Oracle® Communication Services Gatekeeper

Communication Service Reference

Release 4.0

June 2008

ORACLE

Oracle Communication Services Gatekeeper Communication Service Reference, Release 4.0

Contents

4. Parlay X 3.0 Third Party Call Communication Service

5. Parlay X 2.1 Call Notification Communication Service

6. Parlay X 3.0 Call Notification Communication Service

7. Parlay X 2.1 Short Messaging Communication Service

8. Extended Web Services Binary SMS Communication Service

9. Parlay X 2.1 Multimedia Messaging Communication Service

10. Parlay X 2.1 Terminal Location Communication Services

11. Parlay X 2.1 Presence Communication Service

12. Extended Web Services Subscriber Profile Communication Service

13. Extended Web Services WAP Push Communication Service

A. Events, Alarms, and Charging

CHAPTER 1

Document Roadmap

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

Document Scope and Audience

This document provides a detailed reference for information that is specific to individual communication services. It includes:

- An overview of each communication service's functioning
- The network protocols each application facing interface supports
- Configuration specifics, including communication service specific:
  - Charging Data Records
  - Event Data Records

This document will be of use to system administrators charged with installing and maintaining WebLogic Network Gatekeeper, as well as managers, support engineers, and sales and marketing people. For an overview of the characteristics that all communication services have in common,

please see the "Introducing Communication Services" chapter in *Concepts and Architectural Overview*, another document in this set.

Guide to this Document

This document contains the following chapters:

Chapter 1, "Document Roadmap." This chapter

Chapter 2, "Parlay X 3.0 Audio Call Communication Service." Information on the communication service that supports the Parlay X 3.0 Audio Call interface

Chapter 3, "Parlay X 2.1 Third Party Call Communication Services." Information on the communication services that support the Parlay X 2.1 Third Party Call interface

Chapter 4, "Parlay X 3.0 Third Party Call Communication Service." Information on the communication service that supports the Parlay X 3.0 Third Party Call interface

Chapter 5, "Parlay X 2.1 Call Notification Communication Service." Information on the communication service that supports the Parlay X 2.1 Call Notification interface

Chapter 6, "Parlay X 3.0 Call Notification Communication Service." Information on the communication service that supports the Parlay X 3.0 Call Notification interface

Chapter 7, "Parlay X 2.1 Short Messaging Communication Service." Information on the communication service that support the Parlay X 2.1 Short Messaging interface

Chapter 8, "Extended Web Services Binary SMS Communication Service." Information on the communication service that supports Extended Web Service Binary SMS interface

Chapter 9, "Parlay X 2.1 Multimedia Messaging Communication Service." Information on the communication services that support the Parlay X 2.1 Multimedia Messaging interface

Chapter 10, "Parlay X 2.1 Terminal Location Communication Services."Information on the communication services that support the Parlay X 2.1 Terminal Location interface

Chapter 11, "Parlay X 2.1 Presence Communication Service." Information on the communication service that supports the Parlay X 2.1 Presence interface

Chapter 12, "Extended Web Services Subscriber Profile Communication Service." Information on the communication service that supports the Extended Web Services Subscriber Profile interface.

Chapter 13, "Extended Web Services WAP Push Communication Service." Information on the communication service that supports the Extended Web Services WAP Push Message interface

Terminology

The following terms and acronyms may be 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 Network 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 Network Gatekeeper administration. An Application Instance has a user name and password
- 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
- Communication Service—A mechanism by which a particular telecom network capability is made available to Internet based applications. It consists of an application-facing interface (north), a generic capability, and a network-facing interface (south).

- 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 WebLogic Network 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 WebLogic Network 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 Network Gatekeeper. Usually the network operator
- OSA—Open Service Access
- PAP—Push Access Protocol
- Plug-in—See Network Plug-in
- Plug-in Manager—The Network Gatekeeper module charged with routing an application-initiated request to the appropriate network plug-in
- Policy Engine—The Network Gatekeeper module charged with evaluating whether a particular request is acceptable under the rules
- 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—The customizable set of criteria - based on SLAs and operator-desired additions - 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. WebLogic Network Gatekeeper can interact with these on its network-facing interface
- Service Capability—Support for a specific kind of traffic within WebLogic Network Gatekeeper. Defined in terms of communication services
- 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
- USSD—Unstructured Supplementary Service Data
- VAS—Value Added Service
- VLAN—Virtual Local Area Network
- VPN—Virtual Private Network
- WebLogic Network Gatekeeper Core—The container that holds the Core Utilities
- WebLogic Network Gatekeeper Core Utilities—A set of utilities common to all communication services

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

Related Documentation

This communication service reference is a part of the WebLogic Network Gatekeeper documentation set. The other documents include:

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

CHAPTER 2

Parlay X 3.0 Audio Call Communication Service

The following chapter describes the Parlay X 3.0 Audio Call communication service in detail:

An Overview of the Parlay X 3.0 Audio Call Communication Service

The Audio Call communication service implements the Parlay X 3.0 Audio Call set of interfaces. For the exact version of the standard the communication service supports, see "Appendix A: Standards and Specifications" of the *Concepts and Architectural Overview*, a separate document in this set.

Using the Audio Call communication service, an application can:

- Play audio to one or more call participants in an existing call session set up by the Parlay X 3.0 Third Party Call communication service.
- Find out if the audio is currently being played or has not yet started to play.
- Explicitly end playing of the audio.
- Collect digits from call participants in response to an audio message that has been played to them and, in conjunction with the Parlay X 3.0 Call Notification communication service, return the information to the application.
- Interrupt an ongoing interaction such as on-hold music.

How It Works

The Audio Call communication service can be used by applications to play audio messages to one or more call participants in an existing call. The existing call is identified by the `callSessionIdentifier` returned to the application at the time the call session is set up using the Parlay X 3.0 Third Party Call communication service. If desired, applications can receive digits collected from those participants in response to the audio message using a notification set up using the Parlay X 3.0 Call Notification communication service.

The audio message content to be played must be defined in a binary format such as WAV stored at a URL available to the network and rendered by an audio player. Network Gatekeeper does *not* actually render the message: this is the responsibility of equipment that must be present on the target telecom network, such as Interactive Voice Response (IVR) systems.

Supported Network Protocols

Off-the shelf, the Parlay X 3.0 Audio Call communication service supports a combination of the following network protocols:

- Parlay 3.3 Multiparty Call Control and Call User Interaction

Parlay 3.3 Multiparty Call Control and Call User Interaction

When Network Gatekeeper is configured to use these protocols, it connects to a Parlay 3.3 MultiParty Call Control SCS and a Parlay 3.3 Call User Interaction SCS provided by an OSA/Parlay Gateway. See "Appendix A: Standards and Specifications" of the *Concepts and Architectural Overview*, a separate document in this set, for the exact version of the standards Network Gatekeeper supports.

Configuration Specifics for the Audio Call Communication Service

Communication services share many common features, covered in "Introducing Communication Services" in the *Concepts and Architectural Overview*, but each one has a few characteristics that are specific only to that service. This section describes those specific features for the Audio Call communication service, including:

- Charging Data Records
- Event Data Records
- Statistics
- Supported Address Schemes

Charging Data Records

Generation of CDRs

There are two Audio Call-Parlay 3.3 MPCC/CUI specific CDRs. They occur when the following criteria are met:

- When `sendInfoRes` is sent from the network to Network Gatekeeper, indicating that the audio message has completed playing, if this is not the result of an explicit request to stop from the application.
- When `sendInfoAndCollectRes` is sent from the network to Network Gatekeeper, indicating that audio message has completed playing and the call participant's response has been collected in the form of digits.

Event Data Records

The following is a list of `EdrIds` created by the Audio Call-Parlay 3.3 MPCC/CUI communication service. This list does not include EDRs created when exceptions are thrown. For more information, see the Appendix A, “Events, Alarms, and Charging.”

Table 2-1 Event types emitted in Audio Call communication service

EdrId	Method Called
11100	playAudioMessage
11101	getMessageStatus
11102	endMessage
11103	startPlayAndCollectInteraction
11104	stopMediaInteraction
11105	sendInfoRes
11106	SendInfoErr
11107	sendInfoAndCollectRes
11108	SendInfoAndCollectErr
11109	attachMediaRes
111010	attachMediaErr
111011	detachMediaRes
111012	detachMediaErr
111013	abortActionRes
111014	abortActionErr

Statistics

The table below outlines the correlation between the methods being invoked from either the application (in application-initiated requests) or the telecom network (in network-initiated requests) and the transaction type collected by the statistics counters in Network Gatekeeper for the Parlay X 3.0 Audio Call-Parlay 3.3 MPCC/CUI communication service:

Table 2-2 Transaction types for Parlay X 3.0 Audio Call-Parlay MPCC/CUI communication service

Method	Transaction type
startPlayAndCollectInteractions	TRANSACTION_TYPE_CALL_CONTROL_SERVICE_INITIATED
playAudioMessage	TRANSACTION_TYPE_CALL_CONTROL_SERVICE_INITIATED

Supported Address Schemes

The Parlay X 3.0 Audio Call-Parlay 3.3 MPCC/CUI communication service does not use any address schemes directly, but in working with the Parlay X 3.0 Third Party Call and Call Notification communication services, it uses the `tel:` scheme indirectly.

CHAPTER 3

Parlay X 2.1 Third Party Call Communication Services

The following chapter describes the Parlay X 2.1 Third Party Call communication services in detail:

An Overview of the Parlay X 2.1 Third Party Call Communication Services

The Third Party Call Parlay X 2.1 communication services implement the Parlay X 2.1 Third Party Call interface. For the exact version of the standard these communication services support, see "Appendix A: Standards and Specifications" in the *Concepts and Architectural Overview*, another document in this set.

