Server for your Services Gatekeeper domain is in the running state, the REST service descriptions of these operations can be found at

`http://host:port/rest/subscriber_profile/index.html`

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Get

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

To retrieve specific subscriber profile properties, provide the address the subscriber and the required subscriber profile properties within the query object in the Request-URI of the GET method.

If this operation is successful, the response body contains a JSON data object with the required pathnames and values for the required profile properties associated with the specified subscriber.

Authorization

Basic

HTTP Method

`GET`

URI

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

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *${query}* is a JSON object.

The parameters accepted in the *${query}* object are:

- **address**. String. Required. The address associated with the subscriber whose data is being accessed. The supported schemes are:
  - **tel id**
  - **imsi** (International Mobile Subscriber Identity)
  - **IPv4**
- **pathNames**: Array of String values. Required. The requested subscriber profile properties expressed as a relative UNIX path. For example, **serviceName/accessControlId/accessControlId**

The *${query}* object in the URI is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
   "address":"URI",
   "pathNames":["String"]
}
```

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The response body is a JSON object which contains the requested properties as an array value for the **properties** attribute. Each element in the array contains the following:

- **pathName**. String. The pathname for the requested property.
- **propertyValue**. String. The value associated with the requested property.

The response body for this operation is represented by the following JSON data structure, where the value part of the name/value pair indicates its data type:

```
{"properties": [{
  "pathName": "String",
  "propertyValue": "String"
}]}
```

Get Profile

The GET Profile operation retrieves the entire profile for a specific subscriber.

To retrieve the entire profile for a specific subscriber, provide the associated subscriber Id and profile ID within the query object in the Request-URI of the GET method.

If Get Profile is successful, the response body contains a JSON data object with the required pathnames and values for all the profile properties associated with the specified subscriber.

Authorization

Basic

HTTP Method

`GET`

URI

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

where:

- *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.
- *${id}* is a JSON object.

The parameters accepted by *${id}* object are:

- **profileID**. String. Required. 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. This entry may be ignored if Services Gatekeeper connects to the network using certain protocols.
- **subscriberID**. String. Required. The ID that uniquely identifies the subscriber whose profile is being accessed.

The *${id}* object in the URI is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
   "profileID":"a_profileId",}
   "subscriberID":"a_subsc_id"
}
```

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The response body is a JSON object which contains the requested properties as an array value for the **result** attribute. Each element in the array contains the following:

- **pathName**. String. The pathname for the requested property.

- **propertyValue**. String. The value associated with the requested property.

Here is the structure:

```
{"result": [{
  "pathName": "String",
  "propertyValue": "String"
}]}
```

20

WAP Push

This chapter describes the operations in the WAP Push interface of the RESTful Web Services provided in Oracle Communications Services Gatekeeper.

About the WAP Push Interface

Applications use the RESTful WAP Push interface to send a Wireless Application Protocol (WAP) Push message. The content of the message is coded as a (Password Authentication Protocol) 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.

For links to the specifications, see

`http://www.openmobilealliance.org/tech/affiliates/wap/wapindex.html`.

The actual message is sent as an HTTP attachment. See "Headers for Multipart Messages with Attachments" for more information.

REST Service Descriptions Available at Run-time

When the Administration Server for your Services Gatekeeper domain is in the running state, the REST service descriptions of these operations can be found at

`http://`*`host`*`:`*`port`*`/rest/push_message/index.html`

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Send Push Message

The Send Push Message operation sends a WAP Push message.

Authorization

Basic

HTTP Method

`POST`

URI

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

where *host* and *port* are the host name and port of the machine on which Services Gatekeeper is installed.

Request Header

The request headers depend on the type of message:

- If the message content is in the form of an attachment, the request will be multipart and therefore, the request header must contain the header fields that describe the parts of the message.
- If the message does not contain an attachment, the special headers associated with the multipart messaged are not used.

Message Part Content

The Message Part Content for this operation accepts the following parameters:

- **destinationAddresses**. String. Required. An array of terminal addresses.

  Each of the addresses in this array should be formatted according to the Push Proxy Gateway Service Specification (WAP-249-PPGService-20010713-a).
- **pushId**. String. Required. A unique identifier provided by the application. Supported types are PLMN and USER.
- **replaceMethod**. String. Required. Valid entries are:
  - **all**
  - **pendingOnly**

  replaceMethod is used in conjunction with the **pushId** parameter and defines how to replace a previously sent message. This entry will be ignored if **replacePushId** is NULL.
- **requesterId**. String. Required. The application ID as given by the operator.
- **serviceCode**. String. Required. A code for charging purposes.
- **additionalProperties**. An array of JSON objects. Optional.

  Each element in the array is a JSON object used to add additional properties with the following parameters:

  - **name**. String. Required if the `additionalProperties` object is present. Valid entries include **pap.priority**, **pap.delivery-method**, **pap.network-required**, **pap.bearer**, and **pap.bearer-required**.
  - `value`. String. Required if the **additionalProperties** object is present. The value associated with the property.

- **deliverAfterTimeStamp**. String. Optional. The date and time after which the content should be delivered to the wireless device. This entry is in ISO 8601 extended format, as `yyyy-mm-ddThh-mm-ss`.

  If the network does not support this parameter, this entry will be rejected.

