Oracle® Communications Services Gatekeeper

Application Development Guide

Release 4.1

January 2009

ORACLE

Oracle Communication Services Gatekeeper Application Development Guide, Release 4.1

Contents

Session Manager Web Service

Extended Web Services Binary SMS

Extended Web Services WAP Push

Extended Web Services Subscriber Profile

Extended Web Services Common

Parlay X 2.1 Interfaces

Parlay X 3.0 Interfaces

Native Interfaces

Access Web Service (deprecated)

CHAPTER 1

Document Roadmap

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

- Document Scope and Audience
- Guide to This Document
- Terminology
- Related Documentation

Document Scope and Audience

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

Note: This document covers the SOAP-based interface and the native interface for Oracle Communications Services Gatekeeper. A separate document in this set, *RESTful Application Development Guide*, describes how to use the RESTful interface.

Guide to This Document

The document contains the following chapters:

Chapter 1, "Document Roadmap": This chapter

Chapter 2, “Creating Applications for Oracle Communications Services Gatekeeper”: A general introduction to the concepts involved in using Oracle Communications Services Gatekeeper

Chapter 3, “Interacting with Oracle Communications Services Gatekeeper”: SOAP message requirements in Oracle Communications Services Gatekeeper

Chapter 4, “Session Manager Web Service”: A detailed description of the Session Manager Web Service

Chapter 7, “Extended Web Services Subscriber Profile”: A detailed description of the available operations used to get subscriber profile data

Chapter 5, “Extended Web Services Binary SMS”: A detailed description of the available operations used to send SMSes with binary content

Chapter 8, “Extended Web Services Common”: A detailed description of the datatypes shared among the Extended Web Services interfaces

Chapter 9, “Parlay X 2.1 Interfaces”: A description of the Parlay X 2.1 interfaces available with details on how they are implemented in Oracle Communications Services Gatekeeper.

Chapter 10, “Parlay X 3.0 Interfaces”: A description of the Parlay X 3.0 interfaces available with details on how they are implemented in Oracle Communications Services Gatekeeper.

Chapter 11, “Native Interfaces.”: A description of the MM7 and SMPP interfaces available with details on how they are implemented in Oracle Communications Services Gatekeeper

Chapter 12, “Access Web Service (deprecated)”: A description of the Access Web Service

Terminology

The following terms and acronyms are used in this document:

- Account—A registered application or service provider. An account belongs to an account group, which is tied to a common SLA
- Account group—Multiple registered service providers or services which share a common SLA
- Administrative User—Someone who has privileges on the Oracle Communications Services Gatekeeper management tool. This person has an administrative user name and password
- Alarm—The result of an unexpected event in the system, often requiring corrective action
- API—Application Programming Interface

- Application—A TCP/IP based, telecom-enabled program accessed from either a telephony terminal or a computer
- Application-facing interface—The Application Services Provider facing interface
- Application Service Provider—An organization offering application services to end users through a telephony network
- AS—Application Server
- Application Instance—An Application Service Provider from the perspective of internal Oracle Communications Services Gatekeeper administration. An Application Instance has a user name and password or certificate
- CBC—Content Based Charging
- End User—The ultimate consumer of the services that an application provides. An end user can be the same as the network subscriber, as in the case of a prepaid service or they can be a non-subscriber, as in the case of an automated mail-ordering application where the subscriber is the mail-order company and the end user is a customer to this company
- Enterprise Operator —See Service Provider
- Event—A trackable, expected occurrence in the system, of interest to the operator
- HA —High Availability
- HTML—Hypertext Markup Language
- IP—Internet Protocol
- JDBC—Java Database Connectivity, the Java API for database access
- Location Uncertainty Shape—A geometric shape surrounding a base point specified in terms of latitude and longitude. It is used in terminal location
- MAP—Mobile Application Part
- Mated Pair—Two physically distributed installations of Oracle Communications Services Gatekeeper nodes sharing a subset of data allowing for high availability between the nodes
- MM7—A multimedia messaging protocol specified by 3GPP
- MPP—Mobile Positioning Protocol
- Network Plug-in—The Oracle Communications Services Gatekeeper module that implements the interface to a network node or OSA/Parlay SCS through a specific protocol

- NS—Network Simulator
- OAM —Operation, Administration, and Maintenance
- Operator—The party that manages the Oracle Communications Services Gatekeeper. Usually the network operator
- OSA—Open Service Access
- PAP—Push Access Protocol
- Plug-in—See Network Plug-in
- Plug-in Manager—The Oracle Communications Services Gatekeeper module charged with routing an application-initiated request to the appropriate network plug-in
- Quotas—Access rule based on an aggregated number of invocations. See also Rates
- Rates—Access rule based on allowable invocations per time period. See also Quotas
- Rules—An optional set of customizable criteria in addition to those located in SLAs according to which requests are evaluated
- SCF—Service Capability Function or Service Control Function, in the OSA/Parlay sense.
- SCS—Service Capability Server, in the OSA/Parlay sense. Oracle Communications Services Gatekeeper can interact with these on its network-facing interface
- Service Capability—Support for a specific kind of traffic within Oracle Communications Services Gatekeeper. Defined in terms of communication service
- Service Provider—See Application Service Provider
- SIP—Session Initiation Protocol
- SLA—Service Level Agreement
- SMPP—Short Message Peer-to-Peer Protocol
- SMS—Short Message Service
- SMSC—Short Message Service Centre
- SNMP—Simple Network Management Protocol
- SOAP—Simple Object Access Protocol
- SPA—Service Provider APIs

- SS7—Signalling System 7
- Subscriber—A person or organization that signs up for access to an application. The subscriber is charged for the application service usage. See End User
- SQL—Structured Query Language
- TCP—Transmission Control Protocol
- Communication Services—The data flow of a particular request through Oracle Communications Services Gatekeeper. Different Service Capabilities use different communication services
- USSD—Unstructured Supplementary Service Data
- VAS—Value Added Service
- VLAN—Virtual Local Area Network
- VPN—Virtual Private Network
- Oracle Communications Services Gatekeeper Core—The container that holds the Core Utilities
- Oracle Communications Services Gatekeeper Core Utilities—A set of shared utilities
- WSDL —Web Services Definition Language
- XML—Extended Markup Language

Related Documentation

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

- *System Administrator's Guide*
- *Integration Guidelines for Partner Relationship Management*
- *SDK User Guide*
- *Managing Accounts and SLAs*
- *Statement of Compliance and Protocol Mapping*
- *Concepts and Architectural Overview*

- *Communications Services Reference*
- *Handling Alarms*
- *Licensing*
- *Installation Guide*
- *Platform Development Studio - Developer's Guide*
- *RESTful Application Development Guide*

CHAPTER 2

Creating Applications for Oracle Communications Services Gatekeeper

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

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

- Basic Concepts
- Functional Overview
- Application Testing Workflow

Basic Concepts

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

- Communication Services
- Traffic Types

Communication Services

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

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

Traffic Types

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

Application-initiated Traffic

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

Network-triggered Traffic

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

- The application must inform Oracle Communications Services Gatekeeper that it wishes to receive information that originates from the network. It does this by *subscribing* or *registering* for *notifications* via an application-initiated request. (In certain cases, this can also be accomplished by the operator, using OAM procedures.) Often this subscription includes filtering criteria that describes exactly what kinds of traffic it wishes to receive. Depending on the underlying network configuration, Oracle Communications Services Gatekeeper itself, or the operator using manual steps, informs the underlying network about the kind of data that is requested. These notifications may be status updates, as described above, or, in some instances, may even include short or multimedia messages from a terminal on the telecom network.
- The application must arrange to receive the network-triggered information, either by implementing a Web Service endpoint on its own site to which Oracle Communications Services Gatekeeper dispatches the notifications, or by polling Oracle Communications Services Gatekeeper to retrieve them. Notifications are kept in Oracle Communications Services Gatekeeper for retrieval for only limited amounts of time.

Management Structures

In order to help telecom operators organize their relationships with application providers, Oracle Communications Services Gatekeeper uses a hierarchical system of accounts. Each application is assigned a unique application instance ID which is tied to an Application Account. All the Application Accounts that belong to a single entity are assigned to a Service Provider Account. Application Accounts with similar requirements are put into Application Groups and Service Providers with similar requirements are put into Service Provider Groups. Each Application Group is associated with an Application Group Service Level Agreement (SLA) and each Service Provider Group are associated with Service Provider Group SLAs. These SLAs define and

regulate the contractual agreements between the telecom operator and the application service provider, and cover such things as which services the application may access and the maximum bandwidth available for use.

Functional Overview

Oracle Communications Services Gatekeeper allows operators to provide client application developers with a choice of interface types, based on the needs of their applications. Oracle Communications Services Gatekeeper provides SOAP-based Web Services interfaces, RESTful interfaces (see *RESTful Application Development Guide)*, and even two native telephony interfaces (MM7 and SMPP).

Note: The exact mix of interfaces depends on the specific Oracle Communications Services Gatekeeper installation.

The SOAP-based Web Services APIs are based on the Parlay X 2.1 and 3.0 standards and also include three additional Extended Web Services ones to cover Binary SMS, Subscriber Profile, and WAP Push functionality, which are not supported by Parlay X. The functionality supported by these communication services includes:

- Third Party Call (Parlay X 2.1 and 3.0)

  Using this communication service, an application can set up a call between two parties (the caller and the callee), poll for the status of the call, and end the call. In addition, using the 3.0 communication only, an application can set up a call among multiple participants and add, delete, or transfer those participants. The application can also use the Audio Call communication service to play audio messages to one or multiple of the call participants set up using Third Party Call and, using notifications set up with Call Notification PX 3.0, can also collect digits in response to playing the audio message.

- Audio Call (Parlay X 3.0)

  Using this communication service, an application can play audio to one or more call participants in an existing call session set up by PX 3.0 Third Party call, find out if the audio is currently being played, and explicitly end playing the audio. It can also collect digits from a participant in response to an audio message, and in conjunction with a notification set up using Call Notification PX 3. 0, can return that information to the application. It can also interrupt an ongoing interaction such as on-hold music.

- Call Notification (Parlay X 2.1 and 3.0)

  Using this communication service, an application can set up and end notifications on call events, such as a callee in a third party call attempt is busy. In addition, in some cases the

application can then reroute the call to another party. In addition, using the PX 3.0 communication service, an application can interact with PX 3.0 Audio Call to return digits collected from a call participant back to the application and to end calls.

- Short Messaging (Parlay X 2.1)

  Using this communication service, an application can send SMS text messages, ringtones, or logos to one or multiple addresses, set up and receive notifications for final delivery receipts of those sent items, and arrange to receive SMSes meeting particular criteria from the network.

- Multimedia Messaging (Parlay X 2.1)

  Using this communication service, an application can send Multimedia Messages to one or multiple addresses, set up and receive notifications for final delivery receipts of those sent items, and arrange to receive MMSes meeting particular criteria from the network or to poll for such messages.

- Terminal Location (Parlay X 2.1)

  Using this communication service, an application can request the position of one or more terminals or the distance between a given position and a terminal. It can also set up and receive notifications based on geographic location or time intervals.

- Presence (Parlay X 2.1)

  Using this communication service, an application can be a *watcher* for presence information or a *presentity* (an end user who has agreed to have certain data, such as current activity, available communication means, and contact addresses made available to others). So a presentity might say that at this moment he is in the office and prefers to be contacted by SMS at this number. Before the watcher can receive this information, it must subscribe and be approved by the presentity. Once this is done, the watcher can either poll for specific presentity information, or set up status notifications based on a wide range of criteria published by the presentity. The presentity can control the kinds of information, or attributes, that he makes available to watchers.

- Payment (Parlay X 3.0)

  Using the communication service based on this interface, an application can charge an end user's account a specific amount, refund an amount, and split costs among multiple end user accounts. An application can also reserve an amount in an account, extend the amount associated with that reservation, make a charge against that reservation, and release the reservation.

- Binary SMS (EWS)

Using the communication service based on this interface, an application can send and receive generic binary objects (for example, a vCard) using SMS mechanisms, and set up and receive notifications. This interface is not based on the Parlay X standards, but instead belongs to the Oracle Extended Web Services (EWS) set.

- WAP Push (EWS)

  The application-facing interface of this communication service is not based on the Parlay X 2.1 specification. Many elements within it, however, are based on widely distributed standards. Using the communication service based on this interface, an application can send a WAP Push message, send a replacement WAP Push message, or set up status notifications about previously sent messages.

- Subscriber Profile (EWS)

  The application-facing interface of this communication service is based on a subset of that in a proposed Parlay X version. Using this communication service, an application can retrieve particular information or an entire profile (subject to internal filtering) for a subscriber from an LDAP server attached to the network.

- Session Manager (EWS)

  Using this communication service, an application can establish a Oracle Communications Services Gatekeeper session. Whether sessions are used is up to the operator.

There are two native telephony APIs supported by Oracle Communications Services Gatekeeper.

- Native MM7

  The application-facing interface of this communication service is based on the 3GPP MM7 standard. Using the communication service based on this interface, an application can send and receive MMSs and receive status notifications about previously sent messages.

- Native SMPP

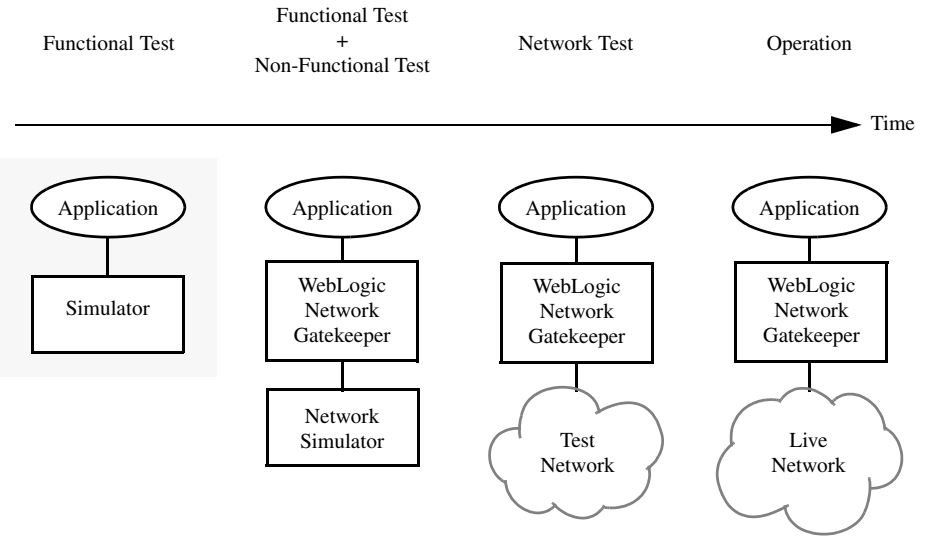
  The application-facing interface of this communication service is based on the SMS Forum standard. Using the communication service based on this interface, an application can send and receive SMSs and receive status notifications about previously sent messages.

Application Testing Workflow

Application testing in a telecom environment is usually conducted in a stepwise manner. For the first step, applications are run against simulators like the optional Oracle Communications Services Gatekeeper Simulator. The Oracle Communications Services Gatekeeper Simulator emulates both the Oracle Communications Services Gatekeeper and the underlying network, and

allows developers to sort out basic functional issues without having to be connected to a network or network simulator. Once basic functional issues are sorted through, the application is connected to an instance of the Oracle Communications Services Gatekeeper attached to a network simulator for non-functional testing. Next the application is tested against a test network, to eliminate any network related issues. Finally, the application can be placed into production on a live network. Figure 2-1 shows the complete application test flow, from the developer's functional tests to deployment in a live network. While Oracle Communications Services Gatekeeper Simulator-based tests may be performed in-house by an Application Service Provider, the other tests require the cooperation of the target network operator.

Figure 2-1 Application Testing Cycle



CHAPTER 3

Interacting with Oracle Communications Services Gatekeeper

In order to interact with Oracle Communications Services Gatekeeper, applications use either SOAP-based, RESTful, or native interfaces. Those applications using SOAP-based interfaces must manipulate the SOAP messages that they use to make requests in certain specialized ways.They must add specific information to the SOAP header, and, if they are using for example Multimedia Messaging, they must send their message payload as a SOAP attachment. Applications using the native interfaces use the normal native interface mechanisms, which are not covered in this document.

The following chapter presents a high-level description of the SOAP mechanisms and how they function to manage the interaction between Oracle Communications Services Gatekeeper and the application. It covers:

- Requirements for Using the SOAP-based Facades
  - Authentication
  - Session Management
  - Service Correlation
  - Parameter Tunneling
  - SOAP attachments

The mechanisms for dealing with these requirements programmatically depend on the environment in which the application is being developed.

Note: Clients created using Axis 1.2 or older will not work with some communication services. Developers should use Axis 1.4 or newer if they wish to use Axis.

For examples using the Oracle WebLogic Server environment to accomplish these sorts of tasks, see:

- Managing SOAP headers and SOAP attachments programmatically

Requirements for Using the SOAP-based Facades

If your application is using the a SOAP-based facade (set of interfaces) to interact with Oracle Communications Services Gatekeeper, there are four types of elements you may need to add to your application's SOAP messages to Oracle Communications Services Gatekeeper.

Authentication

In order to secure Oracle Communications Services Gatekeeper and the telecom networks to which it provides access, applications are usually required to provide authentication information in every SOAP request which the application submits. Oracle Communications Services Gatekeeper leverages the WebLogic Server Web Services Security framework to process this information.

Note: WS Security provides three separate modes of providing security between a Web Service client application and the Web Service itself for message level security - Authentication, Digital Signatures, and Encryption. For an overview of WebLogic Server Web Services Security, see *Oracle WebLogic Server Securing WebLogic Web Services* at http://download.oracle.com/docs/cd/E12840_01/wls/docs103/webserv_sec.

Oracle Communications Services Gatekeeper supports three authentication types:

- Username Token
- X.509 Certificate Token
- SAML Token

The type of token that the particular Oracle Communications Services Gatekeeper operator requires is indicated in the Policy section of the WSDL files that the operator makes available for each application-facing interface it supports. In the following WSDL fragment, for example, the required form of authentication, indicated by the `<wssp:Identity>` element, is Username Token.

Listing 3-1 WSDL fragment showing Policy

```
<s0:Policy s1:Id="Auth.xml">
       <wssp:Identity>
       <wssp:SupportedTokens>
              <wssp:SecurityToken
TokenType="http://docs.oasisopen.org/wss/2004/01/oasis200401wssusernametok
enprofile1.0#UsernameToken">
              <wssp:UsePassword
Type="http://docs.oasisopen.org/wss/2004/01/oasis200401wssusernametokenpro
file1.0#PasswordText"/>
              </wssp:SecurityToken>
              <wssp:SecurityToken
TokenType="http://docs.oasisopen.org/wss/2004/01/oasis200401wssx509tokenpr
ofile1.0#X509v3"/>
       </wssp:SupportedTokens>
       </wssp:Identity>
</s0:Policy>
<wsp:UsingPolicy n1:Required="true"/>
```

Note: If the WSDL also has a <wssp: Integrity> element, digital signing is required (WebLogic Server provides WS-Policy: sign.xml). If it has a <wssp:Confidentiality> element, encryption is required (WebLogic Server provides WS-Policy: encrypt.xml).

SOAP Header Element for Authentication

Below are examples of the three types of authentication that can be used with Oracle Communications Services Gatekeeper.

Username Token

In the Username Token mechanism, which is specified by the use of the <wsse:UsernameToken> element in the header, authentication is based on a username, specified in the <wsse:Username> element and a password, specified in the <wsse:Password> element.

Two types of passwords are possible, indicated by the Type attribute in the Password element:

- `PasswordText` indicates the password is in clear text format.
- `PasswordDigest` indicates that the sent value is a Base64 encoded, SHA-1 hash of the UTF8 encoded password.

There are two more optional elements in Username Token, introduced to provide a countermeasure for replay attacks:

- `<wsse:Nonce>`, a random value that the application creates.
- `<wsu:Created>`, a timestamp.

If either or both `the Nonce` and `Created` elements are present, the Password Digest is computed as: Password_Digest = Base64(SHA-1(nonce+created+password))

When the application sends a SOAP message using Username Token, the WSEE implementation in Oracle Communications Services Gatekeeper evaluates the username using the associated authentication provider. The authentication provider connects to the Oracle Communications Services Gatekeeper database and authenticates the username and the password. In the database, passwords are stored as MD5 hashed representations of the actual password.

Listing 3-2 Example of a WSSE: Username Token SOAP header element

```
<wsse:UsernameToken wsu:Id="Example-1">
  <wsse:Username> myUsername </wsse:Username>
  <wsse:Password Type="PasswordText">myPassword</wsse:Password>
  <wsse:Nonce EncodingType="..."> ... </wsse:Nonce>
  <wsu:Created> ... </wsu:Created>
</wsse:UsernameToken>
```

The UserName is equivalent to the application instance ID. The Password part is the password associated with this UserName when the application credentials was provisioned in Oracle Communications Services Gatekeeper.

For more information on Username Token, see http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0.pdf

X.509 Certificate Token

In the X.509 Token mechanism, the application's identity is authenticated by the use of an X.509 digital certificate.

Typically a certificate binds the certificate holder's public key with a set of attributes linked to the holder's real world identity – for example the individual's name, organization and so on. The certificate also contains a validity period in the form of two date and time fields, specifying the beginning and end of the interval during which the certificate is recognized.

The entire certificate is (digitally) signed with the key of the issuing authority. Verifying this signature guarantees

- that the certificate was indeed issued by the authority in question
- that the contents of the certificate have not been forged, or tampered with in any way since it was issued

For more information on X.509 Token, see
http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0.pdf

The default identity assertion provider in Oracle Communications Services Gatekeeper verifies the authenticity of X.509 tokens and maps them to valid Oracle Communications Services Gatekeeper users.