Using a Third Party Call Parlay X 2.1 communication service, an application can:

- Setup a call between two parties. For example, an application could setup a call between an investor and her broker if a particular stock reaches a pre-determined price. Or, more simply, a computer user could setup a call between himself and someone in his Address Book with a simple click of the mouse.
- Query Network Gatekeeper for the status of a previously setup call
- Cancel a call it is creating as it is about to be setup
- Terminate an ongoing call it created

How It Works

In the Parlay X 2.1 Third Party Call communication services model, a call has two distinct stages:

- Call Setup
- Call Duration

Call Setup

There are two parties involved in Third Party Call calls: the A-party (the caller) and the B-party (the callee). When a call is set up using a Third Party Call communication service, first Network Gatekeeper attempts to set up a call leg to the A-party. Once the caller goes off-hook (“answers”), Network Gatekeeper attempts to set up a call leg to the B-party. When the callee goes off-hook, the two call legs are connected using the underlying telecom network. This ends the call setup-phase.

The application can cancel the call during this phase.

Call Duration

While the call is underway, the audio channel that connects the caller and the callee is completely managed by the underlying telecom network. During this phase of the call, the application can only query as to the status of the call. A call can be terminated in two ways, either using the application-facing interface, or having the caller or callee “hang up.”

Requests using a Parlay X 2.1 Third Party Call communication service flow only in one direction, from the application to the network. Thus this communication service only supports “application-initiated” or “mobile terminated” functionality.

Note: Third Party Call communication services manage only the signalling, or controlling, aspect of a call. The media, or audio, channel is managed by the underlying telecom network. Only parties residing on the same network can be controlled, unless:

- The network plug-in connects to a media gateway controller
- One of the participants is connected to a signalling gateway so that, from a signalling point-of-view, all parties reside on the same network

Supported Network Protocols

Off-the shelf, Parlay X 2.1 Third Party Call communication services can be configured to support the following network protocols:

- SIP
- INAP/SS7

SIP

When Network Gatekeeper is configured to use this protocol it connects to the SIP network. In this case, Network Gatekeeper acts as a Back-to-Back-User-Agent. During the call duration phase the actual call is purely peer-to-peer.

INAP/SS7

When Network Gatekeeper is configured to use this protocol, it connects to the underlying network via an INAP/SS7 interface. See "Appendix A: Standards and Specifications" in the *Concepts and Architectural Overview* for the exact version of the standard Network Gatekeeper supports.

Configuration Specifics for the Parlay X 2.1 Third Party Call Communication Services

Communication services share many common features, covered in the "Introducing Communication Services" chapter of the *Architectural Overview*, but each one has a few characteristics that are specific only to that service. This section describes those specific features for the Parlay X 2.1 Third Party Call communication services, including:

- Charging Data Records
- Event Data Records
- Statistics
- Supported Address Schemes

Charging Data Records

Generation of CDRs

The Parlay X 2.1 Third Party Call using SIP and the Parlay X 2.1 TPC using INAP/SS7 are standard communication services. CDRs are written:

- When Network Gatekeeper has received an event from the network stating that the second call leg has been connected and the associated phone has started to ring. This CDR is *not* dependent on whether the call is answered.
- When Call Information has been successfully delivered to the application
- When the call is ended by the application
- When the call request is canceled by the application

In addition, Parlay X 21 Third Party Call using INAP/SS7 also writes CDRs when:

- When the network notifies Network Gatekeeper that the call is connected, that is, that the second participant has answered the call
- When the network notifies Network Gatekeeper that a call participant has disconnected.

Event Data Records

Both the Parlay X 2.1 Third Party Call SIP and the TPC INAP/SS7 communication services use the 3.0 mechanism and are documented in Table 3-1 and Table 3-2 below. This list does not include EDRs created when exceptions are thrown. For more information, see Appendix A, “Events, Alarms, and Charging.”.

Table 3-1 Event types emitted in the Parlay X 2.1 Third Party Call/SIP communication service

Event data	Description
8022	makeCall
8023	getCallInformation
8024	endCall
8025	cancelCallRequest

.The following events are generated for the INAP/SS7 communication service in addition to those listed in Table 3-1 above:

Table 3-2 Event types emitted in the Parlay X 2.1 Third Party Call/INAP-SS7 communication service

Event data	Description
8026	callConnected
8027	callReleasedNotification

Statistics

The table below outlines the correlation between the methods being invoked from either the application (in application-initiated requests) or the telecom network (in network-initiated requests) and the transaction type collected by the statistics counters in Network Gatekeeper for either the Parlay X 2.1 Third Party Call/SIP or INAP/SS7 communication services:

Table 3-3 Transaction types for Parlay X 2.1 Third Party Call communication service

Method	Transaction type
makeCall	TRANSACTION_TYPE_CALL_CONTROL_SERVICE_INITIATED

Supported Address Schemes

The Parlay X 2.1 Third Party Call using either SIP or INAP/SS7 communication services supports the `tel:` address scheme.

CHAPTER 4

Parlay X 3.0 Third Party Call Communication Service

The following chapter describes the Parlay X 3.0 Third Party Call communication service in detail:

An Overview of the Parlay X 3.0 Third Party Call Communication Service

This communication service implements the Parlay X 3.0 Third Party Call interface. For the exact version of the standard this communication service supports, see "Appendix A: Standards and Specifications" in the *Concepts and Architectural Overview*, another document in this set.

Using the Third Party Call Parlay X 3.0 communication service, an application can:

- Setup a uniquely identified call between one or more participants and add or delete further participants, or transfer them to other established calls
- Indicate charging information to be associated with the call session
- Indicate information on any media to be used in association with the call
- Interact with the functionality of other communication services, such as Audio Call to play audio to call participants or Call Notification to respond to previously established notifications
- Query Network Gatekeeper for the status of an established call or particular call participants
- Terminate an ongoing call it created

How It Works

The Parlay X 3.0 Third Party Call communication service can be used by applications that need to set up calls to one or more participants, as, for example, in establishing a conference call. It can also be used to setup calls that also use the capabilities of other communication services (Audio Call or Call Notification). The application first sets up the call session by using the `makeCallSession` operation, passing in the address of at least one (the A-party) participant.

Note: In the most common case, the address of a second participant, the B-party, is also passed in.

Network Gatekeeper sends a request to establish the first call leg to the underlying network and returns a unique identifier (the `callSessionIdentifier`) to the application synchronously. This identifier allows the application to perform further administrative tasks on the call, as well as provides any other communication services (Audio Call or Call Notification) access to the call at any point during the call session.

Note: The `callSessionIdentifier` is returned to the application before the A-party goes off hook (answers). To receive information on the ongoing status of the call session, the application polls Network Gatekeeper, using the identifier and the `getCallSessionInformation` operation.

While the call is underway, the application can add additional parties, delete one or more parties, or transfer parties to and from other established call sessions, using the identifier returned during the call setup phase. The functionality of the Audio Call (playing media to one or more participants) and Call Notification (for example re-routing a “busy” address to a second pre-defined one) communication services can also be accessed in the context of the call session

using this same identifier. A call can be terminated in two ways, either using the application-facing interface, or having the call participants "hang up."

Requests using the Third Party Call communication service flow only in one direction, from the application to the network. Thus by itself this communication service only supports "application-initiated" or "mobile terminated" functionality. Mobile originated scenarios can be supported when using this communication service in concert with Call Notification.

Note: The Parlay X 3.0 Third Party Call communication service manages only the signalling, or controlling, aspect of a call. The call itself takes place in the underlying telecom network. Only parties residing on the same network can be controlled, unless:

- The network plug-in connects to a media gateway controller
- One of the participants is connected to a signalling gateway so that, from a signalling point-of-view, all parties reside on the same network

Supported Network Protocols

Off-the shelf, the Parlay X 3.0 Third Party Call communication service can be configured to support the following network protocol:

- Parlay 3.3 MultiParty Call Control

Parlay 3.3 MultiParty Call Control

When Network Gatekeeper is configured to use this protocol, it connects to a Parlay 3.3 MultiParty Call Control SCS provided by an OSA/Parlay Gateway. See "Appendix A: Standards and Specifications" in the *Concepts and Architectural Overview* for the exact version of the standard Network Gatekeeper supports.

Configuration Specifics for the Parlay X 3. 0 Third Party Call Communication Service

Communication services share many common features, covered in the "Introducing Communication Services" chapter of the *Concepts and Architectural Overview*, but each one has a few characteristics that are specific only to that service. This section describes those specific features for the Parlay X 3. 0 Third Party Call communication service, including:

- Charging Data Records
- Event Data Records

- Statistics
- Supported Address Schemes

Charging Data Records

Generation of CDRs

There are seven specific Parlay X 3.0 Third Party Call-Parlay 3.3 MPCC communication service CDRs. They occur when the following criteria are met:

- After Network Gatekeeper has created the first call leg of a call session. This is not dependent on whether the participant has answered.
- After a call participant has been added to a call session, deleted from a session, or transferred to or from another session.
- When call information or call participant information has been successfully delivered to the application
- When the call is ended by the application

Event Data Records

The EDRs produced by the Parlay X 3.0 Third Party Call/Parlay 3.3 communication service are of the standard type and are documented below. This list does not include EDRs created when exceptions are thrown. For more information on the contents of standard EDRs, see Appendix A, "Events, Alarms, and Charging.".

Table 4-1 Event types emitted in the Parlay X 3.0 Third Party Call/Parlay 3.3 communication service

EdrId	Method Called
10000	addCallParticipant
10001	deleteCallParticipant
10002	endCallSession
10003	getCallParticipantInformation
10004	getCallSessionInformation
10005	makeCallSession

Table 4-1 Event types emitted in the Parlay X 3.0 Third Party Call/Parlay 3.3 communication service

EdrId	Method Called
10006	transferCallParticipant
10007	eventReportRes
10008	eventReportErr
10009	callLegEnded
10010	callEnded
10011	createAndRouteCallLegErr
10012	getInfoRes
10013	getInfoErr

Statistics

The table below outlines the correlation between the methods being invoked from either the application (in application-initiated requests) or the telecom network (in network-initiated requests) and the transaction type collected by the statistics counters in Network Gatekeeper for the Parlay X 3.0 Third Party Call-Parlay 3.3 MPCC communication service:

Table 4-2 Transaction types for Parlay X 3.0 Third Party Call - Parlay 3.3 MPCC communication service

Method	Transaction type
makeCallSession	TRANSACTION_TYPE_CALL_CONTROL_SERVICE_INITIATED
transferCallParticipant	TRANSACTION_TYPE_CALL_CONTROL_SERVICE_INITIATED
addCallParticipant	TRANSACTION_TYPE_CALL_CONTROL_SERVICE_INITIATED

Supported Address Schemes

The Parlay X 3.0 Third Party Call-Parlay 3.3 MPCC communication service supports the `tel:` address scheme.