- **deliverBeforeTimeStamp**. String. Optional. The date and time by when the content should be delivered to the wireless device. This entry is in ISO 8601 extended format, as `yyyy-mm-ddThh-mm-ss`.

  If the network does not support this parameter, this entry will be rejected.

- **progressNoteRequested**. Boolean. Optional. If **true**, the application wishes to receive progress notes at the address specified by the **resultNotificationEndpoint**.

- **replacePushId**. String. Optional. The Push Id of the message that is to be replaced. If it is set to NULL, the message is treated as a new message. If it is set to a valid ID, message replacement will occur for all currently pending messages. Messages that have already been delivered cannot be cancelled, and therefore cannot.

  Messages that have already been delivered cannot be cancelled, and therefore cannot be replaced

- **resultNotificationEndpoint**. String. Optional. The URI which represents the endpoint address to which the notification should be delivered. This string should be a Bayeux protocol channel name that begins with **`/bayeux/`***`appInstanceID`* where *appInstanceID* is the client application's application instance account ID.

  If the application does not wish to receive notifications, this value should be NULL.

- **sourceReference**. String. Optional. The name of the service provider.

The Message Part Content for this operation is represented by the following JSON data structure, where the value part of each name/value pair indicates its data type:

```
{
  "destinationAddresses": ["String"],
  "pushId": "String",
  "replaceMethod": "all|pending-only",
  "requesterID": "String",
  "serviceCode": "String",
  "additionalProperties": [{
    "name": "String",
    "value": "String"
  }],
  "deliverAfterTimestamp": "Calendar",
  "deliverBeforeTimestamp": "Calendar",
  "progressNotesRequested": "Boolean",
  "replacePushId": "String",
  "resultNotificationEndpoint": "URI",
  "sourceReference": "String"
}
```

Response Header

Standard header fields. If the request fails, the Status-Line header field will contain the status code and the reason for the failure. See "Errors and Exceptions".

Response Body

The response body is a nested JSON object. It contains the **result** attribute with the following attributes and data objects:

- **pushId**. String. Required. A unique identifier provided by the application. Supported types are PLMN and USER.
- **result**: JSON object. This object contains the outcome code. It contains the following two entries:
  - **code**. String. The outcome code generated by the network node. Table 20–1 lists the possible values.
  - **description**. String. The textual description for the code.
- **additionalProperties**. An array of JSON objects used to add additional properties.

  Each element in the array is a JSON object with the following parameters:
  - **name**. String. The name of an additional property. One of: **pap.stage**, **pap.note**, **pap.time**.
  - **value**. String. The value associated with the property.
- **replyTime**. String. The date and time for the reply in ISO 8601 extended format, as `yyyy-mm-ddThh-mm-ss`.
- **senderAddress**. String. Optional. The sender's address.
- **senderName**. String. Optional. The sender's name.

Table 20–1 Possible 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
2008	Cancellation not possible
3000	Internal server error
3001	Not implemented
3002	Version not supported
3003	Not possible

Table 20–1 (Cont.) Possible Outcome Codes

Code	Description
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

The response body for this operation is represented by the following JSON data structure, where the value part of the name/value pair indicates its data type:

```
{"result": {
  "pushId": "String",
  "result": {
    "code": "String",
    "description": "String"
  },
  "additionalProperties": [{
    "name": "String",
    "value": "String"
  }],
  "replyTime": "Calendar",
  "senderAddress": "String",
  "senderName": "String"
}}
```

Result Notification Message Object

The **resultNotificationMessage** object delivered to the **resultNotificationEndpoint** address is a nested JSON object containing the following entries:

- **address**. String. The address of the terminal to which this message is sent.
- **code**. String. The final state of the message, one of the values listed in Table 20–1.
- **messageState**. String. The state of the message. One of:
  - **rejected**
  - **pending**
  - **delivered**
  - **undelivered**
  - **expired**
  - **aborted**
  - **timeout**
  - **cancelled**

  - **unknown**

- **pushId**. String. The unique identifier defined in the initial request, used for correlation.
- **additionalProperties**. An array of JSON objects used to add additional properties.

  Each element in the array is a JSON object with the following parameters:

  - **name**. String. The name of an additional property, dependent on the network node. One of: **pap.priority**, **pap.delivery-method**, **pap.network-required**, **pap.bearer**, and **pap.bearer-required**.
  - **value**. String. The value associated with the property.
- **description**. String. A description of the notification provided by the network. (May not be provided).
- **eventTime**. String. The date and time the message reached the destination in ISO 8601 extended format, as `yyyy-mm-ddThh-mm-ss`
- **receivedTime**. String. The date and time the message was received at the network node in ISO 8601 extended format, as `yyyy-mm-ddThh-mm-ss`
- **senderAddress**. String. Optional. The sender's address.
- **senderName**. String. Optional. The sender's name.

The notification of the result for this operation is represented by the following JSON data structure, where the value part of the name/value pair indicates its data type:

```
{"resultNotificationMessage": {
  "address": "String",
  "code": "String",
  "messageState":
"rejected|pending|delivered|undeliverable|expired|aborted|timeout|cancelled|unknow
n",
  "pushId": "String",
  "additionalProperties": [{
    "name": "String",
    "value": "String"
  }],
  "description": "String",
  "eventTime": "Calendar",
  "receivedTime": "Calendar",
  "senderAddress": "String",
  "senderName": "String"
}}
```