Note: While it is possible to use the out-of-the-box keystore configuration in Oracle Communications Services Gatekeeper for testing purposes, these should not be used for production systems. The digital certificates in these out-of-the-box keystores are only signed by a demonstration certificate authority For information on configuring keystores for production systems, see the section about configuring identity and trust in *Oracle WebLogic Server Securing WebLogic Server* at http://download.oracle.com/docs/cd/E12840_01/wls/docs103/secmanage/.

The x.509 certificate common name (CN) for an application must be the same as the account UserName, which is the string that was referred to as the `applicationInstanceGroupId` in previous versions of Oracle Communications Services Gatekeeper. This is provided by the operator when the account is provisioned.

Listing 3-3 Example of a WSSE: X.509 Certificate SOAP header element

```
<wsse:Security xmlns:wsse="..." xmlns:wsu="...">
  <wsse:BinarySecurityToken wsu:Id="binarytoken"
```

```
    ValueType="wsse:X509v3"
    EncodingType="wsse:Base64Binary">
      MIIEZzCCA9CgAwIBAgIQEmtJZc0…
  </wsse:BinarySecurityToken>
  <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
  <ds:SignedInfo>
    <ds:Reference URI="#body">…</ds:Reference>
    <ds:Reference URI="#binarytoken">…</ds:Reference>
  </ds:SignedInfo>
  <ds:SignatureValue>HFLP…</ds:SignatureValue>
    <ds:KeyInfo>
      <wsse:SecurityTokenReference>
        <wsse:Reference URI="#binarytoken" />
      </wsse:SecurityTokenReference>
    </ds:KeyInfo>
  </ds:Signature>
</wsse:Security>
```

SAML Token

Oracle Communications Services Gatekeeper, using WebLogic Server's WSSE implementation, supports SAML versions 1.0 and 1.1. The versions are similar. See http://www.oasis-open.org/committees/download.php/3412/sstc-saml-diff-1.1-draft-01.pdf for an overview of the differences between the versions.

In SAML, a third party, the Asserting Party, provides the identity information for a Subject that wishes to access the services of a Relying Party. This information is carried in an Assertion. In the SAML Token type of Authentication, the Assertion (or a reference to an Assertion) is provided inside the `<WSSE:Security>` header in the SOAP message. The Relying Party (which in this case is Oracle Communications Services Gatekeeper, using the WebLogic Security

framework) then evaluates the trustworthiness of the assertion, using one of two confirmation methods.

- Holder-of-Key
- Sender-Voucher

For more information on these confirmation methods, see the sections about SAML Token Profile in *Oracle Weblogic Server Understanding WebLogic Security* at http://download.oracle.com/docs/cd/E12840_01/wls/docs103/secintro/.

Listing 3-4 Example of a WSSE: SAML Token SOAP header element

```
<wsse:Security>
<saml:Assertion MajorVersion="1" MinorVersion="0"
    AssertionID="186CB370-5C81-4716-8F65-F0B4FC4B4A0B"
    Issuer="www.test.com" IssueInstant="2001-05-31T13:20:00-05:00">
  <saml:Conditions NotBefore="2001-05-31T13:20:00-05:00"
    NotAfter="2001-05-31T13:25:00-05:00"/>
  <saml:AuthenticationStatement AuthenticationMethod="password"
    AuthenticationInstant="2001-05-31T13:21:00-05:00">
    <saml:Subject>
      <saml:NameIdentifier>
        <SecurityDomain>"www.bea.com"</SecurityDomain>
        <Name>"cn=localhost,co=bea,ou=sales"</Name>
      </saml:NameIdentifier>
    </saml:Subject>
  </saml:AuthenticationStatement>
</saml:Assertion>
...
</wsse:Security>
```

Session Management

Oracle Communications Services Gatekeeper can be configured to run in session mode or sessionless mode. In session mode, an application must establish a session using the Session Manager Web Service before it is allowed to run traffic through Oracle Communications Services Gatekeeper. The session allows Oracle Communications Services Gatekeeper to keep track of all of the traffic sent by a particular application for the duration of the session, which lasts until the session times out, based on an operator-set interval, or until the application closes the session. The session is good for an entire Oracle Communications Services Gatekeeper domain, across clusters, and covers all communication services to which the application has contractual access.

In sessionless mode, the application is not required to establish a session.

Session Mode

An application establishes a session in Oracle Communications Services Gatekeeper by invoking the `getSession()` operation on the Session Manager Web Service. This is the only request that does not require a SessionID. In the response to this operation, a string representing the Session ID is returned to the client, and an Oracle Communications Services Gatekeeper session, identified by the ID, is established. The session is valid until either the session is terminated by the application or an operator-established time period has elapsed. The SessionID must appear in the `wlng:Session` element in the header of every subsequent SOAP request.

Listing 3-5 Example of a SessionID SOAP header element

```
<Session>
  <SessionId>app:-2810834922008400383</SessionId>
</Session>
```

Sessionless Mode

It is also possible to run Oracle Communications Services Gatekeeper without using the session mechanism. In this case the application simply uses whichever WS-Security mechanism is required by the Oracle Communications Services Gatekeeper operator.

Service Correlation

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

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

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

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

The SCID is passed between Oracle Communications Services Gatekeeper and the application through an additional SOAP header element, the SCID element. Because not every application requires the service correlation facility, this is an optional element.

Listing 3-6 Example of a SCID SOAP header element

```
<scid>myId</scid>
```

Parameter Tunneling

Parameter tunneling is a feature that allows an application to send additional parameters to Oracle Communications Services Gatekeeper and lets a plug-in use these parameters. This feature makes it possible for an application to tunnel parameters that are not defined in the application-facing interface and can be seen as an extension to the it.

The application sends the tunneled parameters in the SOAP header of a Web Services request.

The parameters are defined using key-value pairs encapsulated by the tag **<xparams>**. The xparams tag can include one or more **<param>** tags. Each **<param>** tag has a **key** attribute that identifies the parameter and a **value** attribute that defines the value of the parameter. In the example below, the application tunnels the parameter `aParameterName` and assigns it the value `aParameterValue`.

Listing 3-7 SOAP header with a tunneled parameter.

```
<soapenv:Header>
...
  <xparams>
    <param key="aParameterName" value="aParameterValue" />
  </xparams>
...
</soapenv:Header>
```

Depending on the plug-in the request reaches, the parameter is fetched and used in the request towards the network node.

SOAP attachments

In some communication services, the request payload are sent as SOAP attachments. Listing 3-8 below shows a Multimedia Messaging `sendMessage` operation that contains an attachment carrying a jpeg image.

Listing 3-8 Example of a SOAP message with attachment (full content is not shown)

```
POST /parlayx21/multimedia_messaging/SendMessage HTTP/1.1

Content-Type: multipart/related; type="text/xml";
start="<1A07DC767BC3E4791AF25A04F17179EE>";
boundary="----=_Part_0_2633821.1170785251635"

Accept: application/soap+xml, application/dime, multipart/related, text/*

User-Agent: Axis/1.4

Host: localhost:8000

Cache-Control: no-cache

Pragma: no-cache

SOAPAction: ""

Content-Length: 4652

Connection: close

------=_Part_0_2633821.1170785251635

Content-Type: text/xml; charset=UTF-8

Content-Transfer-Encoding: binary

Content-Id: <1A07DC767BC3E4791AF25A04F17179EE>

  <?xml version="1.0" encoding="UTF-8"?>

    <soapenv:Envelope
      xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"

      xmlns:xsd="http://www.w3.org/2001/XMLSchema"

      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

      <soapenv:Header>

        <ns1:Security ns1:Username="app:-4206293882665579772"

          ns1:Password="app:-4206293882665579772"

          soapenv:actor="wsse:PasswordToken"

          soapenv:mustUnderstand="1"
          xmlns:ns1="/parlayx21/multimedia_messaging/SendMessage">
```

```
        </ns1:Security>
      </soapenv:Header>
      <soapenv:Body>
        <sendMessage xmlns=
          "http://www.csapi.org/schema/parlayx/multimedia_messaging/send/v2_4/
local">
          <addresses>tel:234</addresses>
          <senderAddress>tel:567</senderAddress>
          <subject>Default Subject Text</subject>
          <priority>Normal</priority>
          <charging>
            <description xmlns="">Default</description>
            <currency xmlns="">USD</currency>
            <amount xmlns="">1.99</amount>
            <code xmlns="">Example_Contract_Code_1234</code>
          </charging>
        </sendMessage>
      </soapenv:Body>
    </soapenv:Envelope>
------=_Part_0_2633821.1170785251635
Content-Type: image/jpeg
Content-Transfer-Encoding: binary
Content-Id:

  <9FFD47E472683C870ADE632711438CC3>???? JFIF      ??
C#%$""!&+7/&)4)!"0A149;>>>%.DIC<H7=>;??
C;("(;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;??  ? w" ??           ??
7      !1AQ"aq2???#?BRr?3Cb?????             ?? '         !1"AQ2Raq???? ?
??{?????>?"7B?7!1???????Z e{????ax??5??CC??-Du?
??X?)Y!??=R@??g?????T??c????f?Wc??eCi?l?????5s??\E???6I??(?x?^???=??d?#?itoi?{
;? ??G.......
```

```
------=_Part_0_2633821.1170785251635--
```

Managing SOAP headers and SOAP attachments programmatically

This section illustrates how to manage the Oracle Communications Services Gatekeeper required SOAP headers and SOAP attachments when you are using WebLogic Server and WebLogic Server tools to generate stubs for your Web Services clients. If you are using a different environment, the steps you need to take to accomplish these tasks will be different.

For an overview of using WebLogic Server to create Web Service clients, see the sections discussing invoking Web Services in *Oracle WebLogic Server Programming Web Services for WebLogic Server* at http://download.oracle.com/docs/cd/E12840_01/wls/docs103/webserv/.

The following examples show particularly the use of a SOAP message handler.

These examples show the use of a single message handler to add both SOAP Headers and SOAP attachments.

The WebLogic Server environment relies heavily on using supplied Ant tasks. In Listing 3-9 a supplied Ant task, `clientgen`, is added to the standard build.xml file. A handler configuration file, `SOAPHandlerConfig.xml` is added as the value for the `handlerChainFile` attribute. `SOAPHandlerConfig.xml` is shown in Listing 3-10.

Listing 3-9 Snippet from build.xml

```
<clientgen
  wsdl="${wsdl-file}"
  destDir="${class-dir}"
  handlerChainFile="SOAPHandlerConfig.xml"
  packageName="com.bea.wlcp.wlng.test"
  autoDetectWrapped="false"
  generatePolicyMethods="true"
/>
```

The configuration file for the message handler contains the handler-name and the associated handler-class. The handler class, `TestClientHandler`, is described in Listing 3-11.

Listing 3-10 SOAPHandlerConfig.xml

```
<weblogic-wsee-clientHandlerChain
  xmlns="http://www.bea.com/ns/weblogic/90"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:j2ee="http://java.sun.com/xml/ns/j2ee">
  <handler>
    <j2ee:handler-name>clienthandler1</j2ee:handler-name>
    <j2ee:handler-class>
      com.bea.wlcp.wlng.client.TestClientHandler
    </j2ee:handler-class>
  </handler>
</weblogic-wsee-clientHandlerChain>
```

`TestClientHandler` provides the following functionality:

- Adds a Session ID to the SOAP header, see Session Management. The session ID is hardcoded into the member variable `sessionId`.
- Adds a service correlation ID to the SOAP header. See Service Correlation for more information.
- Adds a SOAP attachment in the form of a MIME message with content-type text/plain. See SOAP attachments for more information.

Listing 3-11 TestClientHandler

```
package com.bea.wlcp.wlng.client;
import javax.xml.rpc.handler.Handler;
```

```
import javax.xml.rpc.handler.HandlerInfo;
import javax.xml.rpc.handler.MessageContext;
import javax.xml.rpc.handler.soap.SOAPMessageContext;
import javax.xml.soap.*;
import javax.xml.namespace.QName;
public class TestClientHandler implements Handler{
  public String sessionId = "myID";
  public String SCID = "mySCId";
  public String contenttype = "text/plain";
  public String content = "The content";

  public boolean handleRequest(MessageContext ctx) {
    if (ctx instanceof SOAPMessageContext) {
      try {
        SOAPMessageContext soapCtx = (SOAPMessageContext) ctx;
        SOAPMessage soapmsg = soapCtx.getMessage();
        SOAPHeader header = soapCtx.getMessage().getSOAPHeader();
        SOAPEnvelope envelope =
          soapCtx.getMessage().getSOAPPart().getEnvelope();
        // Begin: Add session ID
        Name headerElementName = envelope.createName("session","",
          "http://schemas.xmlsoap.org/soap/envelope/");
        SOAPHeaderElement headerElement =
          header.addHeaderElement(headerElementName);
        headerElement.setMustUnderstand(false);
        headerElement.addNamespaceDeclaration("soap",
          "http://schemas.xmlsoap.org/soap/envelope/");
```

```
        SOAPElement sessionId = headerElement.addChildElement("SessionId");
        sessionId.addTextNode(sessionId);
        // End: Add session ID
        // Begin: Add Combined Services ID
        Name headerElementName = envelope.createName("SCID","",
          "http://schemas.xmlsoap.org/soap/envelope/");
        SOAPHeaderElement headerElement =
          header.addHeaderElement(headerElementName);
        headerElement.setMustUnderstand(false);
        headerElement.addNamespaceDeclaration("soap",
          "http://schemas.xmlsoap.org/soap/envelope/");
        SOAPElement sessionId = headerElement.addChildElement("SCID");
        sessionId.addTextNode(SCID);
        // End: Add Combined Services ID
        // Begin: Add SOAP attachment
        AttachmentPart part = soapmsg.createAttachmentPart();
        part.setContent(content, contenttype);
        soapmsg.addAttachmentPart(part);
        // End: Add SOAP attachment
      } catch (Exception e) {
        e.printStackTrace();
      }
    }
    return true;
  }
  public boolean handleResponse(MessageContext ctx) {
    return true;
```

```
  }
  public boolean handleFault(MessageContext ctx) {
    return true;
  }
  public void init(HandlerInfo config) {
  }
  public void destroy() {
  }
  public QName[] getHeaders() {
    return null;
  }
}
```

Session Manager Web Service

The Session Manager Web Service contains operations for establishing a session with Oracle Communications Services Gatekeeper, changing the application's password, querying the amount of time remaining in the session, refreshing the session, and terminating the session.

Note: Not all installations of Oracle Communications Services Gatekeeper require session management. The contents of this chapter apply only to those installations that do.

When an operator requires it, an application must establish a session with Oracle Communications Services Gatekeeper before the application can perform any operations on the Parlay X or Extended Web Services interfaces. When a session is established, a session ID is returned which must be used in each subsequent operation towards Oracle Communications Services Gatekeeper.

Endpoint

The WSDL for the Session Manager can be found at
`http://<host>:<port>/session_manager/SessionManager`

where host and port depend on the Oracle Communications Services Gatekeeper deployment.

Interface: SessionManager

Operations to establish a session, change a password, get the remaining lifetime of a session, refresh a session and destroy a session.

Operation: getSession

Establishes a session using Web Services Security. Authentication information must be provided according to WS-Security. See Authentication.

Input message: getSession

Part name	Part type	Optional	Description
-	-	-	-

Output message: getSessionResponse

Part name	Part type	Optional	Description
getSessionReturn	xsd:String	N	The session ID to use in subsequent requests.

Referenced faults

GeneralException

Operation: changeApplicationPassword

Changes the password for an application.

Input message: changeApplicationPassword

Part name	Part type	Optional	Description
sessionId	xsd:string	N	The ID of an established session.

Part name	Part type	Optional	Description
oldPassword	xsd:string	N	The current password.
newPassword	xsd:string	N	The new password.

Output message: changeApplicationPasswordResponse

Part name	Part type	Optional	Description
-	-	-	-

Referenced faults

-

Operation: getSessionRemainingLifeTime

Gets the remaining lifetime of an established session. The default lifetime is configured in Oracle Communications Services Gatekeeper.

Input message: getSessionRemainingLifeTime

Part name	Part type	Optional	Description
sessionId	xsd:string	N	The ID of an established session.

Output message: getSessionRemainingLifeTimeResponse

Part name	Part type	Optional	Description
getSessionRemainingLifeTimeReturn	xsd:string	N	The remaining lifetime of the session. Given in milliseconds.

Referenced faults

-

Operation: refreshSession

Refreshes the lifetime of an session. The session can be refreshed during a time interval after the a session has expired. This time interval is configured in Oracle Communications Services Gatekeeper.

Input message: refreshSession

Part name	Part type	Optional	Description
sessionId	xsd:string	N	The ID of an established session.

Output message: refreshSessionResponse

Part name	Part type	Optional	Description
refreshSessionReturn	xsd:string	N	The session ID to be used in subsequent requests. The same ID as the original session ID is returned.

Referenced faults

-

Operation: destroySession

Destroys an established session.

Input message: destroySession

Part name	Part type	Optional	Description
sessionId	xsd:string	N	The ID of an established session.

Output message: destroySessionResponse

Part name	Part type	Optional	Description
destroySessionReturn	xsd:boolean	N	True if the session was destroyed.

Referenced faults

-

Examples

The code below illustrates how to get the Session Manager and how to prepare the generated stub with Web Service security information. The stub is generated from the Session Manager Web Service.

Listing 4-1 Get hold of the Session Manager

```
protected ClientSessionManImpl(String sessionManagerURL, PolicyBase pbase)
throws Exception {

  SessionManagerService accessservice =

    new SessionManagerService_Impl(sessionManagerURL+"?WSDL");
```

```
  port = accessservice.getSessionManager();
  pbase.prepareStub((Stub)port);
  }
```

Below illustrates how to prepare the Session Manager stub with Username Token information according to WS-Policy.

Listing 4-2 Prepare the Session Manager with Username Token information

```
package com.bea.wlcp.wlng.client.access.wspolicy;
import weblogic.wsee.security.unt.ClientUNTCredentialProvider;
import weblogic.xml.crypto.wss.WSSecurityContext;
import javax.xml.rpc.Stub;
import java.util.ArrayList;
import java.util.List;
public class UsernameTokenPolicy implements PolicyBase {

  private String username;
  private String password;

public UsernameTokenPolicy(String username, String password) {
    this.username = username;
    this.password = password;
  }

  public void prepareStub(Stub stub) throws Exception {
    List<ClientUNTCredentialProvider> credProviders = new
ArrayList<ClientUNTCredentialProvider>();
```

```
    credProviders.add(new ClientUNTCredentialProvider(username.getBytes(),
                                                      password.getBytes()));
    System.out.println("setting standard wssec");
    stub._setProperty(WSSecurityContext.CREDENTIAL_PROVIDER_LIST,
                      credProviders);
  }

}
```

CHAPTER 5

Extended Web Services Binary SMS

The Extended Web Services Binary SMS Web Service allows for the sending and receiving of any generic binary content via SMS. Both application-initiated and network-triggered requests are supported. The binary content can be other than the Logos and Ringtones as specified by Parlay X Short Messaging. An example would be a vCard.

Namespaces

The BinarySMS interface and service use the namespaces:

- http://www.bea.com/wlcp/wlng/wsdl/ews/binary_sms/interface
- http://www.bea.com/wlcp/wlng/wsdl/ews/binary_sms/service

The BinarySmsNotificationManager interface and service use the namespaces:

- http://www.bea.com/wlcp/wlng/wsdl/ews/binary_sms/notification/interface
- http://www.bea.com/wlcp/wlng/wsdl/ews/binary_sms/notification/service

In addition, Extended Web Services Binary SMS uses common data type definitions common for all Extended Web Services interfaces, see Extended Web Services Common.

Fault definitions are according to ETSI ES 202 391-1 V1.2.1 (2006-10) Open Service Access (OSA); Parlay X Web Services; Part 1: Common (Parlay X 2).

Endpoints

The endpoint for the BinarySMS interface is:
`http://<host:port>/ews/binary_sms/BinarySms`

The endpoint for the BinarySmsNotificationManager interface is:
`http://<host:port>/ews/binary_sms_notification/BinarySmsNotificationManager`

The values for host and port depend on the specific Oracle Communications Services Gatekeeper deployment.

Sequence Diagram

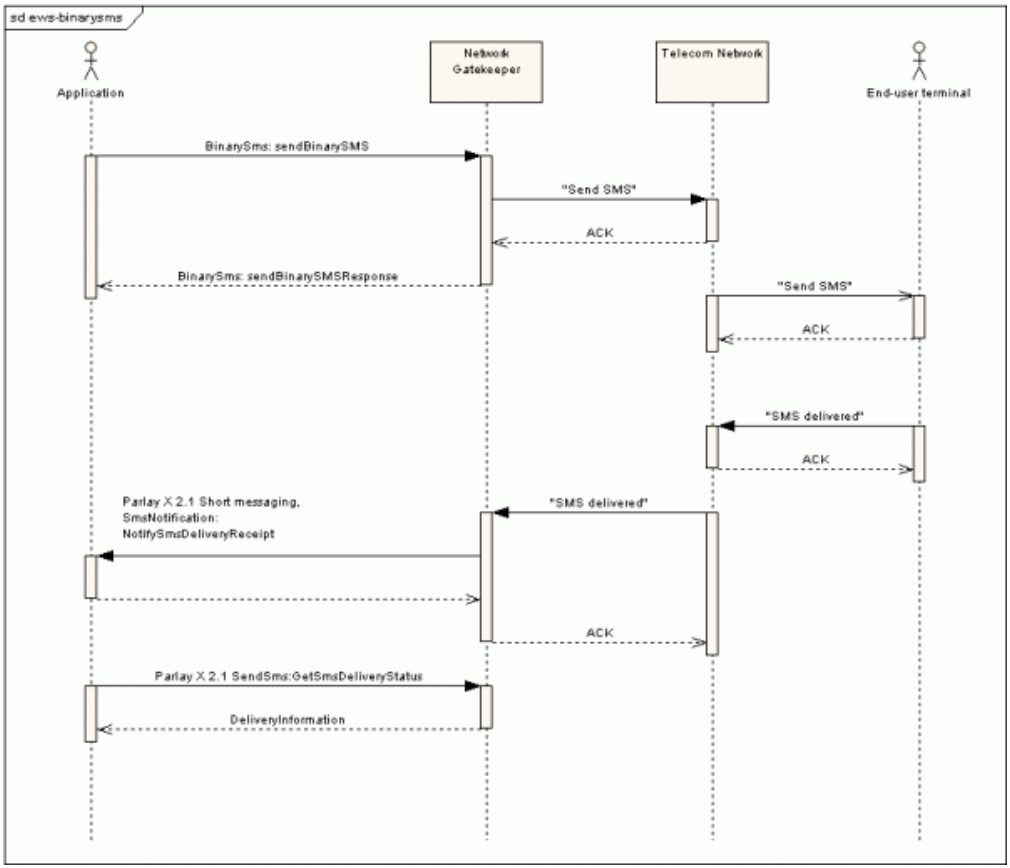
Send SMS

The following diagram shows the general message sequence for sending a binary SMS message from an Extended Web Services Binary SMS application to the network. In this message sequence the application also receives a notification from the network indicating the delivery status of the SMS, that is that the message has reached its destination. It also displays how an application can query the delivery status of the message.