Parlay X 2.1 Call Notification Communication Service

The following chapter describes the Parlay X 2.1 Call Notification communication service in detail:

An Overview of the Parlay X 2.1 Call Notification Communication Service

The Call Notification communication service implements the Parlay X 2.1 Call Notification set of interfaces. For the exact version of the standard this communication service supports, see "Appendix A: Standards and Specifications" in the *Concepts and Architectural Overview*, another document in this set.

Using the Call Notification communication service, an application can:

- Setup and tear down notifications on call events for a given combination of caller and callee
- Receive additional notifications on call events related to the notification in question
- Affect a call during call setup

Note: The operations made available by Call Notification communication service are concerned only with monitoring and in some cases making certain changes to calls. This communication service is not used to setup new calls, only to reroute or terminate calls already in progress.

How It Works

In order for an application to receive notifications about call setup attempts from the network, it must register its interest in these notifications by setting up a subscription in Network Gatekeeper. A subscription, or a notification, is defined by a set of addresses and a set of criteria. The criteria define the events in which the application is interested.

Note: The addresses may possibly be translated by some mechanism in the telecom network prior to reaching Network Gatekeeper.

Two types of notifications exist:

- Simple monitoring
- Monitoring and rerouting

Simple monitoring

An application can register to be notified about the following events as the call between the caller and the callee is set up:

- Callee is busy
- Callee is not reachable
- Callee does not answer
- Call is in progress
- Call setup in progress

Monitoring and rerouting

In addition to monitoring the state of call setup, an application can also choose to make certain changes to the call under certain conditions. An application can:

- Intercept a call setup attempt between the caller and the callee and re-route the call (to a C-party) without making an attempt to connect with callee (B-party). An example might be a general technical support number that is routed to the appropriate call center based on time-of-day.

In addition, if one of the monitored events occurs (busy, not reachable, does not answer), an application can:

- Let further processing of the call be handled by the network
- End the call
- Re-route the call to another callee (C-party.)

Requests (that is, registration for notifications) using the Call Notification communication service flow only in one direction: from the application to Network Gatekeeper. Thus this communication service only supports network triggered requests.

Note: The Call Notification communication service manages only the signalling, or controlling, aspect of a call. The media, or audio, channel is managed by the underlying telecom network. Only parties residing on the same network can be controlled, unless:

- The network plug-in connects to a media gateway controller
- One of the participants is connected to a signalling gateway so that, from a signalling point-of-view, all parties reside on the same network

Supported Network Protocols

Off-the shelf, the Parlay X 2.1 Call Notification communication service can be configured to support the following network protocol:

- SIP

SIP

When Network Gatekeeper is configured to use this protocol it connects to the SIP network. Network Gatekeeper acts as a SIP Back-to-Back-User-Agent. Although from a signalling point-of-view three parties are involved - the caller, the callee, and Network Gatekeeper - from the SIP perspective, the media channel (the actual call) that is established is purely peer-to-peer.

Configuration Specifics for the Call Notification Communication Services

Communication services share many common features, covered in the "Introducing Communication Services" chapter of the *Concepts and Architectural Overview*, but each one has a few characteristics that are specific only to that service. This section describes those specific features for the Parlay X 2.1 Call Notification communication service, including:

- Charging Data Records
- Event Data Records
- Statistics
- Supported Address Schemes

Charging Data Records

Generation of CDRs

The CDRs generated by the Parlay X 2.1 Call Notification to SIP communication service belong to two basic groups. They occur when the following criteria are met:

- After a `notifyBusy`, `notifyCalledNumber`, `notifyNoAnswer`, `notifyAnswer`, or `notifyNotReachable` is sent from the network.
- After a `handleBusy`, `handleCalledNumber`, `handleNoAnswer`, or `handleNotReachable` is called.

Event Data Records

The following are lists of `EdrIds` created by the Parlay X 2.1 Call Notification to SIP communication service. This list does not include EDRs created when exceptions are thrown. For more information, see Appendix A, "Events, Alarms, and Charging.".

Table 5-1 Event types emitted in the Call Notification/SIP communication service

Event data	Description
3000	startCallNotification
3001	stopCallNotification

Table 5-1 Event types emitted in the Call Notification/SIP communication service

Event data	Description
3002	startCallDirectionNotification
3003	stopCallDirectionNotification
8014	notifyBusy
8015	notifyCalledNumber
8016	notifyNoAnswer
8017	notifyNotReachable
8018	handleBusy
8019	handleCalledNumber
8020	handleNoAnswer
8021	handleNotReachable

Statistics

The table below outlines the correlation between the methods being invoked from either the application (in application-initiated requests) or the telecom network (in network-initiated requests) and the transaction type collected by the statistics counters in Network Gatekeeper for the Parlay X 2.1 Call Notification to SIP communication service:

Note: Method names for network-initiated requests are specified by the internal Gatekeeper name, which is not necessarily the same as the message from the network.

Table 5-2 Transaction types for Parlay X 2.1 Call Notification-SIP communication service

Method	Transaction type
notifyBusy	TRANSACTION_TYPE_CALL_CONTROL_NETWORK_INITIATED
notifyNotReachable	TRANSACTION_TYPE_CALL_CONTROL_NETWORK_INITIATED

Table 5-2 Transaction types for Parlay X 2.1 Call Notification-SIP communication service

Method	Transaction type
notifyNoAnswer	TRANSACTION_TYPE_CALL_CONTROL_NETWORK_INITIATED
notifyCalledNumber	TRANSACTION_TYPE_CALL_CONTROL_NETWORK_INITIATED
handleBusy	TRANSACTION_TYPE_CALL_CONTROL_NETWORK_INITIATED
handleNotReachable	TRANSACTION_TYPE_CALL_CONTROL_NETWORK_INITIATED
handleNoAnswer	TRANSACTION_TYPE_CALL_CONTROL_NETWORK_INITIATED
handleCalledNumber	TRANSACTION_TYPE_CALL_CONTROL_NETWORK_INITIATED

Supported Address Schemes

The Parlay X 2.1 Call Notification-SIP communication service supports the `tel:` address scheme.

CHAPTER 6

Parlay X 3.0 Call Notification Communication Service

The following chapter describes the Parlay X 3.0 Call Notification communication service in detail:

An Overview of the Parlay X 3.0 Call Notification Communication Service

This Call Notification communication service implements the Parlay X 3.0 Call Notification set of interfaces. For the exact version of the standard this communication service supports, see "Appendix A: Standards and Specifications" in *Concepts and Architectural Overview*, another document in this set.

Using a Call Notification communication service, an application can:

- Setup and tear down notifications on call events for a given combination of caller and callee
- Receive notifications on call events related to established notifications
- Interact with the functionality of other communication services, including Audio Call to play audio to call participants and/ or collect data from them or Third Party Call to re-route the call or setup additional call legs.
- End the call

Note: The operations made available by the Call Notification communication service are concerned only with monitoring and in some cases making certain changes to calls during the setup phase. By itself, this communication service is not used to setup new calls, only to reroute or terminate calls already in progress.

How It Works

In order for an application to receive notifications about call setup attempts from the network, it must register its interest in these notifications by setting up a subscription in Network Gatekeeper. A subscription, or a notification, is defined by a set of addresses and a set of criteria. The criteria define the events in which the application is interested.

Note: The addresses may possibly be translated by some mechanism in the telecom network prior to reaching Network Gatekeeper.

Two types of notifications exist:

- Monitoring
- Monitoring and rerouting

Monitoring

An application can register to be notified about the following events as the call between the caller and the callee is set up:

- The callee is busy
- The callee is not reachable
- The callee does not answer
- The caller is attempting to call the callee
- The callee has answered the call.

Note: The above notifications may include a `callSessionIdentifier` identifying the call session in the network, if available, to allow interactions with other Parlay X web services, such as Third Party Call and Audio Call. These interactions tend, by nature, to be asynchronous.

- A call participant has interacted with a play and collect media event. The notification contains the results of the interaction, including the digits collected.
- A call participant has interacted with a play and record media event. The notification contains the results of the interaction, including the location of the recorded information.

  Note: Setting up a notification for a play and record media event is supported in Network Gatekeeper version 4.0, but setting up the play and record interaction is *not* supported in the Parlay X 3.0 Audio Call communication service in this version.

Monitoring and rerouting

In addition to monitoring the state of call setup, an application can also choose to make certain changes to the call under certain conditions, in a synchronous manner. In the case of certain monitored events (busy, not reachable, no answer, call attempt), an application can specify how to handle them, including:

- Continue to let the call be managed by the network in the normal manner, by, for example, playing a busy tone.
- End the call
- Intercept the call setup attempt between the caller and the callee and reroute the call to another callee (C-party) without making an attempt to connect with callee (B-party). An example might be a general technical support number that is routed to the appropriate call center based on time-of-day.

  Note: If the call is rerouted, the media type is always negotiated by the underlying network. The MediaInfo parameter is not currently used by the communication service.

Because the Call Notification communication service does handle traffic in two directions (from the application to the network and from the network to the application) its functionality has some aspects of both the “application-initiated” and the “network-initiated” types. It’s important to note that the communication service itself manages only the signalling, or controlling, aspect of the call. The call itself, the media, or audio, channel, is completely handled by the underlying telecom network.

Note: Because the communication service manages only the signalling aspect of the call, only parties residing on the same network can be controlled, unless:

- The network plug-in connects to a media gateway controller
- One of the participants is connected to a signalling gateway so that, from a signalling point-of-view, all parties reside on the same network

Supported Network Protocols

Off-the shelf, the Parlay X 3.0 Call Notification communication service can be configured to support the following network protocol:

- Parlay 3.3 MultiParty Call Control

Parlay 3.3 MultiParty Call Control

When Network Gatekeeper is configured to use this protocol, it connects to a Parlay 3.3 MultiParty Call Control SCS provided by an OSA/Parlay Gateway. See "Appendix A: Standards and Specifications" in *Concepts and Architectural Overview* for the exact version of the standard Network Gatekeeper supports.