The interaction between the network and Oracle Communications Services Gatekeeper is illustrated in a protocol-agnostic manner. The exact operations and sequences depend on which network protocol is being used.

Note: The delivery notifications are sent from the Parlay X 2.1 Short Messaging implementation.

Figure 5-1 Sequence diagram Application-initiated send Extended Web Services Binary SMS

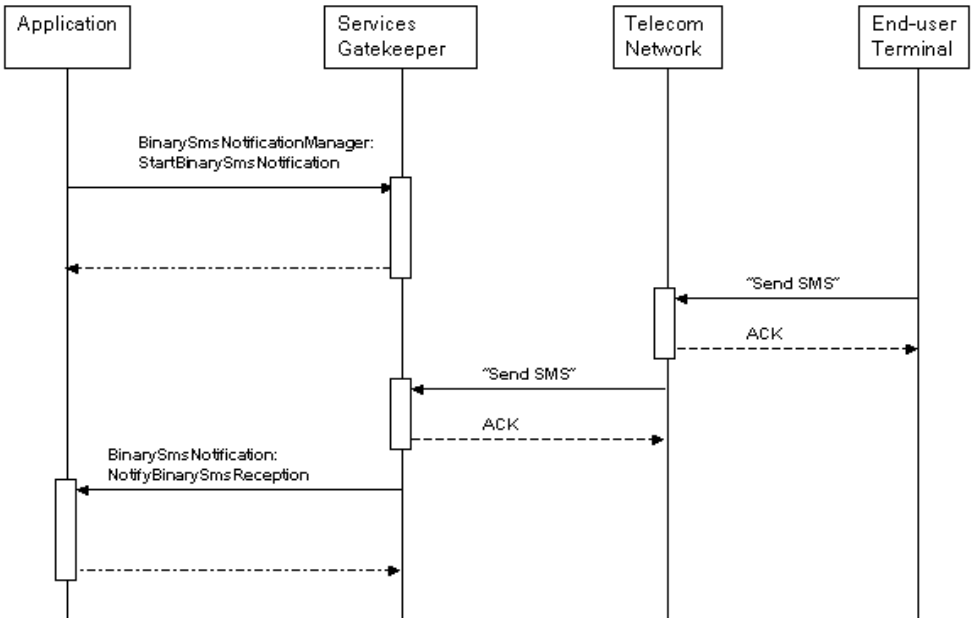


Receive SMS

The following diagram shows the general message sequence for receiving a binary SMS message from the Network using Oracle Communications Services Gatekeeper. In this message sequence the application also subscribes for a notifications on network triggered short messages.

The interaction between the network and Oracle Communications Services Gatekeeper is illustrated in a protocol-agnostic manner. The exact operations and sequences depend on which network protocol is being used.

Figure 5-2 Sequence diagram receive Extended Web Services Binary SMS



XML Schema data type definition

The following data structures are used in the Extended Web Services Binary SMS Web Service.

BinaryMessage structure

Defines the binary payload of the SMS for application-initiated messages.

Defines the TP-User Data (TP-UD).

For a description of TP-User Data (TP-UD), TP-User-Data-Header-Indicator (TP UDHI), see *3GPP TS 23.040 V6.5.0, Technical realization of the Short Message Service (SMS)* at http://www.3gpp.org/ftp/Specs/html-info/23040.htm.

Element Name	Element type	Optional	Description
udh	xsd:base64Binary	Y	Defines if the TP-User Data (TP-UD) field contains only the short message or if it contains a header in addition to the short message. Must be formatted according to TP-User-Data-Header-Indicator (TP UDHI).
message	xsd:base64Binary	Y	Binary message data. Must be formatted according to TP-User Data (TP-UD) excluding the TP-User-Data-Header-Indicator (TP UDHI).

BinarySmsMessage structure

Defines the binary payload of the SMS for network-triggered messages.

Element Name	Element type	Optional	Description
message	ews_binary_sms_xsd:BinaryMessage[1..unbounded]	N	See BinaryMessage structure.
dcs	xsd:byte	N	Data code schema, according to SMPP v3.4.

Element Name	Element type	Optional	Description
protocolId	xsd:byte	Y	TP-Protocol-Identifier according to 3GPP 23.040 6.5.0. Defines the TP-User Data (TP-UD). For a description of TP-User Data (TP-UD), TP-User-Data-Header-Indicator (TP UDHI), see *3GPP TS 23.040 V6.5.0, Technical realization of the Short Message Service (SMS)* at http://www.3gpp.org/ftp/Specs/html-info/23040.htm. The protocol identifier is the information element by which the short message transport layer either refers to the higher layer protocol being used, or indicates interworking with a certain type of telematic device. Example: 123
senderAddress	xsd:anyURI	N	The address of the sender of the short message. Example: tel:1234556
smsServiceActivationNumber	xsd:anyURI	N	The destination address of the short message. Example: tel:1222
dateTime	xsd:dateTime	N	The timestamp of the message.

Interface: BinarySms

Operations to send SMSs with binary content.

Operation: sendBinarySMS

Sends an SMS with any binary data as content.

Input message: sendBinarySMS

Part name	Part type	Optional	Description
addresses	xsd:anyURI[1. .unbounded]	N	An array of end-user terminal addresses. Example: tel:1234
senderName	xsd:string	Y	The name of the sender. Alphanumeric. Example: tel:7485, Mycompany.
dcs	xsd:byte	N	Defines the data encoding scheme for the binaryMessage parameter. Formatted according to data_coding parameter in SMPP v3.4. See http://www.smsforum.net/
binaryMessage	binary_sms_xsd:BinaryMessage[1..unbounded]	N	Message payload. An array comprised of UDH elements and message elements, see BinaryMessage structure. This array must be equal to or less than 140 bytes in size.
protocolId	xsd:byte	Y	TP-Protocol-Identifier (TP-PID) according to *3GPP TS 23.040 V6.5.0, Technical realization of the Short Message Service (SMS)* at http://www.3gpp.org/ftp/Specs/html-info/23040.htm. Specifies the higher layer protocol being used, or indicates interworking with a certain type of telematic device.
validityPeriod	xsd:string	Y	Defines the validity period for the short message. Formatted according to validity_period parameter in SMPP v3.4. See http://www.smsforum.net/

Part name	Part type	Optional	Description
charging	ews_common _xsd:Charging Information	Y	Charging information. See ChargingInformation structure.
receiptReque st	ews_common _xsd:SimpleR eference	Y	It defines the application endpoint, interfaceName and correlator that will be used to notify the application when the message has been delivered to the terminal or if delivery is impossible. See SimpleReference structure

Output message: sendBinarySMSResponse

Part name	Part type	Optional	Description
result	xsd:string	N	Identifies a specific SMS delivery request.

Referenced faults

Table 5-1 exceptions and error codes

Exception	Error code	Reason/Action
SVC0001	BSMS-000001	Unable to perform action. Network error
SVC0001	BSMS-000002	Unable to retrieve configuration, internal error.
SVC0001	BSMS-000003	The used address type is not supported
SVC0001	BSMS-000004	Unable to encode message segments. make sure the number of message segments is not 0.
SVC0001	BSMS-000005	GSM message format error.
SVC0001	BSMS-000006	Binary Message has too many segments.
SVC0001	PLG-000004	General plug-in routing error.

Table 5-1 exceptions and error codes

Exception	Error code	Reason/Action
SVC0002	n/a	SenderName in non-alphanumeric format.
SVC0003	n/a	
SVC0004	n/a	
SVC0005	n/a	
EPOL0001	n/a	

Interface: BinarySmsNotificationManager

Operations to start and stop subscriptions for notifications for short messages with binary content.

Operation: StartBinarySmsNotification

Starts a subscription for notifications for short messages that have content in the form of binary data. A correlator is provided in the request. This correlator is used when stopping the subscription.

Input message: StartBinarySmsNotification

Part name	Part type	Optional	Description
reference	ews_common_xsd:SimpleReference	N	Defines the application endpoint, interfaceName and correlator that will be used to forward a binary short message from the network. See SimpleReference structure
smsServiceActivationNumber	xsd:xsd:anyURI	Y	The destination address of the short message.

Output message: StartBinarySmsNotificationResponse

Part name	Part type	Optional	Description
-	-	-	-

Referenced faults

Table 5-2 exceptions and error codes

Exception	Error code	Reason/Action
SVC0001	BSMS-000001	Unable to perform action. Network error
SVC0001	BSMS-000002	Unable to retrieve configuration, internal error.
SVC0001	BSMS-000003	The used address type is not supported
SVC0001	BSMS-000004	Unable to encode message segments. make sure the number of message segments is not 0.
SVC0001	BSMS-000005	GSM message format error.
SVC0001	BSMS-000006	Binary Message has too many segments.
SVC0001	PLG-000004	General plug-in routing error.
SVC0002	n/a	
SVC0003	n/a	
SVC0004	n/a	
SVC0005	n/a	
EPOL0001	n/a	

Operation: StopBinarySmsNotification

Stops a previously started subscription for notifications for short messages that have content in the form of binary data. A correlator is provided in the request. This correlator was provided when the subscription was started, see Operation: StartBinarySmsNotification.

Input message: StopBinarySmsNotification

Part name	Part type	Optional	Description
correlator	xsd:String	N	The identifier for the subscription.

Output message: StopBinarySmsNotificationResponse

Part name	Part type	Optional	Description
-	-	-	-

Referenced faults

Table 5-3 exceptions and error codes

Exception	Error code	Reason/Action
SVC0001	BSMS-000001	Unable to perform action. Network error
SVC0001	BSMS-000002	Unable to retrieve configuration, internal error.
SVC0001	BSMS-000003	The used address type is not supported
SVC0001	BSMS-000004	Unable to encode message segments. make sure the number of message segments is not 0.
SVC0001	BSMS-000005	GSM message format error.
SVC0001	BSMS-000006	Binary Message has too many segments.

Table 5-3 exceptions and error codes

Exception	Error code	Reason/Action
SVC0001	PLG-000004	General plug-in routing error.
SVC0002	n/a	
SVC0003	n/a	
SVC0004	n/a	
SVC0005	n/a	
EPOL0001	n/a	

Interface: BinarySmsNotification

This interface shall be implemented by the application. It is used by Oracle Communications Services Gatekeeper to deliver short messages with binary content to an application. Only messages that match a previously started subscription for notifications are delivered.

Note: Notifications on delivered short messages are delivered using the Parlay X 2.1 Short Messaging SmsNotification interface, using the method NotifySmsDeliveryReceipt.

Operation: NotifyBinarySmsReception

Oracle Communications Services Gatekeeper calls this methods on

The notification is used to send a short message with binary content to the application. The notification occurs if the short message matched the criteria specified when starting the notification. See Operation: StartBinarySmsNotification.

The method must be implemented by a Web Service at the application side. It is be invoked by Oracle Communications Services Gatekeeper when it receives a short message with binary content form the network and the criteria is fulfilled.

Input message: NotifyBinarySmsReceptionRequest

Part name	Part type	Description
correlator	xsd:String	The correlator for the subscription.
message	ews_binary_sms_xsd:BinarySmsMessage	The message in binary form. See BinarySmsMessage structure.

Output message: NotifyBinarySmsReceptionResponse

Part name	Part type	Optional	Description
-	-	-	-

Referenced faults

Table 5-4 Exceptions and error codes

Exception	Error code	Reason/Action
-	-	-

WSDLs

The document/literal WSDL representation of the interfaces can be retrieved from the Web Services endpoints, see Endpoints.

The notification interface can be downloaded from:

`http://host:port/ews/binary_sms_notification/wsdls/ews_binary_sms_notification_service.wsdl`

`http://host:port/ews/binary_sms_notification/wsdls/ews_binary_sms_notification_interface.wsdl`

Where host and port are depending on the Oracle Communications Services Gatekeeper deployment.

Error Codes

The following error codes are defined for SVC0001: Service error:

- See General error codes.
- Error codes defined for Parlay X 2.1 Short Messaging, see Error Codes.
- 16133 Too many segments in message.

The following error codes are defined for EPOL0001: Policy error:

- See Code examples.
- Policy error codes defined for Parlay X 2.1 Short Messaging, see Error Codes.

Sample Send Binary SMS

Listing 5-1 Example Send Binary SMS

```
BinarySmsService service = new
BinarySmsService_Impl(“http://localhost:8001/ews/binary_sms/BinarySms?WSDL”);

BinarySms port = service.getBinarySms();

com.bea.wlcp.wlng.schema.ews.binary_sms.local.SendBinarySms parameters =

new com.bea.wlcp.wlng.schema.ews.binary_sms.local.SendBinarySms();

URI[] addresses = new URI[1];

addresses[0] = new URI("tel:1234");

parameters.setAddresses(addresses);

parameters.setDcs((byte)0);
```

```
parameters.setProtocolId((byte)0x7b);

parameters.setSenderName("tel:7878");

parameters.setValidityPeriod("020610233429000R");

com.bea.wlcp.wlng.schema.ews.binary_sms.BinaryMessage[] binaryMessages =

new com.bea.wlcp.wlng.schema.ews.binary_sms.BinaryMessage[1];

binaryMessages[0] = new
com.bea.wlcp.wlng.schema.ews.binary_sms.BinaryMessage();

byte[] udh = {0};

byte[] message = {0x4d, 0x61, 0x64, 0x65, 0x20, 0x69, 0x6e, 0x20, 0x2e};

binaryMessages[0].setUdh(udh);

binaryMessages[0].setMessage(message);

parameters.setBinaryMessage(binaryMessages);

port.sendBinarySms(parameters);
```

CHAPTER 6

Extended Web Services WAP Push

The Extended Web Services WAP Push Web Service allows for the sending of messages, which are rendered as WAP Push messages by the addressee's terminal. The content of the message is coded as a PAP message. It also provides an asynchronous notification mechanism for delivery status.

The payload of a WAP Push message must adhere to the following:

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

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

The payload is sent as a SOAP attachment.

Namespaces

The PushMessage interface and service use the namespaces:

- http://www.bea.com/wlcp/wlng/wsdl/ews/push_message/interface
- http://www.bea.com/wlcp/wlng/wsdl/ews/push_message/service

The PushMessageNotification interface and service use the namespaces:

- http://www.bea.com/wlcp/wlng/wsdl/ews/push_message/notification/interface
- http://www.bea.com/wlcp/wlng/wsdl/ews/push_message/notification/service

The data types are defined in the namespace:

- http://www.bea.com/wlcp/wlng/schema/ews/push_message

In addition, Extended Web Services WAP Push uses definitions common for all Extended Web Services interfaces:

- The datatypes are defined in the namespace:
  - http://www.bea.com/wlcp/wlng/schema/ews/common
- The faults are defined in the namespace:
  - targetNamespace="http://www.bea.com/wlcp/wlng/wsdl/ews/common/faults"

Endpoint

The endpoint for the PushMessage interface is:
`http://<host:port>/ews/push_message/PushMessage`

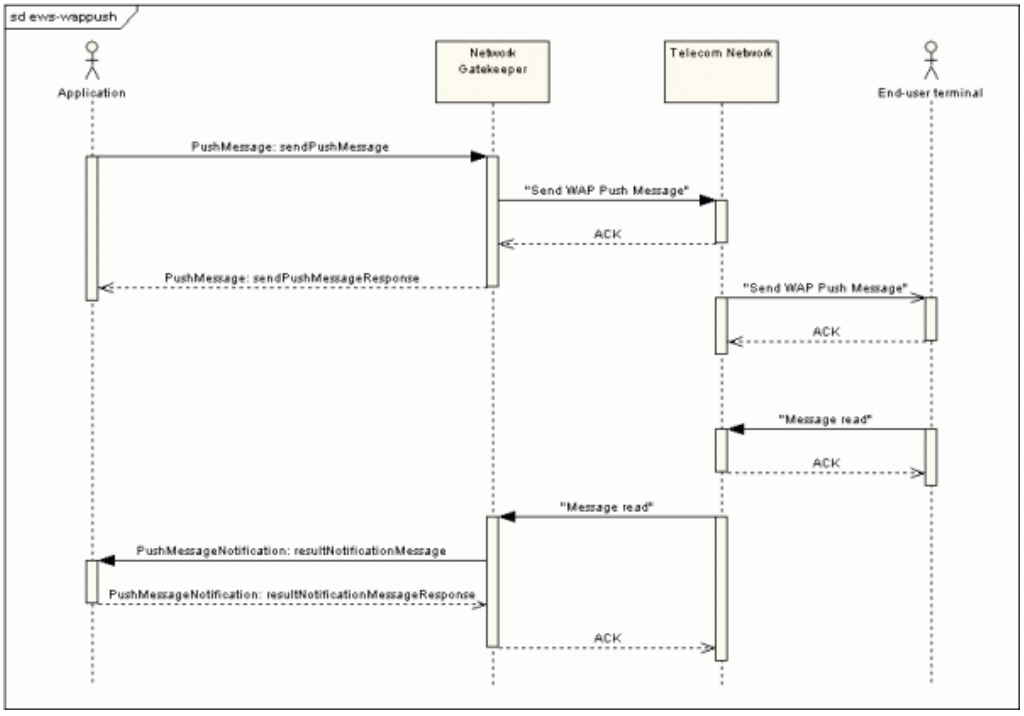
Where host and port depend on the Oracle Communications Services Gatekeeper deployment.

Sequence Diagram

The following diagram shows the general message sequence for sending a WAP Push message from an Extended Web Services WAP Push application to the network. In this message sequence the application also receives a notification from the network indicating the delivery status of the WAP Push message, that is that the message has been read The interaction between the network and Oracle Communications Services Gatekeeper is illustrated in a protocol-agnostic manner. The exact operations and sequences depend on which network protocol is being used.

Note: Zero or more `resultNotificationmesages` are sent to the application, depending on parameters provided in the initial `SendPushMessage` request.

Figure 6-1 Sequence diagram Extended Web Services WAP Push



XML Schema data type definition

The following data structures are used in the Extended Web Services WAP Push Web Service.

PushResponse structure

Defines the response that the Oracle Communications Services Gatekeeper returns from a `sendPushMessage` operation.

Element Name	Element type	Optional	Description
result	push_message_xsd:ResponseResult	N	The ResponseResult allows the server to specify a code for the outcome of sending the push message. See ResponseResult structure
pushId	xsd:string	N	The push ID provided in the request.
senderAddress	xsd:string	Y	Contains the address to which the message was originally sent, for example the URL to the network node.
senderName	xsd:string	Y	The descriptive name of the server.
replyTime	xsd:dateTime	Y	The date and time associated with the creation of the response.
additionalProperties	ews_common_xsd:AdditionalProperty	Y	Additional properties.The supported properties are: pap.stage, pap.note, pap.time

ResponseResult structure

Defines the result element in the `PushResponse` structure, which is used in the response returned from a `sendPushMessage` operation.

Element Name	Element type	Optional	Description
code	xsd:string	N	A code representing the outcome when sending the push message. Generated by the network node. Possible status codes are listed in Table 6-1.
description	xsd:string	N	Textual description.

Table 6-1 Outcome status codes

Status 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.
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.

Status code	Description
3011	Replacement not supported.
4000	Service failure.
4001	Service unavailable.

ReplaceMethod enumeration

Defines the values for the `replacePushId` parameter in the `sendPushMessage` operation. This parameter is used to replace an existing message based on a given push ID. This parameter is ignored if it is set to NULL.

Enumeration value	Description
all	Indicates that this push message MUST be treated as a new push submission for all recipients, no matter if a previously submitted push message with pushId equal to the replacePushId in this push message can be found or not.
pending-only	Indicates that this push message should be treated as a new push submission only for those recipients who have a pending push message that is possible to cancel. In this case, if no push message with pushId equal to the replacePushId in this push message can be found, the server responds with status code PUSH_ID_NOT_FOUND in the responseResult. Status code CANCELLATION_NOT_POSSIBLE may be returned in the responseResult if no message can be cancelled. Status code CANCELLATION_NOT_POSSIBLE may also be returned in a subsequent resultNotification to indicate a non-cancellable message for an individual recipient.

MessageState enumeration

Defines the values for the `messageState` parameter in a `resultMessageNotification`.

Enumeration value	Description
rejected	Message was not accepted by the network.
pending	Message is being processed.
delivered	Message successfully delivered to the network.
undeliverable	The message could not be delivered.
expired	The message reached the maximum allowed age or could not be delivered by the time specified when the message was sent. **Note:** Some network elements allows for defining policies on maximum age of messages.
aborted	The end-user terminal aborted the message.
timeout	The delivery process timed out.
cancelled	The message was cancelled.
unknown	The state of the message is unknown.

Web Service interface description

The following describes the interfaces and operations that are available in the Extended Web Services WAP Push Web Service.

Interface: PushMessage

Operations to send, or to manipulate previously sent, WAP Push messages.

Operation: sendPushMessage

Sends a WAP Push message. The message Content Entity (the payload) is provided as a SOAP attachment in MIME format. The Content Entity is a MIME body part containing the content to be sent to the wireless device. The content type is not defined, and can be any type as long as it can be described by MIME. The Content Entity is included only in the push submission and is not included in any other operation request or response.

Input message: sendPushMessage

Part name	Part type	Optional	Description
pushId	xsd:string	N	Provided by the application. Serves as a message ID. The application is responsible for its uniqueness, for example, by using an address within its control (for example a URL) combined with an identifier for the push message as the value for `pushId`. Supported types are PLMN and USER. For example: "www.wapforum.org/123" or "123@wapforum.org"
destinationAddresses	xsd:string [1..unbounded]	N	An array of end-user terminal addresses. The addresses should be formatted according to the Push Proxy Gateway Service Specification (WAP-249-PPGService-20010713-a). Example addresses: • WAPPUSH=+155519990730 TYPE=PLMN@ppg.carrier.com • WAPPUSH=john.doe%40wapforum.org TYPE=USER@ppg.carrier.com
resultNotificationEndpoint	xsd:anyURI	Y	Specifies the URL the application uses to return result notifications. The presence of this parameter indicates that a notification is requested. If the application does not want a notification, this parameter must be set to NULL.

Part name	Part type	Optional	Description
replacePushId	xsd:string	Y	The `pushId` of the still pending message to replace. The presence of this parameter indicates that the client is requesting that this message replace one previously submitted, but still pending push message. The following rules apply: • Setting the `replacePushId` parameter to NULL indicates that it is a new message. It does not replace any previously submitted message. • The initial pending (pending delivery to the end-user terminal) message is cancelled, if possible, for *all* recipients of the message. This means that it is possible to replace a message for only a subset of the recipients of the original message. • Message replacement will occur only for the recipients for whom the pending message can be cancelled.
replaceMethod	push_message_xsd:ReplaceMethod	N	Defines how to replace a previously sent message. Used in conjunction with the `replacePushId` parameter described above. Ignored if `replacePushId` is NULL.
deliverBeforeTimestamp	xsd:dateTime	Y	Defines the date and time by which the content must be delivered to the end-user terminal. The message is not delivered to the end-user terminal after this time and date. If the network node does not support this parameter, the message is rejected.
deliverAfterTimestamp	xsd:dateTime	Y	Specifies the date and time after which the content should be delivered to the wireless device. The message is delivered to the end-user terminal after this time and date. If the network node does not support this parameter, the message is be rejected.

Part name	Part type	Optional	Description
sourceReference	xsd:string	Y	A textual name of the content provider.
progressNotesRequested	xsd:boolean	Y	This parameter informs the network node if the client wants to receive progress notes. TRUE means that progress notes are requested. Progress notes are delivered via the `PushMessageNotification` interface. If not set, progress notes are not sent.
serviceCode	xsd:string	N	Used for charging purposes.
requesterID	xsd:string	N	The application ID as given by the operator.
additionalProperties	ews_common_xsd:AdditionalProperty [0...unbounded]	Y	Additional properties, defined as name/value pairs, can be sent using this parameter. The supported properties are: pap.priority, pap.delivery-method, pap.network, pap.network-required, pap.bearer, pap.bearer-required.

Output message: sendPushMessageResponse

Part name	Part type	Optional	Description
result	push_message_xsd:PushResponse	N	The response that Oracle Communications Services Gatekeeper returns for `sendPushMessage` operation

Referenced faults

Table 6-2 exceptions and error codes

Exception	Error code	Reason/Action
SVC0001	WNG-000001	Internal problem in Oracle Communications Services Gatekeeper. Contact Oracle Communications Services Gatekeeper administrator.
SVC0001	WNG-000002	Internal problem in Oracle Communications Services Gatekeeper. Contact Oracle Communications Services Gatekeeper administrator.
SVC0001	PUSHMSG-000002	Failed to create push message.
SVC0001	PUSHMSG-000003	Unable to retrieve configuration.
SVC0001	PUSHMSG-000001	Failed to submit push message to PPG.
SVC0001	PLG-000004	General plug-in routing error

Interface: PushMessageNotification

Operation: resultNotificationMessage

Input message: resultNotificationMessage

Part name	Part type	Optional	Description
pushId	xsd:string	N	Defined by the application in the corresponding `sendPushMessage` operation. Used to match the notification to the message.
address	xsd:string	N	The address of the end-user terminal.

Part name	Part type	Optional	Description
messageState	push_message_xsd:MessageState	N	State of the message.
code	xsd:string	N	Final status of the message.
description	xsd:string	Y	Textual description of the notification. Supplied by the network. May or may not be present, depending on the network node used.
senderAddress	xsd:string	Y	Address of the network node. May or may not be present, depending on the network node used.
senderName	xsd:string	Y	Name of the network node. May or may not be present, depending on the network node used.
receivedTime	xsd:dateTime	Y	Time and date when the message was received at the network node.
eventTime	xsd:dateTime	Y	Time and date when the message reached the end-user terminal.
additionalProperties	ews_common_xsd:AdditionalProperty	Y	Additional properties can be sent using this parameter in the form of name/value pairs. The supported properties are: • pap.priority • pap.delivery-method • pap.network • pap.network-required • pap.bearer • pap.bearer-required Which properties are sent, if any, is dependent on the network node.

Output message: resultNotificationMessageResponse

Part name	Part type	Optional	Description
none			

Referenced faults

Table 6-3 exceptions and error codes

Exception	Error code	Reason/Action
SVC0001	PUSHMSG-000004	Failed to send result notification to the application.

WSDLs

The document/literal WSDL representation of the PushMessage interface can be retrieved from the Web Services endpoint.

The document/literal WSDL representation of the PushMessageNotification interface can be downloaded from

```
http://<host>:<port>/ews/push_message/wsdls/ews_common_types.xsd
http://<host>:<port>/ews/push_message/wsdls/ews_push_message_notification_
interface.wsdl
http://<host>:<port>/ews/push_message/wsdls/ews_push_message_notification_
service.wsdl
http://<host>:<port>/ews/push_message/wsdls/ews_push_message_types.xsd
```

Where host and port are depending on the Oracle Communications Services Gatekeeper deployment.

Sample Send WAP Push Message

Listing 6-1 Example Send WAP Push Message

```
// Add handlers for MIME types needed for WAP MIME-types

MailcapCommandMap mc = (MailcapCommandMap) CommandMap.getDefaultCommandMap();

mc.addMailcap("text/vnd.wap.si;;x-java-content-handler=com.sun.mail.handlers.t
ext_xml");

CommandMap.setDefaultCommandMap(mc);

// Create a MIME-message where with the actual content of the WAP Push message.

InternetHeaders headers = new InternetHeaders();

headers.addHeader("Content-type", "text/plain; charset=UTF-8");

headers.addHeader("Content-Id", "mytext");

byte[] bytes = "Test message".getBytes();

MimeBodyPart mimeMessage = new MimeBodyPart(headers, bytes);


// Create PushMessage with only the manadatory parameters


// SendPushMessage is provided in the stubs generated from the WSDL.

SendPushMessage sendPushMessage = new SendPushMessage();

String [] destinationAddresses = {"wappush=461/type=user@ppg.o.se"};

sendPushMessage.setDestinationAddresses(destinationAddresses);

// Create “unique” pushId, using a combination of timestamp and domain.

sendPushMessage.setPushId(System.currentTimeMillis() + "@wlng.bea.com");

// ReplaceMethod is provided by the stubs generated from the WSDL.

sendPushMessage.setReplaceMethod(ReplaceMethod.pendingOnly);

// Defined by the operator/service provider contractual agreement

sendPushMessage.setServiceCode(”Service Code xxx”);
```

```
// Defined by the operator/service provider contractual agreement
sendPushMessage.setRequesterID(”Requester ID xxx”);
// Endpoint to send notifications to. Implemented on the application side.
String notificationEndpoint =
"http://localhost:80/services/PushMessageNotification";
sendPushMessage.setResultNotificationEndpoint(new URI(notificationEndpoint));


// Send the WAP Push message
PushMessageService pushMessageService = null;
// Define the endpoint of the WAP Push Web Service
String endpoint = "http://localhost:8001/ews/push_message/PushMessage?WSDL";
try {
  // Instantiate an representation of the Web Service from the generated stubs.
  pushMessageService = new PushMessageService_Impl(endpoint);
} catch (ServiceException e) {
  e.printStackTrace();
  throw e;
}
PushMessage pushMessage = null;
try {
  // Get the Web Service interface to operate on.
  pushMessage = pushMessageService.getPushMessage();
} catch (ServiceException e) {
    e.printStackTrace();
    throw e;
  }
SendPushMessageResponse sendPushMessageResponse = null;
try {
```

```
  // Send the WAP Push message.
  sendPushMessageResponse = pushMessage.sendPushMessage(sendPushMessage);
} catch (RemoteException e) {
  e.printStackTrace();
  throw e;
} catch (PolicyException e) {
  e.printStackTrace();
  throw e;
} catch (com.bea.wlcp.wlng.schema.ews.common.ServiceException e) {
  e.printStackTrace();
  throw e;
}
// Assign the pushId provided in the in the response to a local variable.
String pushId = sendPushMessageResponse.getPushId();
```

CHAPTER 7

Extended Web Services Subscriber Profile

The Extended Web Services Subscriber Profile Web Service allows an application to get subscriber-specific data from data sources within the network operator's domain.

Examples of data sources are subscriber databases containing information about terminal types in use, preferred language, and currency types. This information can be used by applications in order to control rendering options for rich media, charging information, and the language to be used in voice and text interaction with the end-user.

The interface is built around a model where the data can be retrieved in two different ways:

- Individual attributes, identified using a *path*.
- A collection of attributes.

The attributes are keyed on a *subscriber ID* that uniquely identifies the subscriber for whom the attributes are valid or by an *address* that uniquely identifies the terminal for which the attributes are valid. An attribute is identified by a *path name*, which corresponds to a specific property. The following is an example of a path name:

```
serviceName/accessControlId/accessControlId
```

The syntax for the path is similar to relative file system paths in UNIX.

A collection of attributes is specified in a subscriber profile filter for the application or the service provider. Only allowed attributes, as specified in the filter, are returned.

The returned attributes are returned in the form of name-value pairs, or property tuples, where the name is expressed as a path name with a associated property value.

The interface is based on a proposal for a Parlay X Subscriber Profile Web Service interface.

Namespaces

The SubscriberProfile interface and service use the namespaces:

- http://www.bea.com/wlcp/wlng/wsdl/ews/subscriber_profile/interface
- http://www.bea.com/wlcp/wlng/wsdl/ews/subscriber_profile/service

The data types are defined in the namespace:

- http://www.bea.com/wlcp/wlng/schema/ews/subscriber_profile

In addition, Extended Web Services Subscriber Profile uses definitions common for all Extended Web Services interfaces:

- The datatypes are defined in the namespace:
  - http://www.bea.com/wlcp/wlng/schema/ews/common
- The faults are defined in the namespace:
  - http://www.bea.com/wlcp/wlng/wsdl/ews/common/faults

Endpoint

The endpoint for the PushMessage interface is:
`http://<host>:<port>/ews/subscriber_profile/SubscriberProfile`

Where host and port depend on the Oracle Communications Services Gatekeeper deployment.

Address schemes

Table 7-1 Supported address schemes

Address scheme	Valid for Communication service
tel	Extended Web Services Subscriber profile for LDAPv3
id	Extended Web Services Subscriber profile for LDAPv3
imsi	Extended Web Services Subscriber profile for LDAPv3
ipv4	Extended Web Services Subscriber profile for LDAPv3

XML Schema data type definition

The following data structures are used in the Extended Web Services Subscriber Profile Web Service.

PropertyTuple structure

Defines the response that the Oracle Communications Services Gatekeeper returns from Operation: get and Operation: getProfile.

Element Name	Element type	Optional	Description
pathName	xsd:string	N	The key of the name-value pair. Expressed as a relative UNIX path. Example: `serviceName/accessControlId/accessControlId`
propertyValue	xsd:string	N	The value associated with the key.

Web Service interface description

The following describes the interfaces and operations that are available in the Extended Web Subscriber Profile Web Service.

Interface: SubscriberProfile

Operations to obtain specific subscriber profile attributes and operations to obtain a set of profile properties grouped together in a profile.

Operation: get

Gets specific subscriber profile attributes. The requested attributes are identified by the pathNames parameter, and the possible values are restricted by the configured capabilities of the underlying data source. The allowed path name values are also restricted individually per service provider and application in the SLA.

Input message: get

Part name	Part type	Optional	Description
address	xsd:anyURI	N	Identity to get profile attributes for.
pathNames	xsd:string [1..unbounded]	N	Requested subscriber properties. Expressed as a relative UNIX path. Example: `serviceName/accessControlId/accessControlId`

Output message: getResponse

Part name	Part type	Optional	Description
properties	PropertyTuple [1..unbounded]	N	All retrieved subscription property name and value pairs which are requested by application and allowed by the usage policies as specified in a filter. See PropertyTuple structure.

Referenced faults

Table 7-2 Exceptions and error codes

Exception	Error code	Reason/Action
ESVC0001	WNG000002	Internal problem in Oracle Communications Services Gatekeeper. Contact Oracle Communications Services Gatekeeper administrator.
ESVC0001	SP000001	Internal problem in Oracle Communications Services Gatekeeper. The LDAP connection is not working. There might be a configuration error for with the underlying LDAP server or a network error. Contact Oracle Communications Services Gatekeeper administrator.
ESVC0001	SP000002	Internal problem in Oracle Communications Services Gatekeeper. LDAP operation failed. Contact Oracle Communications Services Gatekeeper administrator.
ESVC0001	SP000003	Internal problem in Oracle Communications Services Gatekeeper. Contact Oracle Communications Services Gatekeeper administrator.
ESVC0001	SP000004	Internal problem in Oracle Communications Services Gatekeeper. Contact Oracle Communications Services Gatekeeper administrator.

Operation: getProfile

Gets a set of profile properties grouped together in a profile identified by a certain profile ID.

Input message: getProfile

Part name	Part type	Optional	Description
subscriberID	xsd:string	N	Identity to get profile attributes for.
profileID	xsd:string	N	Identity of the profile to get.

Profile ID is ignored when connecting the to the network using the LDAPv3 network protocol plug-in. The collection of attributes that identifies the profile are provisioned as filters.

Output message: getResponse

Part name	Part type	Optional	Description
properties	PropertyTuple [1..unbounded]	N	All retrieved subscription property name and value pairs which are requested by application and allowed by the usage policies as specified in a filter. See PropertyTuple structure.

Referenced faults

Table 7-3 exceptions and error codes

Exception	Error code	Reason/Action
SVC0001	WNG-000002	Internal problem in Oracle Communications Services Gatekeeper. Contact Oracle Communications Services Gatekeeper administrator.
SVC0001	SP-000001	Internal problem in Oracle Communications Services Gatekeeper. The LDAP connection is not working. There might be a configuration error for with the underlying LDAP server or a network error. Contact Oracle Communications Services Gatekeeper administrator.
SVC0001	SP-000002	Internal problem in Oracle Communications Services Gatekeeper. LDAP operation failed. Contact Oracle Communications Services Gatekeeper administrator.
SVC0001	SP-000003	Internal problem in Oracle Communications Services Gatekeeper. Contact Oracle Communications Services Gatekeeper administrator.
SVC0001	SP-000004	Internal problem in Oracle Communications Services Gatekeeper. Contact Oracle Communications Services Gatekeeper administrator.
SVC0001	PLG-000004	General plug-in routing error

WSDLs

The document/literal WSDL representation of the SubscriberProfile interface can be retrieved from the Web Services endpoint, see Endpoint, or:

```
http://<host>:<port>/ews/subscriber_profile/SubscriberProfile?WSDL
```

```
http://<host>:<port>/ews/subscriber_profile/SubscriberProfile?WSDL/ews_subscriber_profile_interface.wsdl
```

```
http://<host>:<port>/ews/subscriber_profile/SubscriberProfile?WSDL/ews_common_types.xsd
```

Where host and port depend on the Oracle Communications Services Gatekeeper deployment.

CHAPTER 8

Extended Web Services Common

The Extended Web Services set of Web Services share common definitions described in this section:

- Namespace
- XML Schema datatype definition
  - AdditionalProperty structure
  - ChargingInformation structure
  - SimpleReference structure
- Fault definitions
  - ServiceException
  - PolicyException

Namespace

The namespace for the common data types is:

- http://www.bea.com/wlcp/wlng/schema/ews/common

The namespace for the common faults is:

- http://www.bea.com/wlcp/wlng/wsdl/ews/common/faults

XML Schema datatype definition

AdditionalProperty structure

Defines a name-value pair.

Element Name	Element type	Optional	Description
name	xsd:string	Y	Name part.
value	xsd:string	Y	Value part.

ChargingInformation structure

For services that include charging as an inline message part, the charging information is provided in this data structure.

Element Name	Element type	Optional	Description
description	xsd:string	N	Description text to be use for information and billing text.
currency	xsd:string	Y	Currency identifier as defined in ISO 4217.
amount	xsd:decimal	Y	Amount to be charged.
code	xsd:string	Y	Charging code, referencing a contract under which the charge is applied.

SimpleReference structure

For those services that require a reference to a Web Service, the information required to create the endpoint information is contained in this type.

Element Name	Element type	Optional	Description
endpoint	xsd:anyURI	N	Description text to be use for information and billing text.
interfaceName	xsd:string	Y	Name of interface.
correlator	xsd:decimal	Y	Correlation information.

Fault definitions

ServiceException

Faults related to the operation of the service, not including policy related faults, result in the return of a ServiceException message.

Element Name	Element type	Optional	Description
messageId	xsd:string	N	Message identifier, with prefix SVC.
text	xsd:string	N	Message text, with replacement variables marked with %#
variables	xsd:string [0...unbounded]	Y	Variables to substitute into text string.

Service Exception are related to the operation of the service itself. The following exceptions are general:

- SVC0001: Service error.
- SVC0002: Invalid input value
- SVC0003: Invalid input value with list of valid values
- SVC0004: No valid addresses

- SVC0005: Duplicate correlator
- SVC0006: Invalid group
- SVC0007: Invalid charging information
- SVC0008: Overlapping Criteria

PolicyException

Faults related to policies associated with the service, result in the return of a PolicyException message.

Element Name	Element type	Optional	Description
messageId	xsd:string	N	Message identifier, with prefix POL.
text	xsd:string	N	Message text, with replacement variables marked with %#
variables	xsd:string [0...unbounded]	Y	Variables to substitute into text string.

PolicyExceptions are thrown when a policy has been violated, including violations of a service level agreements. The following general PolicyExceptions are defined:

- POL0001: Policy error
- POL0002: Privacy error
- POL0003: Too many addresses specified
- POL0004: Unlimited notifications not supported
- POL0005: Too many notifications requested
- POL0006: Groups not allowed
- POL0007: Nested groups not allowed
- POL0008: Charging not supported

- POL0009: Invalid frequency requested

CHAPTER 9

Parlay X 2.1 Interfaces

This chapter describes the supported Parlay X 2.1 interfaces and contains information that is specific for Oracle Communications Services Gatekeeper, and not found in the specifications. For detailed descriptions of the interfaces, methods and parameters, refer to the specifications.

See http://portal.etsi.org/docbox/TISPAN/Open/OSA/ParlayX21.html for links to the specifications.

- Parlay X 2.1 Part 2: Third Party Call
  - Interface: ThirdPartyCall
  - Error Codes
- Parlay X 2.1 Part 3: Call Notification
  - Interface: CallDirection
  - Interface: CallNotification
  - Interface: CallNotificationManager
  - Interface: CallDirectionManager
  - Error Codes
- Parlay X 2.1 Part 4: Short messaging
  - Interface: SendSms
  - Interface: SmsNotification
  - Interface: ReceiveSms

Parlay X 2.1 Part 2: Third Party Call

This set of interfaces is compliant to ETSI ES 202 391-2 V1.2.1 (2006-12) Open Service Access (OSA); Parlay X Web Services; Part 2: Third Party Call (Parlay X 2).

Interface: ThirdPartyCall

The endpoint for this interface is:
http://<host>:<port>/parlayx21/third_party_call/ThirdPartyCall

Where values for host and port depend on the Oracle Communications Services Gatekeeper deployment.

MakeCall

Sets up a call between two parties.

GetCallInformation

Displays information about a call.

EndCall

Ends a call.

CancelCall

Cancels a call setup procedure.

Error Codes

See General error codes.

Parlay X 2.1 Part 3: Call Notification

This set of interfaces is compliant to ETSI ES 202 391-3 V1.2.1 (2006-12) Open Service Access (OSA); Parlay X Web Services; Part 3: Call Notification (Parlay X 2).

Interface: CallDirection

This interface is implemented by an application, and the consumer of this interface is Oracle Communications Services Gatekeeper. The WSDL that defines the interface can be downloaded from:

```
http://<host>:<port>/parlayx21/call_notification/wsdls/parlayx_call_direction_interface_2_2.wsdl
```

```
http://<host>:<port>/parlayx21/call_notification/wsdls/parlayx_call_direction_service_2_2.wsdl
```

```
http://<host>:<port>/parlayx21/call_notification/wsdls/parlayx_call_notifi
cation_types_2_2.xsd
```

Where values for host and port depend on the Oracle Communications Services Gatekeeper deployment.

HandleBusy

Oracle Communications Services Gatekeeper calls this method, which is implemented by an application, when the called party is busy.