Configuration Specifics for the Parlay X 3.0 Call Notification Communication Service

Communication services share many common features, covered in the "Introducing Communication Services" chapter of *Concepts and Architectural Overview*, but each one has a few characteristics that are specific only to that service. This section describes those specific features for the Call Notification communication service, including:

- Charging Data Records
- Event Data Records
- Statistics
- Supported Address Schemes

Charging Data Records

Generation of CDRs

There are three CDRs generated by the Call Notification Parlay X 3.0/Parlay 3.3 MPCC communication service. They occur when the following criteria are met:

- After a `reportNotification` is sent from the Parlay gateway to Network Gatekeeper, indicating that a call event defined by the notification has occurred and (in appropriate cases) needs to be handled.
- After a `sendInfoandCollectRes` has been sent from the Parlay gateway to Network Gatekeeper, indicating that a call participant has interacted with a Play and Collect operation. The response includes the digits collected.

Event Data Records

The following are lists of `EdrIds` created by the Parlay X 3.0 Call Notification/Parlay 3.3 MPCC communication service. This list does not include EDRs created when exceptions are thrown. For more information, seeAppendix A, "Events, Alarms, and Charging."..

Table 6-1 Event types emitted in the Parlay X 3.0 Call Notification/Parlay 3.3 MPCC communication service

Event data	Description
11000	startCallDirectionNotification
11001	stopCallDirectionNotification
11002	startCallNotification
11003	stopCallNotification
11004	startPlayAndCollectNotification
11006	stopMediaInteractionNotification
11007	reportNotification
11008	deleteNotification
11009	createNotification
11011	sendInfoAndCollectRes

Statistics

The table below outlines the correlation between the methods being invoked from either the application (in application-initiated requests) or the telecom network (in network-initiated requests) and the transaction type collected by the statistics counters in Network Gatekeeper for the Parlay X 3.0 Call Notification/Parlay 3.3 MPCC communication service:

Note: Method names for network-initiated requests are specified by the internal Gatekeeper name, which is not necessarily the same as the message from the network.

Table 6-2 Transaction types for Parlay X 3.0 Call Notification/Parlay 3.3 MPCC communication service

Method	Transaction type
reportNotification (both CallNotification and CallDirection)	TRANSACTION_TYPE_CALL_CONTROL_NETWORK_INITIATED
sendInfoAndCollectRes (callNotification only)	TRANSACTION_TYPE_CALL_CONTROL_NETWORK_INITIATED

Supported Address Schemes

The Parlay X 3.0 Call Notification/Parlay 3.3 MPCC communication service supports the `tel:` address scheme.

CHAPTER 7

Parlay X 2.1 Short Messaging Communication Service

The following chapter describes the Parlay X 2.1 Short Messaging communication service in detail:

An Overview of the Parlay X 2.1 Short Messaging Communication Service

The Short Messaging communication service implements the Parlay X 2.1 Short Messaging set of interfaces. For the exact version of the standard this communication service supports, see

"Appendix A: Standards and Specifications" in *Concepts and Architectural Overview*, another document in this set.

Using a Short Messaging communication service, an application can:

- Send Short Messages to one or many destination addresses. The payload in these Short Messages can be text, logos, or ringtones

Note: Logos must be in either SmartMessaging or EMS format. The image is not scaled. Ringtones must be in either SmartMessaging or EMS (iMelody) format

- Ask to be notified that Delivery Receipts for sent Short Messages have been received from the network
- Receive Delivery Receipts on sent Short Messages that have arrived from the network
- Explicitly query Network Gatekeeper for Delivery Receipts on sent Short Messages
- Sign up to be notified if specified Short Messages for the application have been received from the network
- Receive notifications that specified Short Messages for the application have arrived from the network. These notifications include the short message payload
- Explicitly poll Network Gatekeeper for Short Messages sent to the application that have arrived from the network and been stored in Network Gatekeeper.

Requests can flow in two directions using the Parlay X 2.1 Short Message communication service: from the application to the network (called "application-initiated" or "mobile terminated") and from the network to the application (called "network-triggered" or "mobile originated"). Both of these scenarios are covered below.

Processing Application-initiated Requests

Once an application has sent a Short Message to one or more destination addresses, two different types of responses can be returned:

- Send Receipts
- Delivery Receipts

Send Receipts

Send Receipts are merely acknowledgements that the network node has received the Short Message from the application by means of Network Gatekeeper. Although a single Short

Message may be sent to multiple destination addresses, normally only one Send Receipt is returned to the application by Network Gatekeeper. The receipt is returned synchronously in the response message to the `sendSms` operation.

Delivery Receipts

Delivery Receipts contain the delivery status of the short message, that is, whether the short message has actually been delivered by the network to the mobile terminal. There is one Delivery Receipt per destination address, with one of three possible states:

- Successful. In the case of concatenated short messages, this is returned only when all the parts have been successfully delivered. (see the **Note** below)
- Unsuccessful. The short message could not be delivered before it expired.
- Delivery notification for this address is not supported. This can occur if the originating network supports delivery receipts, but is unable to acquire the appropriate information for one or more destination addresses. This status is reported for each address for which this is the case.

Because actual delivery of the short message may take several hours, or even days (if, for example, the mobile terminal is turned off at the time the short message is sent), Delivery Receipts are returned asynchronously. Applications can either choose to have Delivery Receipts delivered to them automatically by supplying Network Gatekeeper with a call-back interface or they can chose to poll Network Gatekeeper.

If the application supplies a call-back interface, there are two possible outcomes:

- Network Gatekeeper sends the Delivery Receipt and the application receives and acknowledges it
- Network Gatekeeper sends the Delivery Receipt but the application does not acknowledge reception. In this case, Network Gatekeeper stores the Delivery Receipt in temporary in-memory storage. The application can poll Network Gatekeeper for these Receipts. Each stored Delivery Receipt is time-stamped and, after a configurable time-period, is removed.

If the application chooses not to supply a call-back interface, Network Gatekeeper stores the Delivery Receipt in temporary in-memory storage. The application can poll Network Gatekeeper for these Receipts. Each stored Delivery Receipt is time-stamped and, after a configurable time-period, is removed.

In order to correlate a sent message with a delivery receipt form the network node, information about the message is stored in Gatekeeper for a period of time. This information has a life-span

If the delivery receipt does not arrive prior to the expiration of the message, a cancel request for the message is sent to the SMSC.

Note: The Short Messaging communication service does put any limitation on the size of the payload in a Short Message. If the network protocol used restricts the size of the payload, Network Gatekeeper splits the Short Message into appropriately sized parts.

Processing Network-triggered Requests

Two sorts of traffic destined for an application can arrive at Network Gatekeeper from the network, including:

- Delivery Receipts for application-initiated sent Short Messages (see Delivery Receipts above).
- Mobile originated Short Messages destined for the application

In order for an application to receive Short Messages from the network, it must register its interest in these Short Messages by setting up a subscription in Network Gatekeeper. A subscription, or notification, is defined by a Service Activation Number, the destination address to which the mobile sender directs the Short Message. This is usually a *Short Code*.

Additional criteria can be tied to the Service Activation Number, such as the first word of the text in the Short Message payload. For Network Gatekeeper to accept a message, both the Service Activation Number and the additional criteria must match the details set up in the subscription. Each registered subscription must be unique, and subscription attempts with overlapping criteria are rejected. The application may choose either to poll Network Gatekeeper for received Short Messages or it may include a call-back interface in setting up the original subscription.

If a Short Message that matches a subscription arrives at Network Gatekeeper from the network and the original subscription includes a call-back interface, there are two possible results:

- Network Gatekeeper sends the Short Message on to the application, and the application receives and acknowledges it. In this case Network Gatekeeper simply acknowledges the reception of the Short Message to the network.
- Network Gatekeeper sends the Short Message on to the application, but the application does not acknowledge reception. In this case, Network Gatekeeper can store the Short Message in temporary in-memory storage, if offline provisioning has occurred and a `registrationIdentifier` has been established. In such a case Network Gatekeeper acknowledges the reception of the Short Message to the network. If no provisioning has been done, Network Gatekeeper returns an error to the network. Each stored Short Message is time-stamped and, after a configurable time-period, is flushed to disk and

removed from in-memory storage. The application can poll Network Gatekeeper for any stored Short Messages.

If a Short Message that matches a subscription arrives at Network Gatekeeper, and the original subscription does not include a call-back interface, the Short Message is stored in temporary in-memory storage and Network Gatekeeper acknowledges the reception of the Short Message to the network. The application can poll Network Gatekeeper for any such Short Messages. Each stored Short Message is time-stamped and, after a configurable time-period, is removed.

If a Short Message arrives at Network Gatekeeper and no matching subscription is found, Network Gatekeeper does not acknowledge reception to the network. It is the responsibility of the network node to handle any further processing of the Short Message.

Note: Short Message communication services do not put any limitations on the size of the payload in a message. If the network protocol used restricts the size of the payload, Network Gatekeeper can concatenate Short Message segments into one single Short Message before delivering it to an application. If configured to concatenate segments, it is also configurable whether to deliver parts of the message to the application regardless if all segments have arrived to Network Gatekeeper or not, and after which time period the segments are considered as lost and the message is propagated to the application.

Supported Network Protocols

Off-the shelf, the Short Messaging communication service can be configured to support the following network protocol:

- SMPP v3.4

SMPP v3.4

When Network Gatekeeper is configured to use this protocol, it connects to an SMSC using SMPP v3.4. See "Appendix A: Standards and Specifications" in *Concepts and Architectural Overview* for the exact version of the standard Network Gatekeeper supports. See the *System Administrator's Guide* for a description of how the plug-in connects to an SMPP v.3.4 compliant SMSC.

Note: SMPP expects the sendName value to be in ASCII characters. The use of non-ASCII characters may cause the request to become garbled or even be removed at the SMSC.