HandleNotReachable

Oracle Communications Services Gatekeeper calls this method, which is implemented by an application, when the called party is not reachable.

HandleNoAnswer

Oracle Communications Services Gatekeeper calls this method, which is implemented by an application, when the called party does not answer.

HandleCalledNumber

Oracle Communications Services Gatekeeper calls this method, which is implemented by an application, prior to call setup.

Interface: CallNotification

This interface is implemented by an application, and the consumer of this interface is Oracle Communications Services Gatekeeper. The WSDL that defines the interface can be downloaded from:

```
http://<host>:<port>/parlayx21/call_notification/wsdls/parlayx_call_notifi
cation_interface_2_2.wsdl
```

```
http://<host>:<port>/parlayx21/call_notification/wsdls/parlayx_call_notifi
cation_service_2_2.wsdl
```

```
http://<host>:<port>/parlayx21/call_notification/wsdls/parlayx_call_notifi
cation_types_2_2.xsd
```

NotifyBusy

Oracle Communications Services Gatekeeper calls this method, which is implemented by an application, when the called party is busy.

NotifyNotReachable

Oracle Communications Services Gatekeeper calls this method, which is implemented by an application, when the called party is not reachable.

NotifyNoAnswer

Oracle Communications Services Gatekeeper calls this method, which is implemented by an application, when the called party does not answer.

NotifyCalledNumber

Oracle Communications Services Gatekeeper calls this method, which is implemented by an application, prior to call setup.

Interface: CallNotificationManager

The endpoint for this interface is:
http://<host>:<port>/parlayx21/call_notification/CallNotificationManager

Where values for host and port depend on the Oracle Communications Services Gatekeeper deployment.

StartCallNotification

Starts a subscription for call notifications.

StopCallNotification

Stops a subscription for call notifications.

Interface: CallDirectionManager

The endpoint for this interface is:
http://<host>:<port>/parlayx21/call_notification/CallDirectionManager

Where values for host and port depend on the Oracle Communications Services Gatekeeper deployment.

StartCallDirectionNotification

Starts a subscription for call direction notifications.

StopCallDirectionNotification

Stops a subscription for call direction notifications.

Error Codes

See General error codes.

Parlay X 2.1 Part 4: Short messaging

This set of interfaces is compliant to ETSI ES 202 391-4 V1.2.1 (2006-12) Open Service Access (OSA); Parlay X Web Services; Part 4: Short Messaging (Parlay X 2).

Interface: SendSms

The endpoint for this interface is: http://<host>:<port>/parlayx21/sms/SendSms

Where values for host and port depend on the Oracle Communications Services Gatekeeper deployment.

If a backwards-compatible communication service is used:

- The parameter senderAddress is either of the format tel:<mailbox ID>\<mailbox password>\<Sender name> or just <sender name> depending on how the application was provisioned in Oracle Communications Services Gatekeeper.
- The priority parameter is not supported.

SendSms

Sends an SMS to one or more destinations.

SendSmsLogo

Sends an SMS Logo to one or more destinations.

Logos in SmartMessaging and EMS are supported. The image is not scaled.

Logos in the following raw image formats are supported:

- bmp
- gif
- jpg

- png

The logos are in pure black and white (gray scale not supported). Animated images are not supported. Scaling is not supported.

If the logo shall be converted to SmartMessaging format, the image cannot be larger than 72x14 pixels.

If the logo shall be is sent in EMS format, the following rules apply:

- If the image is 16x16 pixels, the logo is sent as an EMS small picture.
- If the image is 32x32 pixels, the logo is sent as an EMS large picture.
- If the image is of any other size, the logo is sent as an EMS variable picture.
- Images up to 1024 pixels are supported.

SendSmsRingtone

Sends an SMS Ringtone to one or more destinations.

Ringtones can be in any of these formats:

- RTX
- SmartMessaging
- EMS (iMelody)

GetSmsDeliveryStatus

Gets the delivery status of a previously sent SMS.

It is possible to query delivery status of an SMS only if a callback reference was not defined when the SMS was sent. If a callback reference was defined, NotifySmsDeliveryReceipt is invoked by Oracle Communications Services Gatekeeper and the delivery status is not stored. If the delivery status is stored in Oracle Communications Services Gatekeeper, it is stored for a configurable period of time.

Interface: SmsNotification

This interface is implemented by an application, and the consumer of this interface is Oracle Communications Services Gatekeeper. The WSDL that defines the interface can be downloaded from:

```
http://<host>:<port>/parlayx21/sms/wsdls/parlayx_sms_notification_interface_2_2.wsdl
http://<host>:<port>/parlayx21/sms/wsdls/parlayx_sms_notification_service_2_2.wsdl
http://<host>:<port>/parlayx21/sms/wsdls/parlayx_sms_types_2_2.xsd
```

Where values for host and port depend on the Oracle Communications Services Gatekeeper deployment.

NotifySmsReception

Sends an SMS that is received by Oracle Communications Services Gatekeeper to an application given that the SMS fulfills a set of criteria. The criteria is either defined by the application itself, using startSmsNotification or defined using a provisioning step in Oracle Communications Services Gatekeeper.

Shortcode translation, if appropriate, is applied.

NotifySmsDeliveryReceipt

Sends a delivery receipt that a previously sent SMS has been received by its destination. The delivery receipt is propagated to the application given that the application provided a callback reference when sending the SMS.

Interface: ReceiveSms

The endpoint for this interface is: http://<host>:<port>/parlayx21/sms/ReceiveSms

Where values for host and port depend on the Oracle Communications Services Gatekeeper deployment.

GetReceivedSms

Gets messages that have been received by Oracle Communications Services Gatekeeper. The SMSs are fetched using a registrationIdentifier used when the notification was registered using a provisioning step in Oracle Communications Services Gatekeeper.

Interface: SmsNotificationManager

The endpoint for this interface is: http://<host>:<port>/parlayx21/sms/SmsNotificationManager

Where values for host and port depend on the Oracle Communications Services Gatekeeper deployment.

StartSmsNotification

Initiates notifications to the application for a given service activation number and criteria.

Note: Service activation number may be provisioned to cater for a range of numbers via short code translations.

Note: The equivalent to this operation may have been performed as an off-line provisioning step by the Oracle Communications Services Gatekeeper administrator.

StopSmsNotification

Ends a previously started notification.

Error Codes

See General error codes.

Parlay X 2.1 Part 5: Multimedia messaging

This set of interfaces is compliant to ETSI ES 202 391-5 V1.2.1 (2006-12) Open Service Access (OSA); Parlay X Web Services; Part 5: Multimedia Messaging (Parlay X 2).

Interface: SendMessage

The endpoint for this interface is:
http://<host>:<port>/parlayx21/multimedia_messaging/SendMessage

Where values for host and port depend on the Oracle Communications Services Gatekeeper deployment.

SendMessage

Sends a multimedia message. The content of the message is sent as a SOAP attachment. E-mail is not supported.

Table 9-1 exceptions and error codes

Exception	Error code	Reason/Action
SVC0001	WNG-000002	Internal problem in Oracle Communications Services Gatekeeper. Contact Oracle Communications Services Gatekeeper administrator.
SVC0001	MMS-000001	Internal problem in Oracle Communications Services Gatekeeper. Contact Oracle Communications Services Gatekeeper administrator.
SVC0001	MMS-000002	Internal problem in Oracle Communications Services Gatekeeper. Contact Oracle Communications Services Gatekeeper administrator.
SVC0001	MMS-000003	Address is utilizing an unsupported address type.
SVC0001	MMS-000005	Message could not be delivered to MMSC.

GetMessageDeliveryStatus

Gets the delivery status of a previously sent MMS.

It is possible to query delivery status of an MMS only if a callback reference was not defined when the message was sent. If a callback reference was defined, NotifyMessageDeliveryReceipt is invoked by Oracle Communications Services Gatekeeper and the delivery status is not stored. If the delivery status is stored in Oracle Communications Services Gatekeeper, it is stored for a configurable period of time.

Note: Oracle Communications Services Gatekeeper may be configured not to store delivery status for MMS.

Table 9-2 exceptions and error codes

Exception	Error code	Reason/Action
SVC0001	WNG-000002	Internal problem in Oracle Communications Services Gatekeeper. Contact Oracle Communications Services Gatekeeper administrator.
SVC0002	RequestIdentifier	Message is not found.

Interface: ReceiveMessage

The endpoint for this interface is:
http://<host>:<port>/parlayx21/multimedia_messaging/ReceiveMessage

Where the values for host and port depend on the Oracle Communications Services Gatekeeper deployment.

GetReceivedMessages

Polls Oracle Communications Services Gatekeeper for received messages.

The registrationIdentifier is required. Received message are stored in Oracle Communications Services Gatekeeper only for a configurable period of time.

Table 9-3 exceptions and error codes

Exception	Error code	Reason/Action
SVC0001	WNG-000002	Internal problem in Oracle Communications Services Gatekeeper. Contact Oracle Communications Services Gatekeeper administrator.
SVC0002	MMS-000002	Internal problem in Oracle Communications Services Gatekeeper. Contact Oracle Communications Services Gatekeeper administrator.

GetMessageURIs

Not supported.

GetMessage

Gets a specific message that was received by Oracle Communications Services Gatekeeper and belongs to the application.

Table 9-4 exceptions and error codes

Exception	Error code	Reason/Action
SVC0001	WNG-000002	Internal problem in Oracle Communications Services Gatekeeper. Contact Oracle Communications Services Gatekeeper administrator.
SVC0001	MMS-000004	Correlator does not exist, no notification corresponds to the correlator.

Interface: MessageNotification

This interface is implemented by an application, and the consumer of this interface is Oracle Communications Services Gatekeeper. The WSDL that defines the interface can be downloaded from:

```
http://<host>:<port>/parlayx21/multimedia_messaging/wsdls/parlayx_mm_notification_interface_2_4.wsdl
```

```
http://<host>:<port>/parlayx21/multimedia_messaging/wsdls/parlayx_mm_notification_service_2_4.wsdl
```

```
http://<host>:<port>/parlayx21/multimedia_messaging/wsdls/parlayx_mm_types_2_4.xsd
```

Where the values for host and port depend on the Oracle Communications Services Gatekeeper deployment.

NotifyMessageReception

Sends a notification to an application that an MMS destined for the application is received by Oracle Communications Services Gatekeeper.

NotifyMessageDeliveryReceipt

Sends a notification to an application that a previously sent MMS has been delivered to its destination.

Note: Oracle Communications Services Gatekeeper can be configured to support delivery notifications or not.

Interface: MessageNotificationManager

The endpoint for this interface is:
http://<host>:<port>/parlayx21/multimedia_messaging/MessageNotificationManager

Where the values for host and port depend on the Oracle Communications Services Gatekeeper deployment.

StartMessageNotification

Initiates notifications to the application for a given service activation number and criteria.

Note: Service activation number may be provisioned to cater for a range of numbers via short code translations.

Note: The equivalent to this operation may have been performed as an off-line provisioning step by the Oracle Communications Services Gatekeeper administrator.

Table 9-5 exceptions and error codes

Exception	Error code	Reason/Action
SVC0001	WNG-000002	Internal problem in Oracle Communications Services Gatekeeper. Contact Oracle Communications Services Gatekeeper administrator.

StopMessageNotification

Ends a previously started notification.

Table 9-6 exceptions and error codes

Exception	Error code	Reason/Action
SVC0001	WNG-000002	Internal problem in Oracle Communications Services Gatekeeper. Contact Oracle Communications Services Gatekeeper administrator.
SVC0002	Correlator	Correlator does not exist, no notification corresponds to the correlator.

Error Codes

See General error codes.

Parlay X 2.1 Part 9: Terminal location

This set of interfaces is compliant to ETSI ES 202 391-9 V1.2.1 (2006-12), Open Service Access (OSA); Parlay X Web Services; Part 9: Terminal Location (Parlay X 2).

Interface: TerminalLocation

The endpoint for this interface is:
http://<host>:<port>/parlayx21/terminal_location/TerminalLocation

Where values for host and port depend on the Oracle Communications Services Gatekeeper deployment.

GetLocation

Gets the location for a terminal.

Table 9-7 exceptions and error codes

Exception	Error code	Reason/Action
SVC0001	TL-000007	Communication problems between Oracle Communications Services Gatekeeper and the network node. Contact Oracle Communications Services Gatekeeper administrator.
SVC0001	TL-000010	Communication problems between Oracle Communications Services Gatekeeper and the network node, unable to interpret response. Contact Oracle Communications Services Gatekeeper administrator.
SVC0001	TL-000009	No location data received from network.
SVC0001	TL-000011	Unknown error received from network.
SVC0002		Invalid parameter provided in request.
SVC0200		Accuracy of location is not within acceptable limit.
POL0001		General policy error.
POL0002		Privacy error.
POL0230		Requested accuracy not supported.

GetTerminalDistance

Gets the distance from a certain point to the location of a terminal.

Table 9-8 exceptions and error codes

Exception	Error code	Reason/Action
SVC0001	TL-000007	Communication problems between Oracle Communications Services Gatekeeper and network node. Contact Oracle Communications Services Gatekeeper administrator.
SVC0001	TL-000010	Communication problems between Oracle Communications Services Gatekeeper and network node, unable to interpret response. Contact Oracle Communications Services Gatekeeper administrator.
SVC0001	TL-000009	No location data received from network.
SVC0001	TL-000011	Unknown error received from network.
SVC0002		Invalid parameter provided in request.
SVC0200		Accuracy of location is not within acceptable limit.
POL0001		General policy error.
POL0002		Privacy error.
POL0230		Requested accuracy not supported.

GetLocationForGroup

Gets the location for one or more terminals.

Table 9-9 exceptions and error codes

Exception	Error code	Reason/Action
SVC0001	TL-000007	Communication problems between Oracle Communications Services Gatekeeper and network node. Contact Oracle Communications Services Gatekeeper administrator.
SVC0001	TL-000010	Communication problems between Oracle Communications Services Gatekeeper and network node, unable to interpret response. Contact Oracle Communications Services Gatekeeper administrator.
SVC0001	TL-000009	No location data received from network.
SVC0001	TL-000011	Unknown error received from network.
SVC0002		Invalid parameter provided in request.
SVC0004		No valid addresses.
SVC0200		Accuracy of location is not within acceptable limit.
POL0001		General policy error.
POL0002		Privacy error.
POL0230		Requested accuracy not supported.

Interface: TerminalLocationNotificationManager

The endpoint for this interface is:
http://<host>:<port>/parlayx21/terminal_location/TerminalLocationNotificationManager

Where values for host and port depend on the Oracle Communications Services Gatekeeper deployment.

StartGeographicalNotification

Initiates location notifications to the application when one or more terminal changes their location according to a criteria.

Table 9-10 exceptions and error codes

Exception	Error code	Reason/Action
SVC0001	TL-000003	Unable to start geographical notification due to a network error. Contact Oracle Communications Services Gatekeeper administrator.
SVC0001	TL-000004	Unable to start geographical notification due to an internal error. Contact Oracle Communications Services Gatekeeper administrator.
SVC0002		Invalid parameter provided in request.
SVC0004		No valid addresses.
SVC0005		Correlator already exists.
POL0001		General policy error.

StartPeriodicNotification

Initiates location notifications to the application on a periodic basis.

Table 9-11 exceptions and error codes

Exception	Error code	Reason/Action
SVC0001	TL-000005	Unable to start periodic notification due to a network error. Contact Oracle Communications Services Gatekeeper administrator.
SVC0001	TL-000006	Unable to start periodic notification due to an internal error. Contact Oracle Communications Services Gatekeeper administrator.

Table 9-11 exceptions and error codes

Exception	Error code	Reason/Action
SVC0002		Invalid parameter provided in request.
SVC0004		No valid addresses.
SVC0005		Correlator already exists.
POL0001		General policy error.

EndNotification

Ends a previously started notification.

Table 9-12 exceptions and error codes

Exception	Error code	Reason/Action
SVC0001	TL-000001	Unable to start geographical notification due to a network error. Contact Oracle Communications Services Gatekeeper administrator.
SVC0001	TL-000002	Unable to start geographical notification due to an internal error. Contact Oracle Communications Services Gatekeeper administrator.
SVC0002		Invalid parameter provided in request.
POL0001		General policy error.

Interface: TerminalLocationNotification

This interface is implemented by an application, and the consumer of this interface is Oracle Communications Services Gatekeeper. The WSDL that defines the interface can be downloaded from:

```
http://<host>:<port>/parlayx21/terminal_location/wsdls/parlayx_terminal_location_notification_interface_2_2.wsdl
```

```
http://<host>:<port>/parlayx21/terminal_location/wsdls/parlayx_terminal_location_notification_service_2_2.wsdl
```

```
http://<host>:<port>/parlayx21/terminal_location/wsdls/parlayx_terminal_location_types_2_2.xsdl
```

Where values for host and port depend on the Oracle Communications Services Gatekeeper deployment.

LocationNotification

Notifies an application about a change of location for a terminal.

LocationError

Notifies an application that the subscription for location notifications was cancelled by Oracle Communications Services Gatekeeper.

LocationEnd

Notifies an application that no more location notifications are being sent to the application.

Error Codes

See General error codes.

Parlay X 2.1 Part 14: Presence

This set of interfaces is compliant to ETSI ES 202 391-14 V1.2.1 (2006-12), Open Service Access (OSA); Parlay X Web Services; Part 14: Presence (Parlay X 2).

Interface: PresenceConsumer

The endpoint for this interface is: http://<host>:<port>//parlayx21/presence/PresenceConsumer

Where values for host and port depend on the Oracle Communications Services Gatekeeper deployment.

subscribePresence

Requests a subscription for presence information about a presentity.

For the parameter presentity, only SIP URI can be used. Group-URI is not supported

getUserPresence

Gets presence information about a presentity.

For the parameter presentity, only SIP URI can be used. Group-URI is not supported

startPresenceNotification

Initiates presence notifications to the application when one or more presence attributes changes for a presentity.

For the parameter presentity, only SIP URI can be used. Group-URI is not supported

The parameter frequency is not supported. The application is notified when an update of presence information is received from the network.

endPresenceNotification

Ends a previously started notification.

Interface: PresenceNotification

This interface is implemented by an application, and the consumer of this interface is Oracle Communications Services Gatekeeper. The WSDL that defines the interface can be downloaded from:

```
http://<host>:<port>/parlayx21/presence/wsdls/parlayx_presence_notification_interface_2_3.wsdl
```

```
http://<host>:<port>/parlayx21/presence/wsdls/parlayx_presence_notification_service_2_3.wsdl
```

```
http://<host>:<port>/parlayx21/presence/wsdls/parlayx_presence_types_2_3.xsd
```

Where values for host and port depend on the Oracle Communications Services Gatekeeper deployment.

statusChanged

Notifies an application about a change of presence attributes for a presentity.

statusEnd

Notifies an application that no more notifications will be sent to the application.

notifySubscription

Notifies an application that the presentity has handled the request for presence information.

subscriptionEnded

Notifies an application that the subscription for presence information has ended.

Interface: PresenceSupplier

The endpoint for this interface is: http://<host>:<port>//parlayx21/presence/PresenceSupplier where values for host and port depend on the Oracle Communications Services Gatekeeper deployment.

Notes: The Presence Supplier interface requires that a Presence Server be available in the underlying network. Oracle Communications Services Gatekeeper interacts with this server to provide this functionality to the application.

Because the Parlay X 2.1 interface does not provide parameters to identify the presentity URI but this information is required to interact with the network, Oracle Communications Services Gatekeeper by default maps the URI to the client's Application Instance ID. The application can override this value using the Parameter Tunneling function. The <param> to use is key = "wlng.presence.parlay21.presentity.uri" and value = $the_desired_URI.

publish

Publishes presence data about a presentity

Note: The Presence Server in the network must support http://tools.ietf.org/html/draft-ietf-simple-partial-publish-07#page-4 for this functionality to work.

getOpenSubscriptions

Gets a list of new watchers who wish to subscribe to this presentity's data.

Note: The Presence Server in the network must support http://tools.ietf.org/html/draft-ietf-simple-partial-pidf-format-10 and http://tools.ietf.org/html/rfc3857 (http://tools.ietf.org/html/draft-ietf-simple-winfo-package-05) for this functionality to work.

updateSubscriptionAuthorization

Adds new watchers and updates the permissions of existing watchers of this presentity's data. This is the usual follow-up to the getOpenSubscriptions operation.

Note: This operation requires the presence of both a Presence Server and a Data Manipulation Server in the underlying network.

An SVC0220 service exception (NoSubscriptionRequest) is thrown if the presentity attempts to confirm the subscription of a watcher who has not asked to subscribe to the presentity's data.

getMyWatchers

Returns an array of the watchers that are subscribed to the presentity's data.

Note: The Presence Server in the network must support http://tools.ietf.org/html/draft-ietf-simple-partial-pidf-format-10 and http://tools.ietf.org/html/rfc3857 (http://tools.ietf.org/html/draft-ietf-simple-winfo-package-05) for this functionality to work.

getSubscribedAttributes

Not supported.

blockSubscription

Blocks the flow of presence information to a subscribed watcher by cancelling the subscription. The watcher is notified using subscriptionEnded.

An SVC0221 service exception (Not a watcher) is thrown if the URI in the field watcher is not a watcher of the presentity.

Error Codes

See General error codes.

About notifications

When an application has started a notification, the notification is persisted. This means that if an application has started a notification and destroys the session, the notification is still registered and matching notifications are sent to the application when it connects to Oracle Communications Services Gatekeeper.

General Exceptions

This section describes the exception handling for the Parlay X 2.1 interfaces.

These exception types are defined:

- Service Exceptions
- Policy Exceptions

Service Exceptions are related to the operation of the service itself. The following exceptions are general:

- SVC0001: Service error.
- SVC0002: Invalid input value
- SVC0003: Invalid input value with list of valid values
- SVC0004: No valid addresses
- SVC0005: Duplicate correlator
- SVC0006: Invalid group
- SVC0007: Invalid charging information
- SVC0008: Overlapping Criteria

PolicyExceptions are thrown when a policy has been violated, including violations of a service level agreements. The following general PolicyExceptions are defined:

- POL0001: Policy error
- POL0002: Privacy error
- POL0003: Too many addresses specified
- POL0004: Unlimited notifications not supported
- POL0005: Too many notifications requested
- POL0006: Groups not allowed
- POL0007: Nested groups not allowed
- POL0008: Charging not supported
- POL0009: Invalid frequency requested

Within the exception, an error code is defined. The error code details why the exception was thrown. See General error codes

General error codes

The following are general error codes for SVC0001: Service error:

- Null sessionID (loginTicket) expired.
- WNG-000000 No error.
- WNG-000001 Unknown error.
- WNG-000002 Storage service error.
- PLG-000001 Could not find remote ejb home in access tier.
- PLG-000002 Could not create the ejb.
- PLG-000003 Could not access callback ejb.
- PLG-000004 Matching plug-in cannot be found because, for example, route has not been set up
- SIP-000001 Could not find remote ejb home.
- SIP-000002 Could not create the ejb.
- SIP-000003 Could not access remote ejb.
- SIP-000004 Could not create SIP session.
- SIP-000005 Failed to send SIP message.
- SIP-000006 Internal SIP stack error.
- CN-000001 Two requests for call direction overlap with each other
- CN-000002 Internal error when accessing the subscription storage
- CN-000003 Could not find the call-back plug-in
- CN-000004 The call direction routing address is not valid
- PRESENCE-000001 Failed to use the default 'duration' for a notification.
- PRESENCE-000002 Failed to use the default value for count for a notification.
- PRESENCE-000003 The application has no SIP-URI mapping configured.
- PRESENCE-000004 Internal error. Failed to put data in WorkContext.

- PLC-000001 Internal policy engine error
- OSA-000001 Invalid network state.
- OSA-000002 Invalid interface type.
- OSA-000003 Invalid event type.
- OSA-000004 Unsupported address plan.
- OSA-000005 Communication failure between OSA Gateway and Oracle Communications Services Gatekeeper.
- TL-000001 Unable to end notification, network error
- TL-000002 Unable to end notification, internal error
- TL- 000003 Unable to start geographical notification, network error.
- TL-000004 Unable to start geographical notification, internal error
- TL-000005 Unable to start periodic notification, network error.
- TL-000006 Unable to start periodic notification, internal error.
- TL-000007 Unable to get location, network error.
- TL-000008 Unable to get location, internal error.
- TL-000009 Unable to get location, no data found.
- TL-000010 Failed to parse response, internal error.
- TL-000011 Failed to get location information, error returned from MLP server.
- MMS-000001 Unable to perform action. Network error.
- MMS-000002 Unable to retrieve configuration, internal error.
- MMS-000003 The used address type is not supported.
- MMS-000004 An error occurred when an attachment was put into the request context.
- MMS-000005 The MM7 Relay server responded with an error code.
- SMS-000001 Unable to perform action. Network error.
- SMS-000002 Unable to retrieve configuration, internal error.

- SMS-000003 The used address type is not supported
- SMS-000004 Unable to encode message segments
- SMS-000005 GSM message format incorrect
- TPC-000001 Unable to retrieve configuration, internal error

Code examples

Below are some code examples that illustrate how to use the Parlay X 2.1 interfaces.

Example: sendSMS

Below is an example of sending an SMS.

Listing 9-1 SendSMS example

```
org.csapi.schema.parlayx.sms.send.v2_2.local.SendSms request =

new org.csapi.schema.parlayx.sms.send.v2_2.local.SendSms();

SimpleReference sr = new SimpleReference();

sr.setEndpoint(new
URI("http://localhost:8111/SmsNotificationService/services/SmsNotification?WSD
L"));

sr.setCorrelator("cor188");

sr.setInterfaceName("InterfaceName");

ChargingInformation charging = new ChargingInformation();

charging.setAmount(new BigDecimal("1.1"));

charging.setCode("qwerty");

charging.setCurrency("USD");

charging.setDescription("some charging info");

sendInf.setCharging(charging);

URI[] uri = new URI[1];

uri[0] = new URI("1234");
```

```
request.setAddresses(uri);

request.setCharging(charging);

request.setReceiptRequest(sr);

request.setMessage("we are testing sms!");

request.setSenderName(“6001”);

org.csapi.schema.parlayx.sms.send.v2_2.local.SendSmsResponse response =

smport.sendSms(request);

String sendresult = response.getResult();

System.out.println("result: " + sendresult);
```

Example: startSmsNotification

Below is an example of using startSmsNotification.

Listing 9-2 startSmsNotification example

```
org.csapi.schema.parlayx.sms.notification_manager.v2_3.local.StartSmsNotificat
ion parameters =

new
org.csapi.schema.parlayx.sms.notification_manager.v2_3.local.StartSmsNotificat
ion();

parameters.setCriteria("hello");

SimpleReference sr = new SimpleReference();

sr.setEndpoint(new
URI("http://localhost:8111/SmsNotificationService/services/SmsNotification?WSD
L"));

sr.setCorrelator("cor189");

sr.setInterfaceName("interfaceName");

parameters.setReference(sr);

URI uri = new URI("tel:6001");
```

```
parameters.setSmsServiceActivationNumber(uri);
port.startSmsNotification(parameters);
```

Example: getReceivedSms

Below is an example of polling for SMSes using getReceivedSms.

Listing 9-3 getReceivedSms example

```
org.csapi.schema.parlayx.sms.receive.v2_2.local.GetReceivedSms parameters =
new org.csapi.schema.parlayx.sms.receive.v2_2.local.GetReceivedSms();
parameters.setRegistrationIdentifier("1");
org.csapi.schema.parlayx.sms.receive.v2_2.local.GetReceivedSmsResponse
response =
port.getReceivedSms(parameters);
org.csapi.schema.parlayx.sms.v2_2.SmsMessage[] msgs =
response.getResult();
if(msgs != null) {
  for(org.csapi.schema.parlayx.sms.v2_2.SmsMessage msg : msgs) {
    System.out.println(msg.getMessage());
  }
}
```

Example: sendMessage

Below is an example of sending an MMS.

Listing 9-4 sendMessage example

```
org.csapi.schema.parlayx.multimedia_messaging.send.v2_4.local.SendMessage
request =

new
org.csapi.schema.parlayx.multimedia_messaging.send.v2_4.local.SendMessage();

ChargingInformation charging = new ChargingInformation();

charging.setAmount(new BigDecimal("1.1"));

charging.setCode("qwerty");

charging.setCurrency("USD");

charging.setDescription("some charging info");

sendInf.setCharging(charging);

SimpleReference sr = new SimpleReference();

if(getProperty("notification_mt").equalsIgnoreCase("true")) {

  sr.setEndpoint(new
URI(getProperty(ClientConstants.NOTIFICATION_LISTENER_URL)));

  sr.setCorrelator(getProperty("correlator"));

  sr.setInterfaceName(getProperty("interfacename"));

}

URI[] uri = new URI[1];

uri[0] = new URI("1234");

request.setAddresses(uri);

request.setCharging(charging);

request.setPriority(MessagePriority.fromString("Default"));

request.setReceiptRequest(sr);

request.setSenderAddress("6001");

request.setSubject("subject");

org.csapi.schema.parlayx.multimedia_messaging.send.v2_4.local.SendMessageRespo
nse response =
```

```
smport.sendMessage(request);

String sendresult = response.getResult();

System.out.println("sendresult: " + sendresult);
```

Example: getReceivedMessages and getMessage

Below is an example of polling for a received MMS.

Listing 9-5 getReceivedMessages and getMessage example

```
org.csapi.schema.parlayx.multimedia_messaging.receive.v2_4.local.GetReceivedMe
ssages parameters =

new
org.csapi.schema.parlayx.multimedia_messaging.receive.v2_4.local.GetReceivedMe
ssages();

parameters.setPriority(org.csapi.schema.parlayx.multimedia_messaging.v2_4.Mess
agePriority.fromString("Default"));

parameters.setRegistrationIdentifier("2");

org.csapi.schema.parlayx.multimedia_messaging.receive.v2_4.local.GetReceivedMe
ssagesResponse result =

port.getReceivedMessages(parameters);

org.csapi.schema.parlayx.multimedia_messaging.v2_4.MessageReference[] refs =

result.getResult();

if(refs != null) {

  for(org.csapi.schema.parlayx.multimedia_messaging.v2_4.MessageReference ref :
refs) {

    String id = ref.getMessageIdentifier();

    org.csapi.schema.parlayx.multimedia_messaging.receive.v2_4.local.GetMessag
e p2 =

    new
org.csapi.schema.parlayx.multimedia_messaging.receive.v2_4.local.GetMessage();

    p2.setMessageRefIdentifier(id);
```

```
    port.getMessage(p2);
  }
}
```

Example: getLocation

Below is an example of getting the location of a terminal.

Listing 9-6 getLocation example

```
org.csapi.schema.parlayx.terminal_location.v2_2.local.GetLocation parameters =
new org.csapi.schema.parlayx.terminal_location.v2_2.local.GetLocation();
parameters.setAcceptableAccuracy(5);
parameters.setAddress(new URI("1234"));
parameters.setRequestedAccuracy(5);
TimeMetric maximumAge = new TimeMetric();
maximumAge.setMetric(TimeMetrics.fromString("Hour"));
maximumAge.setUnits(10);
parameters.setMaximumAge(maximumAge);
TimeMetric responseTime = new TimeMetric();
responseTime.setMetric(TimeMetrics.fromString("Hour"));
responseTime.setUnits(1);
parameters.setResponseTime(responseTime);
DelayTolerance tolerance = DelayTolerance.fromString("NoDelay");
parameters.setTolerance(tolerance);
org.csapi.schema.parlayx.terminal_location.v2_2.local.GetLocationResponse
response =
port.getLocation(parameters);
org.csapi.schema.parlayx.terminal_location.v2_2.LocationInfo result =
```

```
response.getResult();
System.out.println("accuracy : " + result.getAccuracy());
System.out.println("altitude : " + result.getAltitude().floatValue());
System.out.println("latitude : " + result.getLatitude());
System.out.println("longitude : " + result.getLongitude());
System.out.println("timestamp : " + result.getTimestamp());
```

CHAPTER 10

Parlay X 3.0 Interfaces

This chapter describes the supported Parlay X 3.0 interfaces and contains information that is specific for Oracle Communications Services Gatekeeper, and not found in the specifications. For detailed descriptions of the interfaces, methods and parameters, refer to the specifications.

See http://portal.etsi.org/docbox/TISPAN/Open/OSA/ParlayX30.html for links to the specifications.

- Interaction between Audio Call, Third Party Call, and Call Notification
- Parlay X 3.0 Part 2: Third Party Call
  - Interface: ThirdPartyCall
- Parlay X 3.0 Part 3: Call Notification
  - Interface: CallDirection
  - Interface: CallNotification
  - Interface: CallNotificationManager
  - Interface: CallDirectionManager
- Parlay X 3.0 Part 6: Payment
  - Interface: AmountCharging
  - Interface: VolumeCharging
  - Interface: ReserveAmountCharging
  - Interface: ReserveVolumeCharging

- Parlay X 3.0 Part 11: Audio call
  - Interface: PlayMedia

Interaction between Audio Call, Third Party Call, and Call Notification

The Parlay X 3.0 Part 2: Third Party Call, Parlay X 3.0 Part 3: Call Notification, and Parlay X 3.0 Part 11: Audio call interfaces, when used together with the Parlay-type plug-ins, are designed to interact so they can be used to implement, for example, a conference call application using a combination of the services exposed by these interfaces:

- Call setup
- Call redirection and transfer
- Playing of announcements
- Collection of input from participants in the call using DTMF

A call can have several participants.The call as a whole is represented by a callSessionIdentifier, and each participant is identified by its URI (the phone number, with scheme tel:) can be added to the call.

Note: When the call is initiated from an application, the callSessionIdentifier is returned from Oracle Communications Services Gatekeeper when the call session is established. When the call is initiated from the network, the callSessionIdentifier is provided by Oracle Communications Services Gatekeeper in the requests that report the event.

Application-initiated call setup, tear-down, and transfer is managed using the Parlay X 3.0 Part 2: Third Party Call interfaces.

Subscribing for notification on network-initiated calls, and taking action depending on the events is done using the Parlay X 3.0 Part 3: Call Notification interfaces.

Playing of announcements and initiating collection of input from call participants is done using the Parlay X 3.0 Part 11: Audio call interfaces. Results from the collection of input are reported using Parlay X 3.0 Part 3: Call Notification.

Parlay X 3.0 Part 11: Audio call interfaces must be used together with either Parlay X 3.0 Part 2: Third Party Call or Parlay X 3.0 Part 3: Call Notification since Audio Call does not have any operations to establish a call.

Parlay X 3.0 Part 2: Third Party Call

This set of interfaces is compliant to ETSI ES 202 504-2 v0.0.5 (2007-06) Open Service Access (OSA); Parlay X Web Services; Part 2: Third Party Call (Parlay X 3).

Interface: ThirdPartyCall

The endpoint for this interface is:
http://<host>:<port>/parlayx30/third_party_call/ThirdPartyCall

Where values for host and port depend on the Oracle Communications Services Gatekeeper deployment.

makeCallSession

Sets up a call between two parties.

The parameter:

- `mediaInfo` must be set to NULL.
- `changeMediaNotAllowed` must be set to false.

Table 10-1 exceptions and error codes

Exception	Error code	Explanation
SVC0001	TPC-000001	Error reported from the telecom network.
SVC0001	TPC-000002	Error reported from the telecom network.
SVC0001	OSA-000001	Error reported from the telecom network.
SVC0001	OSA-000002	Error reported from the telecom network.
SVC0001	OSA-000003	Error reported from the telecom network.
SVC0001	OSA-000004	Error reported from the telecom network.
SVC0001	OSA-000006	Error reported from the telecom network.
SVC0001	OSA-000007	Error reported from the telecom network.
SVC0001	OSA-000008	Error reported from the telecom network.

Table 10-1 exceptions and error codes

Exception	Error code	Explanation
SVC0001	OS-A000009	Error reported from the telecom network.
SVC0001	OS-A000010	Error reported from the telecom network.
SVC0001	OS-A000011	Error reported from the telecom network.
SVC0001	OSA-000012	Error reported from the telecom network.
SVC0001	OS-A000013	Error reported from the telecom network.
SVC0001	OSA-000014	Error reported from the telecom network.
SVC0001	OSA-000015	Error reported from the telecom network.
SVC0001	WNG-000002	Failed to store information in Oracle Communications Services Gatekeeper.
SVC0001	PLG-000004	General plug-in routing error.
SVC0002	n/a	Invalid input parameter.
POL0001		Faults related to policies associated with the service, including service level agreements.
POL0008		Charging not supported.
POL0011		Media type not supported.

addCallParticipant

Adds a participant to a an existing call session. The call session may have been established using makeCallSession or any of the methods in Interface: CallDirection and Interface: CallNotification.

parameter `mediaInfo` must be set to NULL.

Table 10-2 exceptions and error codes

Exception	Error code	Explanation
SVC0001	TPC-000002	Error reported from the telecom network.
SVC0001	OSA-000001	Error reported from the telecom network.
SVC0001	OSA-000002	Error reported from the telecom network.
SVC0001	OSA-000003	Error reported from the telecom network.
SVC0001	OSA-000004	Error reported from the telecom network.
SVC0001	OSA-000006	Error reported from the telecom network.
SVC0001	OSA-000007	Error reported from the telecom network.
SVC0001	OSA-000008	Error reported from the telecom network.
SVC0001	OSA-000009	Error reported from the telecom network.
SVC0001	OSA-000010	Error reported from the telecom network.
SVC0001	OSA-000011	Error reported from the telecom network.
SVC0001	OSA-000012	Error reported from the telecom network.
SVC0001	OSA-000014	Error reported from the telecom network.
SVC0001	OSA-000015	Error reported from the telecom network.
SVC0001	WNG-000002	Failed to store information in Oracle Communications Services Gatekeeper.
SVC0001	PLG-000004	General plug-in routing error.
SVC0002	n/a	Call session identifier is null. Or Error reported from the telecom network.
SVC0261	n/a	The call is already terminated.
POL0001		The application is not the owner of the call. Error reported form the network.

Table 10-2 exceptions and error codes

Exception	Error code	Explanation
POL0011	n/a	Media type not supported.
POL0240	n/a	Too many participants.

transferCallParticipant

Transfers a participant from one call session to another call session.

Table 10-3 exceptions and error codes

Exception	Error code	Explanation
SVC0001	TPC-000002	No call leg identifier reported form network. Error reported from the telecom network.
SVC0001	TPC-000003	The destination call session reference or call leg reference is null, internal error.
SVC0001	TPC-000007	The participant does not belong to this call.
SVC0001	OSA-000001	Error reported from the telecom network.
SVC0001	OSA-000002	Error reported from the telecom network.
SVC0001	OSA-000003	Error reported from the telecom network.
SVC0001	OSA-000004	Error reported from the telecom network.
SVC0001	OSA-000006	Error reported from the telecom network.
SVC0001	OSA-000007	Error reported from the telecom network.
SVC0001	OSA-000008	Error reported from the telecom network.
SVC0001	OSA-000009	Error reported from the telecom network.
SVC0001	OSA-000010	Error reported from the telecom network.
SVC0001	OSA-000011	Error reported from the telecom network.

Table 10-3 exceptions and error codes

Exception	Error code	Explanation
SVC0001	OSA-000012	Error reported from the telecom network.
SVC0001	OSA-000014	Error reported from the telecom network.
SVC0001	OSA-000015	Error reported from the telecom network.
SVC0001	WNG-000002	Failed to store information in Oracle Communications Services Gatekeeper.
SVC0001	PLG-000004	General plug-in routing error.
SVC0002		Error reported from the telecom network. Or Source call session identifier is invalid. Or Destination call session identifier is invalid. Or Participant part is invalid.
SVC0261	n/a	The call is already terminated.
POL0001		TPC100001
POL0240	n/a	Too many participants.

getCallParticipantInformation

Gets information about a certain participant in a call session.

Table 10-4 exceptions and error codes

Exception	Error code	Explanation
SVC0001	TPC-000007	The participant does not belong to this call.
SVC0001	OSA-000001	Error reported from the telecom network.
SVC0001	OSA-000002	Error reported from the telecom network.

Table 10-4 exceptions and error codes

Exception	Error code	Explanation
SVC0001	OSA-000003	Error reported from the telecom network.
SVC0001	OSA-000004	Error reported from the telecom network.
SVC0001	OSA-000006	Error reported from the telecom network.
SVC0001	OSA-000007	Error reported from the telecom network.
SVC0001	OSA-000008	Error reported from the telecom network.
SVC0001	OSA-000009	Error reported from the telecom network.
SVC0001	OSA-000010	Error reported from the telecom network.
SVC0001	OSA-000011	Error reported from the telecom network.
SVC0001	OSA-000012	Error reported from the telecom network.
SVC0001	OSA-000014	Error reported from the telecom network.
SVC0001	OSA-000015	Error reported from the telecom network.
SVC0001	WNG-000002	Failed to store information in Oracle Communications Services Gatekeeper.
SVC0001	PLG-000004	General plug-in routing error.
SVC0002	n/a	Call session identifier is invalid.
SVC0261	n/a	The call is already terminated.
POL0001	TPC-100001	The application is not the owner of the call session.

getCallSessionInformation

Displays information about a call session.

Table 10-5 exceptions and error codes

Exception	Error code	Explanation
SVC0001	TPC-000007	The participant does not belong to this call.
SVC0001	OSA-000001	Error reported from the telecom network.
SVC0001	OSA-000002	Error reported from the telecom network.
SVC0001	OSA-000003	Error reported from the telecom network.
SVC0001	OSA-000004	Error reported from the telecom network.
SVC0001	OSA-000006	Error reported from the telecom network.
SVC0001	OSA-000007	Error reported from the telecom network.
SVC0001	OSA-000008	Error reported from the telecom network.
SVC0001	OSA-000009	Error reported from the telecom network.
SVC0001	OSA-000010	Error reported from the telecom network.
SVC0001	OSA-000011	Error reported from the telecom network.
SVC0001	OSA-000012	Error reported from the telecom network.
SVC0001	OSA-000014	Error reported from the telecom network.
SVC0001	OSA-000015	Error reported from the telecom network.
SVC0001	WNG-000002	Failed to store information in Oracle Communications Services Gatekeeper.
SVC0001	PLG-000004	General plug-in routing error.
SVC0002	n/a	network error. Or Call session identifier is invalid.
SVC0261	n/a	The call is already terminated.
POL0001	TPC-100001	The application is not the owner of the call session.

deleteCallParticipant

Deletes a participant from a call session.

Table 10-6 exceptions and error codes

Exception	Error code	Explanation
SVC0001	TPC-000006	There are no participants in this call.
SVC0001	OSA-000001	Error reported from the telecom network.
SVC0001	OSA-000002	Error reported from the telecom network.
SVC0001	OSA-000003	Error reported from the telecom network.
SVC0001	OSA-000004	Error reported from the telecom network.
SVC0001	OSA-000006	Error reported from the telecom network.
SVC0001	OSA-000007	Error reported from the telecom network.
SVC0001	OSA-000008	Error reported from the telecom network.
SVC0001	OSA-000009	Error reported from the telecom network.
SVC0001	OSA-000010	Error reported from the telecom network.
SVC0001	OSA-000011	Error reported from the telecom network.
SVC0001	OSA-000012	Error reported from the telecom network.
SVC0001	OSA-000014	Error reported from the telecom network.
SVC0001	OSA-000015	Error reported from the telecom network.
SVC0001	WNG-000002	Failed to store information in Oracle Communications Services Gatekeeper.
SVC0001	PLG-000004	General plug-in routing error.