Short Code Translation

A common feature of Messaging capable networks is the use of short codes and message prefixes to help route traffic and to make access to certain features easier for the end user. Instead of having to use the entire address, users can enter shorter sequences when they dial, which are then mapped to the full address in the network. The Parlay X 2.1 Short Messaging to SMPP communication service supports short codes and message prefixes, which allow the same short code to be mapped to multiple addresses, based on what is prepended to the enclosed message.

Configuration Specifics for the Parlay X 2.1 Short Messaging Communication Service

Communication services share many common features, covered in the "Introducing Communication Services" chapter of *Concepts and Architectural Overview*, but each one has a few characteristics that are specific only to that service. This section describes those specific features for the Parlay X 2.1 Short Messaging communication service, including:

- Charging Data Records
- Event Data
- Statistics

Charging Data Records

Generation of CDRs

There are three Short Messaging/SMPP specific CDRs. They occur when the following criteria are met:

- After a `sendSms` is sent from Network Gatekeeper to the network, using either plug-in.
- After a `reportNotification` is sent from the network to Network Gatekeeper, indicating that a Delivery Receipt has been returned for the application, using either plug-in.
- When a Mobile Originated message has been successfully delivered to the application.

Event Data

The following is a list of `EdrIds` created by the Short Messaging/SMPP communication service. This list does not include EDRs created when exceptions are thrown. For more information, see Appendix A, "Events, Alarms, and Charging.".

Table 7-1 Event types emitted in the Short Messaging/SMPP communication service

EdrId	Method Called
6000	notifySmsDeliveryReceipt
6001	notifiySmsReception
7000	sendSms
7001	sendSmsLogo
7002	sendSmsRingtone
7003	startSmsNotification
7004	stopSmsNotification
7005	sendSubmit
7006	receivedSMSDeliveryReport
7007	receivedMobileOriginatedSMS
7008	sendDeliverSMResp
7009	sendSubmitMulti
7010	sendCancel

Statistics

The table below outlines the correlation between the methods being invoked from either the application (in application-initiated requests) or the telecom network (in network-initiated requests) and the transaction type collected by the statistics counters in Network Gatekeeper for the Parlay X 2. Short Messaging/SMPP communication service:

Note: Method names for network-initiated requests are specified by the internal Gatekeeper name, which is not necessarily the same as the message from the network.

Table 7-2 Transaction types for Parlay X 2.1 Short Messaging communication service

Method	Transaction type
sendSms	TRANSACTION_TYPE_MESSAGING_SEND
sendSmsLogo	TRANSACTION_TYPE_MESSAGING_SEND
sendSmsRingtone	TRANSACTION_TYPE_MESSAGING_SEND
receivedMobileOriginatedSMS	TRANSACTION_TYPE_MESSAGING_RECEIVE

Supported Address Schemes

The Parlay X 2.1 Short Messaging/SMPP communication service supports the `tel:` address scheme.

CHAPTER 8

Extended Web Services Binary SMS Communication Service

The following chapter describes the Extended Web Services Binary SMS Communication Service in detail.

An Overview of the EWS Binary SMS Communication Service

The Extended Web Services Binary SMS communication service allows applications to use Short Messaging to send generic binary object attachments, such as vCards. The only operation it supports is `sendBinarySms`. Messages can be sent to one or more destination addresses.

The actual message element is made up of an array of UDH and message parts, encoded in Base64. See 3rd Generation Partnership Project; Technical Specification Group Terminals; Technical realization of the Short Message Service (SMS); (Release 6) 3GPP 23.040 Version 6.5.0, http://www.3gpp.org/ftp/Specs/html-info/23040.htm.

This interface gives an application the flexibility to manipulate the SMPP UDH and message data. Both the UDH and message data elements are optional, but the overall element, `binaryMessage`, is required. The contents of the UDH and the message can be of any binary data, although any byte array should be less than 140 bytes due to SMPP limitations, and the number of BinaryMessage arrays should be less than SegmentsLimit specified in OAM. The default value is 1024.

Send Receipts

Send Receipts are returned to the application synchronously. Send Receipts are merely acknowledgements that the network node has received the Short Message from the application by means of Network Gatekeeper. Although a single Short Message may be sent to multiple destination addresses, normally only one Send Receipt is returned to the application by Network Gatekeeper. The receipt is returned synchronously in the response message to the `sendBinarySms` operation.

Delivery Receipts

Delivery receipt notifications can be set up using this operation, but the actual asynchronous delivery of receipts is accomplished using the Parlay X 2.1 Short Messaging interface. See "Delivery Receipts" on page 7-3 for more information on receiving Delivery Receipts.

Note: The EWS Binary SMS communication service does not support receiving mobile originated messages.

Supported Network Protocols

The Extended Web Services Binary SMS communication service supports the SMPP v3.4 network protocol. When using this protocol, Network Gatekeeper connects to an SMSC using SMPP v3.4. See "Appendix A: Standards and Specifications" in the *Architectural Overview* for the exact version of the standard Network Gatekeeper supports. See the *WebLogic Network Gatekeeper System Administrator's Guide* for a description of how the plug-in connects to an SMPP v.3.4 compliant SMSC.

Note: SMPP expects the sendName value to be in ASCII characters. The use of non-ASCII characters may cause the request to become garbled or even be removed at the SMSC.

Configuration Specifics for the EWS Binary SMS Communication Service

Communication services share many common features, covered in the "Introducing Communication Services" chapter of *Concepts and Architectural Overview*, but each one has a few characteristics that are specific only to that service. This section describes those specific features for the Extended Web Service Binary SMS communication service, including:

- Charging Data Records
- Event Data
- Statistics
- Supported Address Schemes

Charging Data Records

There is one Binary SMS specific CDR. It occurs when the following criterion is met:

- After a `sendBinarySms` is sent from Network Gatekeeper to the network

Event Data

The following is the sole `EdrId` created by the EWS Binary SMS/SMPP communication service. This list does not include EDRs created when exceptions are thrown. For more information, seeAppendix A, "Events, Alarms, and Charging.".

Table 8-1 Event types emitted in the Binary SMS /SMPP communication service

EdrId	Method Called
7101	sendBinarySms

The following is a list of `EdrIds` used by the EWS Binary SMS/SMPP communication service that are inherited from the Parlay X 2.1/SMPP communication service:

Table 8-2 Inherited event types emitted in the Binary SMSBinary SMS /SMPP communication service

EdrId	Method Called
7005	sendSubmit
7009	sendSubmitMulti
7010	sendCancel

Statistics

The table below outlines the correlation between the methods being invoked from either the application (in application-initiated requests) or the telecom network (in network-initiated requests) and the transaction type collected by the statistics counters in Network Gatekeeper for the EWS Binary SMS communication service:

Note: Method names for network-initiated requests are specified by the internal Gatekeeper name, which is not necessarily the same as the message from the network.

Table 8-3 Transaction types for EWS Binary SMS communication service

Method	Transaction type
sendBinarySMS	TRANSACTION_TYPE_MESSAGING_SEND

Supported Address Schemes

The EWS Binary SMS communication service supports the `tel:` address scheme.

CHAPTER 9

Parlay X 2.1 Multimedia Messaging Communication Service

The following chapter describes the Parlay X 2.1 Multimedia Messaging communication service in detail:

An Overview of the Multimedia Messaging Communication Service

The Multimedia Messaging communication service implements the Parlay X 2.1 Multimedia Messaging set of interfaces. For the exact version of the standard this communication service

supports, see "Appendix A: Standards and Specifications" in *Concepts and Architectural Overview*, another document in this set.

Using a Multimedia Messaging communication service, an application can:

- Send Multimedia Messages to one or many destination addresses. The payload in these Multimedia Messages can be any type that can be specified using MIME, including multipart messages
- Sign up to be notified that Delivery Receipts for sent Multimedia Messages have been received from the network
- Receive Delivery Receipts on sent Multimedia Messages that have arrived from the network
- Explicitly query Network Gatekeeper for Delivery Receipts on sent Multimedia Messages
- Sign up to be notified if specified Multimedia Messages for the application have been received from the network
- Receive notifications that specified Multimedia Messages for the application have arrived from the network. These notifications do not include the message payload, but do provide a message ID
- Explicitly poll Network Gatekeeper for Multimedia Messages sent to the application that have arrived from the network and been stored in Network Gatekeeper.

The polling capability must be set up in advance. It is controlled through the combined use of request parameter settings and OAM attributes See Table 9-1 below for more information:.

Table 9-1 Setting up polling functionality

OAM attribute setting: RequestDeliveryReportFlag	Description
0	No polling functionality is available
1	Polling is available if*:* • the application's initial send request includes a notification endpoint **or** • the application includes a tunneled parameter with the value 'com.bea.wlcp.wlng.plugin.multimediamessaging.RequestDeliveryReportFlag' set to 'true' in the initial send request or such a parameter is added during policy evaluation
2	Polling functionality is always available

Requests can flow in two directions using the Multimedia Message communication service: from the application to the network (called "application-initiated" or "mobile terminated") and from the network to the application (called "network-triggered" or "mobile originated"). Both of these scenarios are covered below.

Processing Application-initiated Requests

Once an application has sent a Multimedia Message to one or more destination addresses, two different types of response can be returned:

- Send Receipts
- Delivery Receipts

Send Receipts

Send Receipts are merely acknowledgements that the network node has received the Multimedia Message from the application by means of Network Gatekeeper. Although a single Multimedia Message may be sent to multiple destination addresses, normally only one Send Receipt is

returned to the application by Network Gatekeeper. The receipt is returned synchronously in the response message to the `sendMessage` operation.

Delivery Receipts

Delivery Receipts contain the delivery status of the Multimedia Message, that is, whether the Multimedia Message has actually been delivered by the network to the mobile terminal. There is one Delivery Receipt per destination address, with one of three possible states:

- Successful
- Unsuccessful. The Multimedia Message could not be delivered before it expired.
- Delivery notification for this address is not supported. This can occur if the originating network supports delivery receipts, but is unable to acquire the appropriate information for one or more destination addresses. This status is reported for each address for which this is the case.