Table 10-6 exceptions and error codes

Exception	Error code	Explanation
SVC0002	n/a	Network error. Or Call session identifier is invalid. Or Call participant identifier is invalid.
SVC0261	n/a	The call is already terminated.
POL0001	TPC-100001	The application is not the owner of the call session.

endCallSession

Ends a call session.

Table 10-7 exceptions and error codes

Exception	Error code	Explanation
SVC0001	TPC-000005	The call reference in Oracle Communications Services Gatekeeper storage is invalid.
SVC0001	TPC-000006	There are no participants in the call session.
SVC0001	OSA-000001	Error reported from the telecom network.
SVC0001	OSA-000002	Error reported from the telecom network.
SVC0001	OSA-000003	Error reported from the telecom network.
SVC0001	OSA-000004	Error reported from the telecom network.
SVC0001	OSA-000006	Error reported from the telecom network.
SVC0001	OSA-000007	Error reported from the telecom network.
SVC0001	OSA-000008	Error reported from the telecom network.
SVC0001	OSA-000009	Error reported from the telecom network.

Table 10-7 exceptions and error codes

Exception	Error code	Explanation
SVC0001	OSA-000010	Error reported from the telecom network.
SVC0001	OSA-000011	Error reported from the telecom network.
SVC0001	OSA-000012	Error reported from the telecom network.
SVC0001	OSA-000014	Error reported from the telecom network.
SVC0001	OSA-000015	Error reported from the telecom network.
SVC0001	WNG-000002	Failed to store information in Oracle Communications Services Gatekeeper.
SVC0001	PLG-000004	General plug-in routing error.
SVC0002	n/a	Error reported from the telecom network. Or Call session identifier is invalid. Or Call participant identifier is invalid.
SVC0261	n/a	The call is already terminated.
POL0001	TPC-100001	The application is not the owner of the call session.

Parlay X 3.0 Part 3: Call Notification

This set of interfaces is compliant to ETSI ES 202 504-3 v0.0.3 (2007-06) Open Service Access (OSA); Parlay X Web Services; Part 3: Call Notification (Parlay X 3).

Interface: CallDirection

This interface is implemented by an application, and the consumer of this interface is Oracle Communications Services Gatekeeper. The WSDL that defines the interface can be downloaded from:

```
http://<host>:<port>/parlayx30/call_notification/wsdls/parlayx_call_direction_service_3_2.wsdl
```

```
http://<host>:<port>/parlayx30/call_notification/wsdls/parlayx_call_direction_interface_3_2.wsdl
http://<host>:<port>/parlayx30/call_notification/wsdls/parlayx_common_types_3_1.xsd
http://<host>:<port>/parlayx30/call_notification/wsdls/parlayx_common_faults_3_0.wsdl
http://<host>:<port>/parlayx30/call_notification/wsdls/parlayx_call_notification_types_3_1.xsd
```

Where values for host and port depend on the Oracle Communications Services Gatekeeper deployment.

HandleBusy

Oracle Communications Services Gatekeeper calls this method, which is implemented by an application, when the called party is busy.

HandleNotReachable

Oracle Communications Services Gatekeeper calls this method, which is implemented by an application, when the called party is not reachable.

HandleNoAnswer

Oracle Communications Services Gatekeeper calls this method, which is implemented by an application, when the called party does not answer.

HandleCalledNumber

Oracle Communications Services Gatekeeper calls this method, which is implemented by an application, prior to call setup.

Interface: CallNotification

This interface is implemented by an application, and the consumer of this interface is Oracle Communications Services Gatekeeper. The WSDL that defines the interface can be downloaded from:

```
http://<host>:<port>/parlayx30/call_notification/wsdls/parlayx_call_notification_interface_3_2.wsdl
http://<host>:<port>/parlayx30/call_notification/wsdls/parlayx_call_notification_service_3_2.wsdl
```

```
http://<host>:<port>/parlayx30/call_notification/wsdls/parlayx_common_types_3_1.xsd
http://<host>:<port>/parlayx30/call_notification/wsdls/parlayx_common_faults_3_0.wsdl
http://<host>:<port>/parlayx30/call_notification/wsdls/parlayx_call_notification_types_3_1.xsd
```

Where values for host and port depend on the Oracle Communications Services Gatekeeper deployment.

notifyBusy

Oracle Communications Services Gatekeeper calls this method, which is implemented by an application, when the called party is busy.

notifyNotReachable

Oracle Communications Services Gatekeeper calls this method, which is implemented by an application, when the called party is not reachable.

notifyNoAnswer

Oracle Communications Services Gatekeeper calls this method, which is implemented by an application, when the called party does not answer.

notifyCalledNumber

Oracle Communications Services Gatekeeper calls this method, which is implemented by an application, prior to call setup.

notifyAnswer

Oracle Communications Services Gatekeeper calls this method, which is implemented by an application, when the called party answered.

notifyPlayAndCollectEvent

Oracle Communications Services Gatekeeper calls this method, which is implemented by an application, to provide the result of a media interaction of type play and collect information.

notifyPlayAndRecordEvent

Oracle Communications Services Gatekeeper calls this method, which is implemented by an application, to provide the result of a media interaction of type play and record information.

Interface: CallNotificationManager

The endpoint for this interface is:

http://<host>:<port>/parlayx30/call_notification/CallNotificationManager

Where values for host and port depend on the Oracle Communications Services Gatekeeper deployment.

startCallNotification

Starts a subscription for call notifications.

Table 10-8 exceptions and error codes

Exception	Error code	Explanation
SVC0001	WNG-000002	Failed to store information in Oracle Communications Services Gatekeeper. Contact support.
SVC0001	OSA-000002	P_INVALID_INTERFACE_TYPE thrown by OSA Gateway. check the interface name
SVC0001	OSA-000003	P_INVALID_EVENT_TYPE thrown by OSA Gateway Check the event type
SVC0001	OSA-000006	TpCommonExceptions thrown by OSA Gateway. Exception type is P_RESOURCE_UNAVAILABLE (13). Check OSA Gateway status.
SVC0001	OSA-000007	TpCommonExceptions thrown by OSA Gateway. Exception type is P_TASK_REFUSED(14). Check OSA Gateway status.

Table 10-8 exceptions and error codes

Exception	Error code	Explanation
SVC0001	OSA-000008	TpCommonExceptions thrown by OSA Gateway. Exception type is P_TASK_CANCELLED(15). Check invocation parameters of createNotification()
SVC0001	OSA-000009	TpCommonExceptions thrown by OSA Gateway. Exception type is P_NO_CALLBACK_ADDRESS_SET (17). Check OSA Gateway.
SVC0001	OSA-000010	TpCommonExceptions thrown by OSA Gateway. Exception type is P_METHOD_NOT_SUPPORTED (22). Check OSA Gateway.
SVC0001	OSA-000011	TpCommonExceptions thrown by OSA Gateway. Exception type is P_INVALID_STATE (744). Check OSA Gateway.
SVC0001	OSA-000015	P_INVALID_CRITERIA thrown by OSA Gateway. Check the criteria
SVC0001	PLG-000004	General plug-in routing error.
SVC0002	reference	Parameter reference is null.
SVC0002	correlator	Parameter correlator is null.
SVC0002	endPoint	Parameter endPoint is null or empty String.
SVC0002	addresses	Parameter reference is null.
SVC0005	<correlator value>, reference	Correlator %1 specified in message part %2 is a duplicate.
POL0001	Service contract not found	No Service Level Agreement found for the service provider or application associated with the request.

startPlayAndCollectNotification

Starts a subscription for notifications on media interactions of type play and collect.

Table 10-9 exceptions and error codes

Exception	Error code	Explanation
SVC0001	WNG-000002	Failed to store information in Oracle Communications Services Gatekeeper. Contact support.
SVC0001	CN-000001	Parlay call session does not exist.
SVC0001	CN-000002	Parlay call session has terminated.
SVC0001	PLG-000004	General plug-in routing error.
SVC0002	reference	Parameter reference is null.
SVC0002	correlator	Parameter correlator is null.
SVC0002	endPoint	Parameter endPoint is null or empty String.
SVC0002	callSessionIdentifier	Parameter callSessionIdentifier is null.
SVC0005	<correlator value>, reference	Correlator %1 specified in message part %2 is a duplicate.
SVC0005	callSessionId:<value>	startPlayAndCollectNotification has been invoked earlier on the same callSessionIdentifier or startCallDirection has been invoked earlier with an address that is identical to the address represented by the call session.
POL0001	Service contract not found	No Service Level Agreement found for the service provider or application associated with the request.

startPlayAndRecordNotification

Not supported.

stopCallNotification

Stops a subscription for call notifications.

Table 10-10 exceptions and error codes

Exception	Error code	Explanation
SVC0001	WNG-000002	Failed to store information in Oracle Communications Services Gatekeeper. Contact support.
SVC0001	CN-000003	The requester is not the owner of the notification, that is, did not start the notification.
SVC0001	CN-000004	The parameter correlator does not exist.
SVC0001	OSA-000006	TpCommonExceptions thrown by OSA GW. Exception type is P_RESOURCE_UNAVAILABLE (13).Check OSA Gateway status.
SVC0001	OSA-000007	TpCommonExceptions thrown by OSA Gateway. Exception type is P_TASK_REFUSED (14). Check OSA Gateway status.
SVC0001	OSA-000008	TpCommonExceptions thrown by OSA Gateway. Exception type is P_TASK_CANCELLED (15). Check OSA Gateway status.
SVC0001	OSA-000009	TpCommonExceptions thrown by OSA Gateway. Exception type is P_NO_CALLBACK_ADDRESS_SET (17). Check invocation parameters of destroyNotification()
SVC0001	OSA-000010	TpCommonExceptions thrown by OSA Gateway. Exception type is P_METHOD_NOT_SUPPORTED (22). Check OSA Gateway.
SVC0001	OSA-000011	TpCommonExceptions thrown by OSA Gateway. Exception type is P_INVALID_STATE (744). Check OSA Gateway.
SVC0001	correlator	The parameter correlator does not exist.
SVC0001	PLG-000004	General plug-in routing error.
POL0001	n/a	No Service Level Agreement found for the service provider or application associated with the request.

stopMediaInteractionNotification

Stops a subscription for notifications on media interactions.

Table 10-11 exceptions and error codes

Exception	Error code	Explanation
SVC0001	WNG-000002	Failed to store information in Oracle Communications Services Gatekeeper. Contact support.
SVC0001	CN-000004	The parameter correlator does not exist.
SVC0001	CN-000003	The requester is not the owner of the notification, that is, did not start the notification.
SVC0001	correlator	The parameter correlator does not exist.
SVC0001	PLG-000004	General plug-in routing error.
POL0001	Service contract not found	No Service Level Agreement found for the service provider or application associated with the request.

Interface: CallDirectionManager

The endpoint for this interface is:

http://<host>:<port>/parlayx30/call_notification/CallDirectionManager

Where values for host and port depend on the Oracle Communications Services Gatekeeper deployment.

StartCallDirectionNotification

Starts a subscription for call direction notifications.

Table 10-12 exceptions and error codes

Exception	Error code	Explanation
SVC0001	WNG-000002	Failed to store information in Oracle Communications Services Gatekeeper. Contact support.
SVC0001	OSA-000002	P_INVALID_INTERFACE_TYPE thrown by OSA Gateway. check the interface name
SVC0001	OSA-000003	P_INVALID_EVENT_TYPE thrown by OSA Gateway Check the event type
SVC0001	OSA-000015	P_INVALID_CRITERIA thrown by OSA Gateway. Check the criteria
SVC0001	OSA-000006	TpCommonExceptions thrown by OSA GW. Exception type is P_RESOURCE_UNAVAILABLE (13).Check OSA Gateway status.
SVC0001	OSA-000007	TpCommonExceptions thrown by OSA Gateway. Exception type is P_TASK_REFUSED(14). Check OSA Gateway status.
SVC0001	OSA-000008	TpCommonExceptions thrown by OSA Gateway. Exception type is P_TASK_CANCELLED (15). Check OSA Gateway status.
SVC0001	OSA-000009	TpCommonExceptions thrown by OSA Gateway. Exception type is P_NO_CALLBACK_ADDRESS_SET (17). Check invocation parameters of createNotification()
SVC0001	OSA-000010	TpCommonExceptions thrown by OSA Gateway. Exception type is P_METHOD_NOT_SUPPORTED (22). Check OSA Gateway.

Table 10-12 exceptions and error codes

Exception	Error code	Explanation
SVC0001	OSA-000011	TpCommonExceptions thrown by OSA Gateway. Exception type is P_INVALID_STATE (744). Check OSA Gateway.
SVC0001	PLG-000004	General plug-in routing error.
SVC0002	reference	Parameter reference is null.
SVC0002	correlator	Parameter correlator is null.
SVC0002	endPoint	Parameter endPoint is null or empty String.
SVC0002	addresses	Parameter reference is null.
SVC0005	<correlator value>, reference	Correlator %1 specified in message part %2 is a duplicate.
POL0001	Service contract not found	No Service Level Agreement found for the service provider or application associated with the request.

StopCallDirectionNotification

Stops a subscription for call direction notifications.

Table 10-13 exceptions and error codes

Exception	Error code	Explanation
SVC0001	WNG-000002	Failed to store information in Oracle Communications Services Gatekeeper. Contact support.
SVC0001	CN-000003	The requester is not the owner of the notification, that is, did not start the notification.
SVC0001	CN-000004	The parameter correlator does not exist.
SVC0001	OSA-000006	TpCommonExceptions thrown by OSA GW. Exception type is P_RESOURCE_UNAVAILABLE (13).Check OSA Gateway status.

Table 10-13 exceptions and error codes

Exception	Error code	Explanation
SVC0001	OSA-000007	TpCommonExceptions thrown by OSA Gateway. Exception type is P_TASK_REFUSED (14). Check OSA Gateway status.
SVC0001	OSA-000008	TpCommonExceptions thrown by OSA Gateway. Exception type is P_TASK_CANCELLED (15). Check OSA Gateway status.
SVC0001	OSA-000009	TpCommonExceptions thrown by OSA Gateway. Exception type is P_NO_CALLBACK_ADDRESS_SET (17). Check invocation parameters of destroyNotification()
SVC0001	OSA-000010	TpCommonExceptions thrown by OSA Gateway. Exception type is P_METHOD_NOT_SUPPORTED (22). Check OSA Gateway.
SVC0001	OSA-000011	TpCommonExceptions thrown by OSA Gateway. Exception type is P_INVALID_STATE (744). Check OSA Gateway.
SVC0001	correlator	The parameter correlator does not exist.
SVC0001	PLG-000004	General plug-in routing error.
POL0001	n/a	No Service Level Agreement found for the service provider or application associated with the request.

Parlay X 3.0 Part 6: Payment

This set of interfaces is compliant to Draft ETSI ES 202 504-6 v0.0.4 (2007-06), Open Service Access (OSA); Parlay X Web Services; Part 6: Payment (Parlay X 3)

Interface: AmountCharging

The endpoint for this interface is: http://<host>:<port>/parlayx30/payment/AmountCharging

Where values for host and port depend on the Oracle Communications Services Gatekeeper deployment.

chargeAmount

Charges the account indicated by the end user identifier.

Table 10-14 Exceptions and Error Codes

Exception	Error code	Explanation
Protocol Errors		
SVC0001	PAYMENT000002	A protocol related error returned from the Diameter server. Make sure the server is running and reachable. Check the log files for more information.
Transient Failures		
SVC0001	PAYMENT000003	A transient error returned from the Diameter server, such as authentication failure. Make sure the server is reachable and has adequate storage space. Check the log files for more information.
SVC0001	PLG-000004	General plug-in routing error.
Permanent Failures		
SVC0001	PAYMENT000004	A permanent failure returned from the Diameter server, such as invalid data in the AVP. Check the log files for more information.
SVC0270	PAYMENT000004	A permanent failure returned from the Diameter server, such as the user could not be authorized. Check the log files for more information.

refundAmount

Refunds the account indicated by the end user identifier.

Table 10-15 Exceptions and Error Codes

Exception	Error code	Explanation
Protocol Errors		
SVC0001	PAYMENT000002	A protocol related error returned from the Diameter server. Make sure the server is running and reachable. Check the log files for more information.
Transient Failures		
SVC0001	PAYMENT000003	A transient error returned from the Diameter server, such as authentication failure. Make sure the server is reachable and has adequate storage space. Check the log files for more information.
SVC0001	PLG-000004	General plug-in routing error.
Permanent Failures		
SVC0001	PAYMENT000004	A permanent failure returned from the Diameter server, such as invalid data in the AVP. Check the log files for more information.
SVC0270	PAYMENT000004	A permanent failure returned from the Diameter server, such as the user could not be authorized. Check the log files for more information.

chargeSplitAmount

Charges multiple end user accounts concurrently.

Table 10-16 Exceptions and Error Codes

Exception	Error code	Explanation
Protocol Errors		
SVC0001	PAYMENT000002	A protocol related error returned from the Diameter server. Make sure the server is running and reachable. Check the log files for more information.
Transient Failures		
SVC0001	PAYMENT000003	A transient error returned from the Diameter server, such as authentication failure. Make sure the server is reachable and has adequate storage space. Check the log files for more information.
SVC0001	PLG-000004	General plug-in routing error.
Permanent Failures		
SVC0001	PAYMENT000004	A permanent failure returned from the Diameter server, such as invalid data in the AVP. Check the log files for more information.
SVC0270	PAYMENT000004	A permanent failure returned from the Diameter server, such as the user could not be authorized. Check the log files for more information.

Interface: VolumeCharging

Not supported

Interface: ReserveAmountCharging

The endpoint for this interface is:
http://<host>:<port>/parlayx30/payment/ReserveAmountCharging

Where values for host and port depend on the Oracle Communications Services Gatekeeper deployment.

reserveAmount

Reserves a charge for an account indicated by the end user identifier.

Table 10-17 Exceptions and Error Codes

Exception	Error code	Explanation
Protocol Errors		
SVC0001	PAYMENT000002	A protocol related error returned from the Diameter server. Make sure the server is running and reachable. Check the log files for more information.
Transient Failures		
SVC0001	PAYMENT000003	A transient error returned from the Diameter server, such as authentication failure. Make sure the server is reachable and has adequate storage space. Check the log files for more information.
SVC0001	PLG-000004	General plug-in routing error.
Permanent Failures		
SVC0001	PAYMENT000004	A permanent failure returned from the Diameter server, such as invalid data in the AVP. Check the log files for more information.
SVC0270	PAYMENT000004	A permanent failure returned from the Diameter server, such as the user could not be authorized. Check the log files for more information.

reserveAdditionalAmount

Adds to or subtracts from a charge to or from an existing reservation.

Table 10-18 Exceptions and Error Codes

Exception	Error code	Explanation
Protocol Errors		
SVC0001	PAYMENT000002	A protocol related error returned from the Diameter server. Make sure the server is running and reachable. Check the log files for more information.
Transient Failures		
SVC0001	PAYMENT000003	A transient error returned from the Diameter server, such as authentication failure. Make sure the server is reachable and has adequate storage space. Check the log files for more information.
SVC0001	PLG-000004	General plug-in routing error.
Permanent Failures		
SVC0001	PAYMENT000004	A permanent failure returned from the Diameter server, such as invalid data in the AVP. Check the log files for more information.
SVC0270	PAYMENT000004	A permanent failure returned from the Diameter server, such as the user could not be authorized. Check the log files for more information.

chargeReservation

Charges to a reservation indicated by the reservation ID.

Table 10-19 Exceptions and Error Codes

Exception	Error code	Explanation
Protocol Errors		
SVC0001	PAYMENT000002	A protocol related error returned from the Diameter server. Make sure the server is running and reachable. Check the log files for more information.
Transient Failures		
SVC0001	PAYMENT000003	A transient error returned from the Diameter server, such as authentication failure. Make sure the server is reachable and has adequate storage space. Check the log files for more information.
SVC0001	PLG-000004	General plug-in routing error.
Permanent Failures		
SVC0001	PAYMENT000004	A permanent failure returned from the Diameter server, such as invalid data in the AVP. Check the log files for more information.
SVC0270	PAYMENT000004	A permanent failure returned from the Diameter server, such as the user could not be authorized. Check the log files for more information.

releaseReservation

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

Table 10-20 Exceptions and Error Codes

Exception	Error code	Explanation
Protocol Errors		
SVC0001	PAYMENT000002	A protocol related error returned from the Diameter server. Make sure the server is running and reachable. Check the log files for more information.
Transient Failures		
SVC0001	PAYMENT000003	A transient error returned from the Diameter server, such as authentication failure. Make sure the server is reachable and has adequate storage space. Check the log files for more information.
SVC0001	PLG-000004	General plug-in routing error.
Permanent Failures		
SVC0001	PAYMENT000004	A permanent failure returned from the Diameter server, such as invalid data in the AVP. Check the log files for more information.
SVC0270	PAYMENT000004	A permanent failure returned from the Diameter server, such as the user could not be authorized. Check the log files for more information.

Interface: ReserveVolumeCharging

Not supported

Parlay X 3.0 Part 11: Audio call

This set of interfaces is compliant to ETSI ES 202 504-11 v0.0.3 (2007-06), Open Service Access (OSA); Parlay X Web Services; Part 11: Audio Call (Parlay X 3).

Interface: PlayMedia

The endpoint for this interface is:
http://<host>:<port>/parlayx30/audio_call/AudioCallPlayMedia

Where values for host and port depend on the Oracle Communications Services Gatekeeper deployment.

playTextMessage

Not supported.

playAudioMessage

Plays a message to the given destination address. The message is given as a URL to an audio file. The file must be reachable by the underlying telecom network node and the audio-format must be supported by the telecom network.