Because actual delivery of the Multimedia Message may take several hours, or even days (if, for example, the mobile terminal is turned off at the time the multimedia message is sent), Delivery Receipts are returned asynchronously. Applications can either choose to have Delivery Receipts delivered to them automatically by supplying Network Gatekeeper with a call-back interface or they can chose to poll Network Gatekeeper.

If the application supplies a call-back interface, there are two possible outcomes:

- Network Gatekeeper sends the Delivery Receipt and the application receives and acknowledges it
- Network Gatekeeper sends the Delivery Receipt but the application does not acknowledge reception. In this case, Network Gatekeeper stores the Delivery Receipt in temporary in-memory storage. The application can poll Network Gatekeeper for these Receipts. Each stored Delivery Receipt is time-stamped and, after a configurable time-period, is removed.

If the application chooses not to supply a call-back interface, Network Gatekeeper stores the Delivery Receipt in temporary in-memory storage. The application can poll Network Gatekeeper for these Receipts. Each stored Delivery Receipt is time-stamped and, after a configurable time-period, is removed.

Note: The Multimedia Messaging communication service does not put any limitation on the size of the payload in a Multimedia Message. If the network restricts the size of the payload, Network Gatekeeper does not split the multimedia message.

Processing Network-triggered Requests

Two sorts of traffic destined for an application can arrive at Network Gatekeeper from the network, including:

- Delivery Receipts for application-initiated sent Multimedia Messages (see Delivery Receipts above).
- Mobile-originated Multimedia Messages destined for the application

In order for an application to receive Multimedia Messages from the network, it must register its interest in these Multimedia Messages by setting up a subscription in Network Gatekeeper. A subscription, or notification, is defined by a Service Activation Number, the destination address of the Multimedia Message.

Note: The Service Activation Number may possibly be translated by some mechanism, such as short codes, in the telecom network.

Additional criteria can be tied to the Service Activation Number, such as the start of the first plain/text part in the Multimedia Message payload, or the subject of the Multimedia Message. For the message to be accepted by Network Gatekeeper, both the Service Activation Number and any additional criteria must match the subscription. Each registered subscription must be unique, and subscription attempts with overlapping criteria are rejected. The application may choose either to poll Network Gatekeeper for received Multimedia Messages (if it polling is enabled) or it may include a call-back interface when it sets up the original subscription.

If a Multimedia Message that matches a subscription arrives at Network Gatekeeper from the network and the original subscription includes a call-back interface, there are two possible results:

- Network Gatekeeper sends the notification that the Multimedia Message has arrived on to the application, and the application receives and acknowledges it. In this case Network Gatekeeper stores the Multimedia Message in temporary in-memory storage and acknowledges the reception of the Multimedia Message to the network. Each stored Multimedia Message is time-stamped and, after a configurable time-period, is removed. The application can poll Network Gatekeeper for any stored Multimedia Messages.
- Network Gatekeeper sends the notification that Multimedia Message has arrived on to the application, but the application does not acknowledge reception. Network Gatekeeper does not acknowledge reception to the network. In this case, it is the responsibility of the network node to handle any further processing of the Multimedia Message.

If a Multimedia Message that matches a subscription arrives at Network Gatekeeper and the original subscription does not include a call-back interface, but polling is available, the

Multimedia Message is stored in temporary in-memory storage and Network Gatekeeper acknowledges the reception of the Multimedia Message to the network. The application can poll Network Gatekeeper for any such Multimedia Messages. Each stored Multimedia Message is time-stamped and, after a configurable time-period, is removed.

If a Multimedia Message arrives at Network Gatekeeper and no matching subscription is found and polling is not otherwise enabled, Network Gatekeeper does not acknowledge reception to the network. It is the responsibility of the network node to handle any further processing of the Multimedia Message.

Note: Multimedia Messaging communication services do not put any limitations on the size of the payload in a Multimedia Message. If the network protocol used restricts the size of the payload, Network Gatekeeper does not concatenate Multimedia Message segments into one single Multimedia Message before delivering it to an application. Network Gatekeeper regards these as independent Multimedia Messages.

Supported Network Protocols

Off-the shelf, the Multimedia Messaging communication service is configured to support the following network protocol:

- MM7

MM7

Network Gatekeeper connects to an MMSC using MM7. See "Appendix A: Standards and Specifications" in *Concepts and Architectural Overview* for the exact version of the standard Network Gatekeeper supports.

Short Code Translation

A common feature of Messaging capable networks is the use of short codes and message prefixes to help route traffic and to make access to certain features easier for the end user. Instead of having to use the entire address, users can enter shorter sequences when they dial, which are then mapped to the full address in the network. The Parlay X 2.1 Multimedia Messaging to MM7 communication service supports short codes and message prefixes, which allow the same short code to be mapped to multiple addresses, based on what is prepended to the enclosed message.

Configuration Specifics for the Multimedia Messaging Communication Services

Communication services share many common features, covered in the "Introducing Communication Services" chapter of *Concepts and Architectural Overview*, but each one has a few characteristics that are specific only to that service. This section describes those specific features for the Multimedia Messaging communication service, including:

- Charging Data Records
- Event Data Records
- Statistics
- Supported Address Schemes

Charging Data Records

Generation of CDRs

The Multimedia Messaging/MM7 communication service writes CDRs in the following conditions:

- After a `sendMessage` request has entered the network plug-in from the north
- After a `notifyMessageDeliveryReceipt` has entered the network plug-in from the south
- After a `notifyMessageReception` has been delivered to the application
- When there is an error

Event Data Records

Multimedia Messaging using MM7 uses the standard EDR mechanism, documented in Table 9-2. For more information, see Appendix A, "Events, Alarms, and Charging."

Table 9-2 Event types emitted in Multimedia Messaging/MM7 communication service

EdrId	Meaning
8100	An MO message has arrived from the network
8101	An MO delivery receipt has arrived from the network.
8102	The application has requested that a notification be started
8103	The application has requested that a notification be stopped
8104	The application has polled for a list of received messages
8106	The application has polled for actual messages, returned as attachments.

Statistics

The table below outlines the correlation between the methods being invoked from either the application (in application-initiated requests) or the telecom network (in network-initiated requests) and the transaction type collected by the statistics counters in Network Gatekeeper for the Parlay X 2. Multimedia Messaging/MM7 communication service:

Note: Method names for network-initiated requests are specified by the internal Gatekeeper name, which is not necessarily the same as the message from the network.

Table 9-3 Transaction types for Parlay X 2.1 Multimedia Messaging/MM7 communication service

Method	Transaction type
sendMessage	TRANSACTION_TYPE_MESSAGING_MMS_SEND
deliver	TRANSACTION_TYPE_MESSAGING_MMS_RECEIVE
deliveryReport	TRANSACTION_TYPE_MESSAGING_MMS_RECEIVE

Supported Address Schemes

The Parlay X 2.1 Multimedia Messaging/MM7 communication service supports the `tel:` and `mailto:` address schemes.

CHAPTER 10

Parlay X 2.1 Terminal Location Communication Services

The following chapter describes the Parlay X 2.1 Terminal Location communication services in detail:

An Overview of the Terminal Location Communication Service

The Terminal Location communication services implement the Parlay X 2.1 Terminal Location set of interfaces. For the exact version of the standard these communication services support, see "Appendix A: Standards and Specifications" in *Concepts and Architectural Overview*, another document in this set.

Using a Terminal Location communication service, an application can:

- Ask for the location of one or many terminals by polling
- Ask for the distance between a given terminal and a given position
- Sign up to be notified when a terminal enters or leaves a specified geographical area
- Receive notifications when the terminal enters or leaves the specified geographical area
- Sign up to be notified periodically about the location of a terminal
- Receive periodic location notifications about the location of a terminal

The application can specify a number of parameters vis-a-vis the nature of the notification. These include:

- Requested accuracy
- Accepted accuracy
- Accepted response time
- Maximum age of location data
- Tolerance, which expresses the priority of response time versus accuracy
- Minimum frequency of notifications
- Duration of notifications
- Maximum number of notifications

Ultimately the nature of the information that is available to the application depends on the network node and network protocol used. Not all networks or protocols support all operations. The accuracy of the location provided, the response times, and the frequency of notification are all also dependent on the specifics of the protocol and network node used.

Processing Direct Queries/Application-initiated Requests

If an application directly queries Network Gatekeeper for the location of a terminal or group of terminals, Network Gatekeeper sends the request to the network node, and the location information is sent back synchronously in the response to the request.

Processing Notifications/Network-triggered Requests

If an application registers for periodic or geographically-defined notifications, information for the application (which may or may not include the location data for one or more terminals) arrives at Network Gatekeeper from the network. The notification is passed on to the application. If the application acknowledges the reception of the notification, Network Gatekeeper acknowledges the reception of the notification to the network. If the application does not acknowledge the reception of the notification, Network Gatekeeper does not acknowledges the reception of the notification to the network.

Supported Network Protocols

Off the shelf, the Terminal Location communication service can be configured to support the following network protocols:

- MLP 3.0 and 3.2

MLP 3.0 and 3.2

Network Gatekeeper acts as an LCS/MLS Client. It is possible to use either version of MLP. See "Appendix A: Standards and Specifications" in *Concepts and Architectural Overview* for the exact version of the standard Network Gatekeeper supports.

Network Gatekeeper can be configured to act in Standard Location or Emergency Location Immediate mode on the node level.

Network Gatekeeper connects to the Location Server using HTTP. It always acts as a single LCS/MLS Client towards the Location Server.

Note: When using MLP 3.0, triggered notifications are not supported.

Configuration Specifics for the Terminal Location Communication Services

Communication services share many common features, covered in the "Introducing Communication Services" chapter of *Concepts and Architectural Overview*, but each one has a few characteristics that are specific only to that service. This section describes those specific features for the Terminal Location communication services, including:

- Charging Data Records

Charging Data Records

Generation of CDRs

In the Terminal Location/MLP communication services, CDRs are written:

- When the response to a polling request (of whatever type) is successfully delivered to the application
- When a notification is received from the network
- When an error occurs

Event Data Records

Events for Terminal Location using MLP are documented in Table 10-1. For more information, see *Appendix A, "Events, Alarms, and Charging."*

.