Table 10-21 exceptions and error codes

Exception	Error code	Explanation
SVC0001	AC-100001	Call session has expired.
SVC0001	AC-100003	Call state is invalid.
SVC0001	AC-100004	Participant is not connected.
SVC0001	AC-100005	Not all participants are available. Possibly in already in playing or collecting mode.
SVC0001	AC-100006	Could not find callleg session information. Callsession may be empty.
SVC0001	OSA-000001	P_INVALID_NETWORK_STATE exception received from OSA Gateway.
SVC0001	OSA-000011	EC_OSA_P_INVALID_STATE exception received from OSA Gateway.
SVC0001	OSA-000012	TP_COMMON_EXCEPTIONS exception received from OSA Gateway.

Table 10-21 exceptions and error codes

Exception	Error code	Explanation
SVC0001	WNG-000002	Failed to store information in Services Gatekeeper. Contact support.
SVC0001	PLG-000004	General plug-in routing error.
SVC0002	callSessionIdentifier	Invalid input value for message part %1
SVC0002	callParticipants	Invalid input value for message part %1
SVC0002		P_INVALID_SESSION_ID exception received from OSA Gateway.
SVC0261		Call has already been terminated.
POL0001	Service contract not found	No Service Level Agreement found.
POL0001	AC-100002	Application is not the owner of the call.
POL0008		Charging not supported.

playVoiceXmlMessage

Not supported.

playVideoMessage

Not supported.

getMessageStatus

Gets the status of a message, that is, if the message is currently being played, if it is has finished playing and more.

Table 10-22 exceptions and error codes

Exception	Error code	Explanation
SVC0001	WNG-000002	Failed to store information in Oracle Communications Services Gatekeeper. Contact support.
SVC0001	PLG-000004	General plug-in routing error.
SVC0002	correlator	Invalid input value for message part %1.
POL0001	Service contract not found	No Service Level Agreement found.
POL0001	AC-100002	Application is not the owner of the call.

endMessage

Cancel or stops the playing of the message.

Table 10-23 exceptions and error codes

Exception	Error code	Explanation
SVC0001	WNG-000002	Failed to fetch information from Oracle Communications Services Gatekeeper. Contact support.
SVC0001	PLG-000004	General plug-in routing error.
SVC0002	correlator	Invalid input value for message part %1
POL0001	Service contract not found	No Service Level Agreement found.
POL0001	AC-100002	Application is not the owner of the call.

Interface: CaptureMedia

The endpoint for this interface is:

http://<host>:<port>/parlayx30/audio_call/AudioCallCaptureMedia

Where values for host and port depend on the Oracle Communications Services Gatekeeper deployment.

startPlayAndCollectInteraction

Starts a media interaction with one or all participants in a call session. Plays a media file and collects digits from one or all call participants. The results of the interaction is notified using the operation notifyPlayAndCollectEvent in the Parlay X 3.0 Part 3: Call Notification set of interfaces.

Table 10-24 exceptions and error codes

Exception	Error code	Explanation
SVC0001	WNG-000002	Failed to store information in Oracle Communications Services Gatekeeper. Contact support.
	AC-100001	Call session has expired.
	AC-100003	Call state is invalid.
	AC-100004	Participant is not connected.
	AC-100005	Not all participants are available. Possibly in already in playing or collecting mode.
SVC0001	AC-100006	Could not find callleg session information. Callsession may be empty.
SVC0001	OSA-000001	P_INVALID_NETWORK_STATE exception received from OSA Gateway.
SVC0001	OSA-000011	EC_OSA_P_INVALID_STATE exception received from OSA Gateway.
SVC0001	OSA-000012	TP_COMMON_EXCEPTIONS exception received from OSA Gateway.
SVC0001	PLG-000004	General plug-in routing error.
SVC0002	callSessionIdentifier	Invalid input value for message part %1
SVC0002	callParticipants	Invalid input value for message part %1
SVC0002	playFileLocation	Invalid input value for message part %1

Table 10-24 exceptions and error codes

Exception	Error code	Explanation
SVC0002		P_INVALID_SESSION_ID exception received from OSA Gateway.
SVC0261		Call has already been terminated
POL0001	Service contract not found	No Service Level Agreement found.
POL0001	AC-100002	Application is not the owner of the call.
POL0001	AC-100007	The value of maxDigits is too big.
POL0001	AC-100008	The value of minDigits is too small.

startPlayAndRecordInteraction

Not supported.

stopMediaInteraction

Explicitly stops an ongoing media interaction session.

Table 10-25 exceptions and error codes

Exception	Error code	Explanation
SVC0001	OSA-000001	P_INVALID_NETWORK_STATE exception received from OSA Gateway.
SVC0001	OSA-000011	EC_OSA_P_INVALID_STATE exception received from OSA Gateway.
SVC0001	OSA-000012	TP_COMMON_EXCEPTIONS exception received from OSA Gateway.
SVC0001	WNG-000002	Failed to store information in Oracle Communications Services Gatekeeper. Contact support.
SVC0001	PLG-000004	General plug-in routing error.

Table 10-25 exceptions and error codes

Exception	Error code	Explanation
SVC0002	mediaIdentifier	Invalid input value for message part %1
SVC0002		P_INVALID_SESSION_ID exception received from OSA Gateway.
POL0001	Service contract not found	No Service Level Agreement found.
POL0001	AC-100002	Application is not the owner of the call.

Interface: Multimedia

addMediaForParticipants

Not supported.

deleteMediaForParticipants

Not supported.

getMediaForParticipant

Not supported.

getMediaForCall

Not supported.

General Exceptions

This section describes the exception handling for the Parlay X 3.0 interfaces.

These exception types are defined:

- Service Exceptions
- Policy Exceptions

Service Exception are related to the operation of the service itself. The following exceptions are general:

- SVC0001: Service error.
- SVC0002: Invalid input value
- SVC0003: Invalid input value with list of valid values
- SVC0004: No valid addresses
- SVC0005: Duplicate correlator
- SVC0006: Invalid group
- SVC0007: Invalid charging information
- SVC0008: Overlapping Criteria

PolicyExceptions are thrown when a policy has been violated, including violations of a service level agreements. The following general PolicyExceptions are defined:

- POL0001: Policy error
- POL0002: Privacy error
- POL0003: Too many addresses specified
- POL0004: Unlimited notifications not supported
- POL0005: Too many notifications requested
- POL0006: Groups not allowed
- POL0007: Nested groups not allowed
- POL0008: Charging not supported
- POL0009: Invalid frequency requested

Within the exception, an error code is defined. The error code details why the exception was thrown.

CHAPTER 11

Native Interfaces

This chapter describes the supported native interfaces and contains information that is specific for Oracle Communications Services Gatekeeper, and not found in the specifications. For detailed descriptions of the interfaces, methods and parameters, refer to the specifications.

MM7

The MM7 specification can be found at http://www.3gpp.org/ftp/Specs/html-info/23140.htm. Messages are compliant with the schema defined by Rel-5-MM7-1-2.xsd. Because the network-facing interface supports Rel-5-MM7-1-5.xsd, Rel-5-MM7-1-2.xsd and a modified version of REL-5-MM7-1-0.xsd, some mapping may be done during processing.

The endpoint for this interface is: http://<host>:<port>/mm7/Mms

where values for host and port depend on the Oracle Communications Services Gatekeeper deployment.

Note: The MM7 facade uses HTTP basic authentication, username/password. The username is the application instance ID.

MM7_submit

Sends an application-initiated multimedia message

Table 11-1 Error Codes

Error code	Reason/Action
4006	Service unavailable. Communication error within Oracle Communications Services Gatekeeper or between Oracle Communications Services Gatekeeper and MMSC Transient error. The client should try again.
4007	Service denied. The request was not allowed by policy. Contact Oracle Communications Services Gatekeeper administrator.
<all MMSC fault codes>	Passed along transparently Contact Oracle Communications Services Gatekeeper administrator.

MM7_deliver

Oracle Communications Services Gatekeeper delivers a network-triggered message to the application, at an endpoint implemented by the application.

MM7_cancel

Not supported.

MM7_replace

Not supported

MM7_delivery_report

Oracle Communications Services Gatekeeper delivers a delivery report on a previously sent message to the application, at an endpoint implemented by the application.

MM7_read_reply_report

Oracle Communications Services Gatekeeper delivers a read reply report on a previously sent message to the application, at an endpoint implemented by the application.

SMPP

The Native SMPP Service Facade exposes SMPP version 3.4 to applications.

The specification ha the title *Short Message Peer to Peer, Protocol Specification v3.4, Document Version:- 12-Oct-1999 Issue 1.2* and can be downloaded from http://smsforum.net/

It supports all Protocol Data Units (PDUs), and all header and body elements except when stated otherwise.

An application using this interface acts as an External Short Message Entity (ESME).

General for bind PDUs and Sessions

An application must bind to the Native SMPP Communications Service. It can bind using:

- bind_transmitter PDU
- bind_receiver PDU
- bind_transceiver PDU

As a result of a bind operation, Oracle Communications Services Gatekeeper authenticates the application and a session established.

The following is valid for all bind operations:

- An application binds using host name or IP-address and a port that depends on the installation. The server to bind to is a Network Tier Server.
- system_id field must be the application instance group ID assigned to the application instance.
- password field must be the same as the password for the application instance group.

Oracle Communications Services Gatekeeper Native SMPP Communication Service can be configured to allow only a certain number of sessions per application.

A session is maintained until the application sends an unbind PDU.

Oracle Communications Services Gatekeeper can be configured to terminate a session if:

- there is too long time of inactivity on the session.
- the application takes too long time to respond to a request.

Error Handling

All errors are reported in the command_status field of a response PDU.

Table 11-2 lists the error codes that are specific for Oracle Communications Services Gatekeeper. Errors from the SMSC are transparently forwarded to the application.

Table 11-2 Error codes for SMPP Communication Service

SMPP PDU	Error Code in Response (command_status)	Description
bind_transmitter	ESME_RBINDFAIL	Could not bind.
bind_receiver	ESME_RBINDFAIL	Could not bind.
bind_transceiver	ESME_RBINDFAIL	Could not bind.
submit_sm	ESME_RTHROTTLED	Throttling limit or quota limit exceeded. The application has performed too many requests per time unit and has exceeded the Service Level Agreement.
	ESME_RSUBMITFAIL	Could not submit the message. Possible reasons includes no Service Level Agreement found, time-out encountered when sending the message, and configuration error.
submit_sm_multi	ESME_RTHROTTLED	Same as for submit_sm.
	ESME_RSUBMITFAIL	Same as for submit_sm.

bind_transmitter PDU

The application binds as an SMPP transmitter to Oracle Communications Services Gatekeeper.

Oracle Communications Services Gatekeeper Native SMPP Communication Service can be configured to:

- allow or block this operation.
- allow only a certain number of sessions to be established.

bind_transmitter_resp PDU

Oracle Communications Services Gatekeeper sends this PDU to an application as a response to bind_transmitter PDU.

bind_receiver PDU

The application binds as an SMPP receiver to Oracle Communications Services Gatekeeper.

The address_range field must be the same as provisioned for the application instance group.

Oracle Communications Services Gatekeeper Native SMPP Communication Service can be configured to:

- allow or block this operation.
- allow only a certain number of sessions to be established.

bind_receiver_resp PDU

Oracle Communications Services Gatekeeper sends this PDU to an application as a response to bind_receiver PDU.

bind_transceiver PDU

The application binds as an SMPP transceiver to Oracle Communications Services Gatekeeper.

address_range field must be the same as provisioned for the application instance group.

Oracle Communications Services Gatekeeper Native SMPP Communication Service can be configured to:

- allow or block this operation.
- allow only a certain number of sessions to be established.

bind_transceiver_resp PDU

Oracle Communications Services Gatekeeper sends this PDU to an application as a response to bind_transceiver PDU.

outbind PDU

Not supported.

unbind PDU

The application unbinds from Oracle Communications Services Gatekeeper.

unbind_resp PDU

Oracle Communications Services Gatekeeper sends this PDU to an application as a response to unbind PDU.

generic_nack PDU

Oracle Communications Services Gatekeeper sends this PDU as a negative acknowledgement of a PDU sent from the application if the PDU can not be recognized.

If this PDU is sent from the application, it is propagated to the SMPP SMSC.

submit_sm PDU

The application sends a short message to Oracle Communications Services Gatekeeper, which forwards it to the destination address via an SMSC.

submit_sm_resp PDU

Oracle Communications Services Gatekeeper sends this PDU to an application as a response to submit_sm PDU.

submit_multi PDU

The application sends a short message to Oracle Communications Services Gatekeeper, which forwards it to a set of destination addresses via an SMSC.

submit_multi_resp PDU

Oracle Communications Services Gatekeeper sends this PDU to an application when as a response to submit_multi PDU.

deliver_sm PDU

Oracle Communications Services Gatekeeper sends this PDU to an application when a network-triggered short message that matches the destination addresses that the application has subscribed for notifications on has been received by Oracle Communications Services Gatekeeper from an SMSC. The PDU contains the short message.

deliver_sm_resp PDU

The application sends this PDU to acknowledge the reception of a deliver_sm PDU.

data_sm PDU

Not supported.

data_sm_resp PDU

Not supported.

query_sm PDU

The application sends this PDU to query the status of a previously sent short message.

Oracle Communications Services Gatekeeper Native SMPP Communication Service can be configured to allow or block this operation.

query_sm_resp PDU

Oracle Communications Services Gatekeeper sends this PDU to an application as a response to query_sm PDU.

cancel_sm PDU

The application sends this PDU to cancel the sending of one more previously sent short messages, given that the message has not yet been delivered to the end-user terminal.

Oracle Communications Services Gatekeeper Native SMPP Communication Service can be configured to allow or block this operation.

cancel_sm_resp PDU

Oracle Communications Services Gatekeeper sends this PDU to an application as a response to cancel_sm PDU.

replace_sm PDU

The application sends this PDU to replace a previously sent short message with the short message provided in this PDU, given that the message has not yet been delivered to the end-user terminal.

Oracle Communications Services Gatekeeper Native SMPP Communication Service can be configured to allow or block this operation.

replace_sm_resp PDU

Oracle Communications Services Gatekeeper sends this PDU to an application as a response to replace_sm PDU.

enquire_link PDU

The application, or Oracle Communications Services Gatekeeper, sends this PDU to verify the connection between the application and Oracle Communications Services Gatekeeper.

Oracle Communications Services Gatekeeper Native SMPP Communication Service can be configured to send this PDU to the application on a regular interval. When an application receives this PDU it must respond with enquire_link_resp PDU within a certain time-interval. This time-interval is configurable in Oracle Communications Services Gatekeeper Native SMPP Communication Service.

enquire_link_resp PDU

Oracle Communications Services Gatekeeper, or an application, sends this PDU as a response to enquire_link PDU.

alert_notification PDU

Not supported.

Access Web Service (deprecated)

Note: The Access Web Service is not deployed in a standard installation.

It is deprecated and should only be used by older, existing applications, in order to provide a migration path for these applications. *WebLogic Server Web Services security cannot be used when using the Access Web Service* and must be turned off in Oracle Communications Services Gatekeeper to be able to use the Access Web Service.

The Access Web Service contains operations for establishing a session with Oracle Communications Services Gatekeeper, changing the application's password, querying the amount of time remaining in the session, refreshing the session, and terminating the session.

Before an application can perform any operations on the Parlay X or Extended Web Services interfaces, a session must be established with Oracle Communications Services Gatekeeper. When a session is established, a session ID (loginTicket) is returned which must be used in each subsequent operation towards Oracle Communications Services Gatekeeper.

The loginTicket shall be present in the SOAP Header element Security, see below. Once the login ticket is acquired, it must be sent in the SOAP header together with a username/password combination each time a Web Service method is invoked. See Examples.

Endpoint

The WSDL for the Access Web Service can be found at
`http://<host:port>/parlayx21/access/Access`

where host and port depend on the Oracle Communications Services Gatekeeper deployment.

Interface: Access

Operations to establish a session, change a password, get the remaining lifetime of a session, refresh a session and destroy a session.

Operation: applicationLogin

Logs the application into the Oracle Communications Services Gatekeeper and retrieves a login ticket. This login ticket represents the session and must be added to the SOAP header of every subsequent request that the application makes to the Oracle Communications Services Gatekeeper.

In most cases, the login ticket is only valid for a certain time interval, set by the operator. Once the time period has expired, the application has a second operator-set time period to refresh the login ticket. Until the ticket is refreshed, the application can not make any other requests. The operation used to refresh the ticket is refreshLoginTicket, see Operation: refreshLoginTicket. If the ticket is not refreshed during this second period, the session is destroyed, and the application must log back in.

Input message: applicationLoginRequest

Part name	Part type	Optional	Description
serviceProvider	s1:String	N	ID of the service provider as given by the operator or the service provider.
application	s1:String	N	ID of the application as given by the operator or the service provider.
applicationInstanceGroup	s1:String	N	ID of the application instance group as given by the operator or the service provider.
password	s1:String	N	Password for the application as given by the operator or the service provider. Note that this may also have been changed by the by the application provider.

Output message: applicationLoginResponse

Part name	Part type	Optional	Description
applicationLoginReturn	s1:string	N	ID of the login-session. This ID is used for each request towards Oracle Communications Services Gatekeeper. It must be included in the SOAP header of every subsequent request. If an application logs in several times, the same ID is returned.

Referenced faults

AccessException

GeneralException

Operation: applicationLogout

Logs an application out of the Oracle Communications Services Gatekeeper. Destroys the login session and the corresponding login ticket.

Input message: applicationLogoutRequest

Part name	Part type	Optional	Description
loginTicket	s1:string	N	ID of the login-session. The login ticket is retrieved when the application logs in or when it refreshes the login ticket.

Output message: applicationLogoutResponse

Part name	Part type	Optional	Description
-	-	-	-

Referenced faults

AccessException

GeneralException

Operation: changeApplicationPassword

Changes the password for an application.

Input message: changeApplicationPasswordRequest

Part name	Part type	Optional	Description
loginTicket	s1:string	N	ID of the login-session. The login ticket is retrieved when the application logs in or when it refreshes the login ticket.
oldPassword	s1:string	N	The current password.
newPassword	s1:string	N	The new password.

Output message: changeApplicationPasswordResponse

Part name	Part type	Optional	Description
-	-	-	-

Referenced faults

AccessException

GeneralException

Operation: getLoginTicketRemainingLifeTime

Reports the remaining amount of time the login ticket is valid.

Input message: getLoginTicketRemainingLifeTimeRequest

Part name	Part type	Optional	Description
sessionId	s1:string	N	ID of the login-session. The login ticket is retrieved when the application logs in or when it refreshes the login ticket.

Output message: getLoginTicketRemainingLifeTimeReturn

Part name	Part type	Optional	Description
getLoginTicketRemainingLifeTimeReturn	s1:int	N	The time until the login ticket expires. The time is given in minutes.

Referenced faults

AccessException

GeneralException

Operation: refreshLoginTicket

Refreshes the login ticket. This refreshed login ticket must be provided in the SOAP header in all subsequent method calls. The login ticket can be refreshed for a limited, operator-set time interval after the previous login ticket has expired. If this time interval expires, the application must login again. Oracle Communications Services Gatekeeper expiration timers are reset, but the same login ticket is returned.

Input message: refreshLoginTicketRequest

Part name	Part type	Optional	Description
loginTicket	s1:string	N	The ID of an established session.
serviceProviderID	s1:string	N	ID of the service provider as given by the operator or the service provider.
applicationID	s1:string	N	ID of the application as given by the operator or the service provider.
applicationInstanceGroupID	s1:string	N	ID of the application instance group as given by the operator or the service provider.
password	s1:string	N	Password for the application as given by the operator or the service provider. Note that this may also have been changed by the by the application provider.

Output message: refreshLoginTicketResponse

Part name	Part type	Optional	Description
refreshLoginTicketReturn	s1:string	N	The refreshed ID of the login-session. This ID is used in each request towards Oracle Communications Services Gatekeeper. It must be included in the SOAP header of every subsequent request.

Referenced faults

AccessException

GeneralException

Exceptions

AccessException

Exceptions of this type are raised when there are error conditions related to the Access Web Service. Other error conditions are reported using the exception GeneralException.

Part name	Part type	Optional	Description
exceptionMessage	xsd:string	Y	Description of exception.
errorCode	xsd:int	N	Code defining the exception.

GeneralException

Exceptions of this type are raised when the applications session has expired or there are communication problems with the underlying platform.

Part name	Part type	Optional	Description
exceptionMessage	xsd:string	Y	Description of exception.
errorCode	xsd:int	N	Code defining the exception.

Examples

Defining the security header

The loginTicket shall be present in the SOAP Header element Security, see below. Once the login ticket is acquired, it must be sent in the SOAP header together with a username/password combination each time a Web Service method is invoked.

The Access Web Service uses a security header described below.

The loginTicket is supplied in the Password attribute.

Listing 12-1 Access security header (example)

```
<soapenv:Header>
  <ns1:Security ns1:Username="app:-2810834922008400383"
    ns1:Password="app:-2810834922008400383" soapenv:actor="wsse:PasswordToken"
    soapenv:mustUnderstand="1"
    xmlns:ns1="http://localhost:6001/parlayx21/terminal_location/TerminalLocat
ion">
  </ns1:Security>
</soapenv:Header>
```

Below is an example of how to add an Access security header using Axis. The Username attribute must be present but is not used. The header must be added to the Web Service request.

Listing 12-2 Access security header (Axis)

```
org.apache.axis.message.SOAPHeaderElement header =
 new org.apache.axis.message.SOAPHeaderElement(wsdlUrl, "Security", "");
header.setActor("wsse:PasswordToken");
 header.addAttribute(wsdlUrl, "Username", ""+userName);
 header.addAttribute(wsdlUrl, "Password", ""+loginTicket);
 header.setMustUnderstand(true)
```