Table 10-1 Event types emitted in Terminal Location/MLP communication services

EdrId	Description
9001	getLocation
9002	getTerminalDistance
9003	getLocationForGroup
9004	sendLocationRequest
9011	LocationEnd
9012	LocationError
9013	LocationNotification

Statistics

The table below outlines the correlation between the methods being invoked from either the application (in application-initiated requests) or the telecom network (in network-initiated requests) and the transaction type collected by the statistics counters in Network Gatekeeper for the Parlay X 2.1 Terminal Location/MLP communication services:

Note: Method names for network-initiated requests are specified by the internal Gatekeeper name, which is not necessarily the same as the message from the network.

Table 10-2 Transaction types for Parlay X 2.1 Terminal Location/MLP communication services

Method	Transaction type
getLocation	TRANSACTION_TYPE_USER_LOCATION
getLocationForGroup	TRANSACTION_TYPE_USER_LOCATION
getLocationDistance	TRANSACTION_TYPE_USER_LOCATION
locationNotification	TRANSACTION_TYPE_USER_LOCATION

Supported Address Schemes

The Parlay X 2.1 Terminal Location/MLP communication services support the `tel:` address scheme.

CHAPTER 11

Parlay X 2.1 Presence Communication Service

The following chapter describes the Parlay X 2.1 Presence communication service in detail:

An Overview of the Presence Communication Service

The Presence communication service implements the watcher aspect of the Parlay X 2.1 Presence set of interfaces. For the exact version of the standard these communication services support, see "Appendix A: Standards and Specifications" in *Concepts and Architectural Overview*, another document in this set.

Presence information is a collection of data on an end user's status, including such things as current activity, environment, available communication means, and contact addressees. Using the Presence functionality, an application can function as a client in two modes: as a watcher or as a presentity. A watcher is a client that is interested in consuming presence information. A presentity

is a client that allows its presence information to be delivered to watchers. WebLogic Network Gatekeeper currently supports the watcher functionality.

The Client as Presence Consumer

An application acting as a watcher can:

- Subscribe to obtain presence data. Each subscription requires authorization by the presentity. The authorization is returned asynchronously via the notification interface.
- Choose to acquire presence information once a subscription has been established using:
  - Direct synchronous polling. This is only effective for a single presentity. Groups are not supported
  - Specific notifications. These can be used for a single presentity. The watcher sets a notification trigger based on certain user presence attribute changes.

    Possible attribute types include:

    - Activity (User's status: Available, Busy, At Lunch, etc.)
    - Place (User's current location: Home, In a Public Place, etc.)
    - Privacy (Degree of privacy the user has: Surrounded by Others, Alone and Can Talk Openly, etc.)
    - Sphere (User's personal status: In his Work Capacity; In his Personal Capacity)
    - Communication Means (Type of communication client preferred: Phone, Email, SMS, etc.)
    - Other (name-value pair for arbitrary information)

    Non-attribute notification parameters can include:

    - Maximum frequency of notifications
    - Duration of time during which notifications should occur
    - Count - the maximum number of notifications
    - Whether status should be checked immediately after notification setup
- End notifications. In this case the subscription to the presentity is retained, but the specific notification is ended
- Receive information that:

  - The initial conditions of the notification setup have been met (count or duration) and this specific setup has been ended
  - The subscription itself has ended

Supported Network Protocols

Off the shelf, the Presence communication service is configured to support the following network protocol:

- SIP

Configuration Specifics for the Presence Communication Service

Communication services share many common features, covered in the "Introducing Communication Services" chapter of *Concepts and Architectural Overview*, but each one has a few characteristics that are specific only to that service. This section describes those specific features for the Presence communication service, including:

- Charging Data Records
- Event Data Records
- Statistics
- Supported Address Schemes

Charging Data Records

Generation of CDRs

There are two Presence/SIP specific CDRs. They are generated when the following criteria are met:

- After the result of a poll for presence is successfully returned to the application
- After a notification for presence information is successfully sent to the application

CDR Details

A Presence CDR contains the standard information with this additional information in the additional_info field for a notification call:

Table 11-1 Additional Info in Presence CDR

Column	Description
additional_info	Endpoint (string)

Event Data Records

The following is a list of `EdrIds` created by the Presence/SIP communication service. This list does not include EDRs created when exceptions are thrown. For more information, see Appendix A, “Events, Alarms, and Charging.”.

.

Table 11-2 Event types emitted in the Presence/SIP communication service

EdrId	Method Called
2000	notifyReceived
2001	makeNotifySubscriptionCallback (includes Endpoint, string)
2002	makeSubscriptionEndedCallback (includes Endpoint, string)
2003	makeStatusChangedCallback (includes Endpoint, string)
2004	makeStatusEndCallback (includes Endpoint, string)
2005	subscribePresence (processes from application)
2006	getUserPresence
2007	startPresenceNotification
2008	endPresenceNotification
2009	subscribePresence (forwards to WLSS)

Statistics

The table below outlines the correlation between the methods being invoked from either the application (in application-initiated requests) or the telecom network (in network-initiated requests) and the transaction type collected by the statistics counters in Network Gatekeeper for the Parlay X 2.1 Presence/SIP communication service:

Note: Method names for network-initiated requests are specified by the internal Gatekeeper name, which is not necessarily the same as the message from the network.

Table 11-3 Transaction types for Parlay X 2.1 Presence/SIP communication service

Method	Transaction type
getUserPresence	TRANSACTION_TYPE_PRESENCE_SERVICE_INITIATED
makeStatusChangedCallback	TRANSACTION_TYPE_PRESENCE_NETWORK_INITIATED

Supported Address Schemes

The Parlay X 2.1 Presence/SIP communication service supports the `tel:` address scheme.

CHAPTER 12

Extended Web Services Subscriber Profile Communication Service

The following chapter describes the Extended Web Services Subscriber Profile communication service in detail:

An Overview of the Extended Web Services Subscriber Profile Communication Service

There is no standard available for this service, although it uses elements from preliminary Parlay X drafts (preliminary as of January 2008) for this functionality. Using the Extended Web Services subscriber profile communication service, an application can:

- Retrieve the specific value for a particular property belonging to a subscriber profile stored in an LDAP data source.
- Retrieve an entire subscriber profile from an LDAP data source, subject to SLA filtering.

How It Works

The Extended Web Services Subscriber Profile communication service can be used by applications that need to retrieve subscriber profile data from an LDAP server attached to the underlying network.

Supported Network Protocols

Off-the shelf, the Extended Web Services Subscriber Profile communication service can be configured to support the following network protocol:

- LDAPv3

LDAPv3

When Network Gatekeeper is configured to use this protocol, it connects to a LDAP server. See "Appendix A: Standards and Specifications" in *Concepts and Architectural Overview* for the exact version of the standard Network Gatekeeper supports.

Configuration Specifics for the Extended Web Services Subscriber Profile Communication Service

Communication services share many common features, covered in the "Introducing Communication Services" chapter of *Concepts and Architectural Overview*, but each one has a few characteristics that are specific only to that service. This section describes those specific features for the Extended Web Services Subscriber Profile communication service, including:

- Charging Data Records
- Event Data Records
- Statistics
- Supported Address Schemes

Charging Data Records

There are two specific CDRs associated with the Extended Web Services Subscriber Profile/LDAP communication service. They occur when the following criteria are met:

- After Network Gatekeeper has returned a full or partial subscriber profile to an application based on one or more attributes requested by that application.
- After Network Gatekeeper has returned a subscriber profile to an application based on the ID of the profile.

Event Data Records

The EDRs produced by the Extended Web Services Subscriber Profile/LDAPv3 communication service are documented below. This list does not include EDRs created when exceptions are thrown. For more information on the contents of standard EDRs, see Appendix A, "Events, Alarms, and Charging.".

Table 12-1 Event types emitted in the EWS Subscriber Profile/LDAP communication service

EdrId	Method Called
13001	get
13002	getProfile

Statistics

The table below outlines the correlation between the methods being invoked from either the application (in application-initiated requests) or the telecom network (in network-initiated requests) and the transaction type collected by the statistics counters in Network Gatekeeper for the EWS Subscriber Profile/LDAP communication service:

Note: Method names for network-initiated requests are specified by the internal Gatekeeper name, which is not necessarily the same as the message from the network.

Table 12-2 Transaction types for EWS Subscriber Profile/LDAP communication service

Method	Transaction type
get	TRANSACTION_TYPE_SUBSCRIBER_PROFILE
getProfile	TRANSACTION_TYPE_SUBSCRIBER_PROFILE

Supported Address Schemes

The EWS Subscriber Profile/LDAP communication service supports the `tel`, `id`, `imsi` and `ipv4` address schemes.

CHAPTER 13

Extended Web Services WAP Push Communication Service

The following chapter describes the Extended Web Services WAP Push communication service in detail:

An Overview of the WAP Push Communication Service

The WAP Push communication service implements the BEA Extended Web Services WAP Push interface. Although the specific interface is not standardized, it uses standardized elements. For more information on these elements, see "Appendix A: Standards and Specifications" in the *Concepts and Architectural Overview*, another document in this set.

Using the WAP Push communication service, an application can:

- Send a WAP Push message
- Send a replacement WAP Push message

- Ask to be notified asynchronously of the status of WAP Push messages that have been sent. The possible values returned include:
  - Rejected: The message was not accepted
  - Pending: The message is in process
  - Delivered: The message was successfully delivered to the end-user
  - Undeliverable: The message could not be delivered because of a problem
  - Expired: The message reached the maximum age allowed by server policy or could not be delivered by the time specified in the push submission
  - Aborted: The mobile device aborted the message
  - Timeout: The delivery process timed out
  - Cancelled: The message was cancelled through the cancel operation
  - Unknown: The server does not know the state of the message

Note: The result notification message is only sent if the initial push submission was accepted for processing. One result notification message is sent per destination address.

Supported Network Protocols

Off the shelf, WAP Push communication services can be configured to support the following network protocols:

- Push Access Protocol (PAP) 2.0

Push Access Protocol (PAP) 2.0

When Network Gatekeeper is configured to use this protocol, the EWS WAP Push communication service supports a subset of its operations including:

- push-message: Submits a message to be delivered. This operation is also used to send a replacement message
- push-response: The response to the push-message operation. This response includes a code specifying the immediate status of the message submission, of the following general types:
  - 1xxx Success: The action was successfully received, understood, and accepted
  - 2xxx Client Error: The request contains bad syntax or cannot be fulfilled
  - 3xxx Server Error: The server failed to fulfil an apparently valid request

  - 4xxx: Service Failure: The service could not be performed. The operation may be retried

- resultnotification-message: Specifies the final outcome of a specific message for a specific recipient. Sent only if the initial request includes the URL to which this notification is to be delivered. Includes both textual indication of state and a status code including the following general types:
  - 1xxx Success: The action was successfully received, understood, and accepted
  - 2xxx Client Error: The request contains bad syntax or cannot be fulfilled
  - 3xxx Server Error: The telecom network node failed to fulfil an apparently valid request
  - 4xxx: Service Failure: The service could not be performed. The operation may be retried
  - 5xxx: Mobile Device Abort: The mobile device aborted the operation.
- resultnotification-response: The response to the result notification. This response includes a code specifying the status of the notification
  - 1xxx Success: The action was successfully received, understood, and accepted
  - 2xxx Client Error: The request contains bad syntax or cannot be fulfilled
- badmessage-response: A response indicating that request is unrecognizable or is of a protocol version that is not supported. This response contains either a 3002 code (`Version not supported`) or a 2000 code (`Bad Request`). In the case of a `Bad Request`, a fragment of the unrecognizable message is included in the response

See "Appendix A: Standards and Specifications" in *Concepts and Architectural Overview*, for the exact version of the protocol standard Network Gatekeeper supports.

Configuration Specifics for the WAP Push Communication Service

Communication services share many common features, covered in the "Introducing Communication Services" chapter of *Concepts and Architectural Overview*, but each one has a few characteristics that are specific only to that service. This section describes those specific features for the WAP Push communication service, including:

- Charging Data Records

Charging Data Records

Generation of CDRs

There are two WAP Push specific CDRs. They are written:

- When the `sendPushMessage` response returns from the network
- When a `sendResultNotificationMessage` response returns from the application

Event Data Records

The following is a list of EdrIds created by the EWS WAP Push communication service. For more information see *Appendix A, "Events, Alarms, and Charging."*

Table 13-1 Event types emitted by the EWS WAP Push Message communication service

EdrId	Meaning
14001	sendPushMessage
14002	sendResultNotificationMessage

Statistics

The table below outlines the correlation between the methods being invoked from either the application (in application-initiated requests) or the telecom network (in network-initiated requests) and the transaction type collected by the statistics counters in Network Gatekeeper for the Parlay X 2. Multimedia Messaging communication service:

Note: Method names for network-initiated requests are specified by the internal Gatekeeper name, which is not necessarily the same as the message from the network.

Table 13-2 Transaction types for EWS WAP Push Message communication service

Method	Transaction type
sendPushMessage	TRANSACTION_TYPE_MESSAGE_SENDER_SEND
sendResultNotificationMessage	TRANSACTION_TYPE_MESSAGE_SENDER_NOTIFY

Supported Address Schemes

The EWS WAP Push Message communication service supports the `tel:` and `wapuser:` address schemes.

APPENDIX A

Events, Alarms, and Charging

Overview

The following chapter describes the features common to the handling of events, charging, and alarms in Network Gatekeeper.

- Events
- Alarms
- Charging Data Records

Events

Events are handled differently in the Access Tier and Network Tier.

Event handling in the Access Tier

The Access Tier runs in the WebLogic Server's Web Services Container, so events or alarms that are raised there can be monitored through standard JMX mechanisms or by using the WebLogic Diagnostics Framework.

See *Developing Manageable Applications with JMX* and *Configuring and Using the WebLogic Diagnostics Framework* for more information on how this works.

Event handling in the Network Tier

In the Network Tier, much of the functionality comes from the interaction between communication services and Network Gatekeeper's Container Services. To capture this specialized level of information, and other pertinent information about the status of the tier, Network Gatekeeper has developed specific mechanisms to record the data.

In standard communication services, all status information generated by the Network Tier - events, alarms, charging data, and usage statistics - begins as an event, which is fired whenever designated methods are called or exceptions are thrown. These events are then sent to the EDR Service. In the EDR Service, events are processed through XML-based filters, which provide the criteria by which the events are classified into types. The filters can also be used to transform the data in the original event, including adding other useful information. Once the information has been processed by the filters, it is delivered to type-specific listeners. Out of the box, there are three types of filters that are all found in tthe file `wlng-edr.xml`. They produce three distinct types of data: *Event Data Records* (EDRs), *Charging Data Records* (CDRs), and *Alarms*. All three of these filters can be customized as desired, using the Administrative Console. These filters can also deliver desired event-based information to external JMS-based listeners. Such listeners are set up as standard JMS topic subscribers and can be anywhere on the network. See the *WebLogic Network Gatekeeper - System Administrator's Guide* for more information on setting up these filters.

Note: For the purposes of backwards compatibility, events, alarms, and charging records can be published and delivered to 2.2 style, as well as standard style, listeners, but this mechanism has been deprecated since version 3.0.

Each EDR always includes:

EdrId	The type of EDR
ServiceName	The service type (SMS, Call Handling, etc.) that produced the event
ServerName	The name of the WLS host
Timestamp	The time at which the event was triggered (in milliseconds from midnight 1 January 1970)
ContainerTransactionID	The transaction ID from WebLogic Server, if available. This identifies the thread on which the request is executed

Class	The name of the class that logged the event
Method	The name of the method that logged the event
Source	The kind of event. There are two possible values for this field: • Method: the event was fired in relation to a method call • Exception: the event was fired in relation to an exception being thrown

In addition, most events include:

Direction	The direction in which the request is traveling. There are two possible values for this field: • **South**: traveling toward the network node • **North:** traveling toward the application
Position	The position of the EDR relative to the method that logged the EDR. There are two possible values for this field: • **Before**: the event occurred before the method • **After**: the event occurred after the method
Interface	The interface at which the EDR is logged. There are three possible values for this field: • **North**: the event was logged at the north plug-in interface • **South**: the event was logged at the south plug-in interface • **Other**: the event was logged someplace other than the north or south interfaces
Exception	The name of the exception that triggered the EDR
SessionId	The application's session identifier
ServiceProviderId	The service provider account identifier
ApplicationId	The application account identifier

AppInstanceGroupId	The authentication user name of the Application Account. This is a string that is equivalent to the 2.2 value: Application Instance Group ID
OrigAddress	The originating address with scheme. For example: tel:12123334444
DestAddress	The destination address. If this is a send list, the first address will be listed here. Additional addresses are stored in the AdditionalInfo field.
AdditionalInfo	Variable information depending on the communication service. Stored as “key=value\n” pairs.
PluginID	The unique id of the plug-in instance.

Alarms

Network Tier alarms are those events that are of immediate interest to the operator. They are EDRs that are defined via filters created in the internal configuration file. While each alarm begins as an EDR, not all the information available in the EDR is stored when the alarm is written to the database (although that information can be retrieved using an external listener). Each alarm entry in the database includes the following information:

alarm_id	A unique sequential identifier
source	The name of the software module that raised the alarm and the IP address of the server in which the module runs. This is *not* the same as the Source field in the event
timestamp	The time at which the event was triggered (in milliseconds from midnight 1 January 1970)
severity	The importance of the alarm. There are four possible values for this field: • 4 for warning • 3 for minor • 2 for major • 1 for critical

identifier	The alarm type
alarm_info	Information provided by the module that raised the alarm
additional_info	This field includes: • Service Provider ID • Application ID • Application Instance ID • Plug-in instance ID • Other information depending on context

Management integration

Network Gatekeeper supports integration of its alarm and event mechanisms with external management tools.

OSS

An Operation Support System (OSS) can integrate with WebLogic Network Gatekeeper alarm and event services through the creation of external JMS listeners. As well, integration can be managed via OAM scripts through the use of JMX-based tools.

SNMP

WebLogic Network Gatekeeper also supports the sending of alarms as SNMP traps to SNMP managers. The alarms sent to the SNMP managers can be filtered on alarm severity.

Charging Data Records

Charging Data Records originate as filtered EDRs. While each CDR begins as an EDR, not all the information available in the EDR is stored when the CDR is written to the database (although that information can be retrieved using an external listener). Each CDR entry in the database includes the following information:

transaction_id	The Network Gatekeeper transaction sequence number
service_name	The communication service whose use is being tracked

service_provider	The Service Provider ID
application_id	The Application ID
application_instance_id	The user name of the Application Account. This is a string that is equivalent to the 2.2 value: Application Instance Group ID
container_transaction_id	The transaction ID from WebLogic Server, if available. This identifies the thread on which the request is executed
server_name	The name of the server in which the CDR was generated
timestamp	The time at which the event was triggered (in milliseconds from midnight 1 January 1970)
service_correlation_ID	An identifier that allows the usage of multiple service types to be correlated into a single charging unit
charging_session_id	An ID correlating related transactions within a service capability module that belong to one charging session. For example, a call containing three call-legs will produce three separate transactions within the same session. **Note:** In installations where sessions are not used, this field contains only a placeholder value.
start_of_usage	The date and time the request began to use the services of the underlying network.
connect_time	The date and time the destination party responded. Used for Call Control traffic only.
end_of_usage	The date and time the request stopped using the services of the underlying network.
duration_of_usage	The total time the request used the services of the underlying network.
amount_of_usage	The used amount. Used when charging is not time dependent, as in, for example, flat rate services.
originating_party	The originating party's address.
destination_party	The destination party's address. This is the first address in the case of send lists, with all additional addresses placed in the additional_info field.

charging_info	A service code added by the application or by policy service.
additional_info	If the communication service supports send lists, all destination addresses other than the first, under the key “destinationParty”. In addition any other information provided by the communication service