Oracle® Communication Services Gatekeeper

Concepts and Architectural Overview

Release 4.0

June 2008

ORACLE

Oracle Communication Services Gatekeeper Concepts and Architectural Overview, Release 4.0

Contents

3. Software Architecture Overview

4. Introducing Communication Services

5. Developing and Testing Applications

6. Managing Application Service Providers

7. Managing Network Gatekeeper: OAM

8. Charging and Billing Integration

9. Redundancy, Load Balancing, and High Availability

10. Service Extensibility

A. Standards and Specifications

CHAPTER 1

Document Roadmap

The following sections describe both the audience for and organization of this document:

Document Scope and Audience

This document provides a high-level account of WebLogic Network Gatekeeper, its structure and capabilities, consisting of:

- an overview of how it works and what benefits it provides
- an explanation of the interfaces it offers third-party application developers, including a description of the tools it furnishes for development and testing
- a summary view of its internals, including:
  - service capabilities
  - OAM mechanisms
  - policy enforcement
  - security

  - billing capabilities
  - integration with PRM/CRMs and OSS tools
  - product extensibility
- a description of supported hardware architecture and components
- a description of software architecture

The document will be of use to third-party application developers who wish to integrate telephony-based functionality into their products and operator-based system developers who wish to extend the functionality of the WebLogic Network Gatekeeper or to integrate it with PRM and/or OSS tools. It will also be of use to system administrators charged with installing and maintaining WebLogic Network Gatekeeper. Managers, support engineers, and sales and marketing people will also find information of value here.

Guide to This Document

The document contains the following chapters:

- Chapter 1, "Document Roadmap." This chapter
- Chapter 2, "Introducing WebLogic Network Gatekeeper." An overview of the benefits Network Gatekeeper provides both application developers and network operators
- Chapter 3, "Software Architecture Overview." A high level look at Network Gatekeeper's internal architecture
- Chapter 4, "Introducing Communication Services." An overview of the communication service functionality
- Chapter 5, "Developing and Testing Applications." The interfaces offered to third-party developers and the tools available to aid in testing and development
- Chapter 6, "Managing Application Service Providers." An overview of the administration model for third-party application service providers
- Chapter 7, "Managing Network Gatekeeper: OAM."The Network Gatekeeper's application management tool and the main Operation, Administration and Maintenance (OAM) tasks. Integrating with OSS
- Chapter 8, "Charging and Billing Integration." Supported charging types. Integrating WebLogic Network Gatekeeper's internal charging mechanism with external billing and settlement systems

- Chapter 9, “Redundancy, Load Balancing, and High Availability.” Fault tolerance, high availability, and load balancing mechanisms from an application and network perspective. Geo-redundancy.
- Chapter 10, “Service Extensibility.” Extending WebLogic Network Gatekeeper by creating modules to support additional application service providers and/or network connectivity interfaces
- Appendix A, “Standards and Specifications.” Detailed description of the standards and specifications supported by Network Gatekeeper’s application-facing interfaces, network facing protocols, and security mechanisms

Terminology

The following terms and acronyms are used in this document:

- Account—A registered application or service provider. An account belongs to an account group, which is tied to a common SLA
- Account group—Multiple registered service providers or services which share a common SLA
- Administrative User—Someone who has privileges on the 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 traceable, expected occurrence in the system, of interest to the operator
- HA —High Availability
- HTML—Hypertext Markup Language
- IP—Internet Protocol
- JDBC—Java Database Connectivity, the Java API for database access
- Location Uncertainty Shape—A geometric shape surrounding a base point specified in terms of latitude and longitude. It is used in terminal location
- MAP—Mobile Application Part
- Mated Pair—Two physically distributed installations of WebLogic Network Gatekeeper nodes sharing a subset of data allowing for high availability between the nodes
- MM7—A multimedia messaging protocol specified by 3GPP
- 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
- Presence Information—A status indicator that conveys the accessibility and the willingness of a potential communication partner.

- Presentity—A supplier of presence information.
- 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
- 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
- SS7—Signalling System 7
- Subscriber—A person or organization that signs up for access to an application. The subscriber is charged for the application service usage. See End User
- SQL—Structured Query Language
- TCP—Transmission Control Protocol
- Communication Service—The data flow of a particular request through WebLogic Network Gatekeeper. Different Service Capabilities use different communication services
- USSD—Unstructured Supplementary Service Data
- VAS—Value Added Service
- VLAN—Virtual Local Area Network

- VPN—Virtual Private Network
- Watcher—A consumer of presence information
- WSDL —Web Services Definition Language
- XML—Extended Markup Language

Related Documentation

This architectural overview is a part of the WebLogic Network Gatekeeper documentation set. The other documents include:

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

Additionally, many documents in the WebLogic Server 10 documentation set are of interest to users of WebLogic Network Gatekeeper, including:

- *Introduction to BEA WebLogic Service and BEA WebLogic Express™*
- *WebLogic Server - Installation Guide*
- *Managing Server Startup and Shutdown*
- *Programming Web Services for WebLogic Server*
- *WebLogic Web Services, Security*

- *Developing Manageable Application with JMX*
- *Configuring and Using the WebLogic Diagnostics Framework*
- *Using WebLogic Clusters*
- *Securing WebLogic Server*

CHAPTER 2

Introducing WebLogic Network Gatekeeper

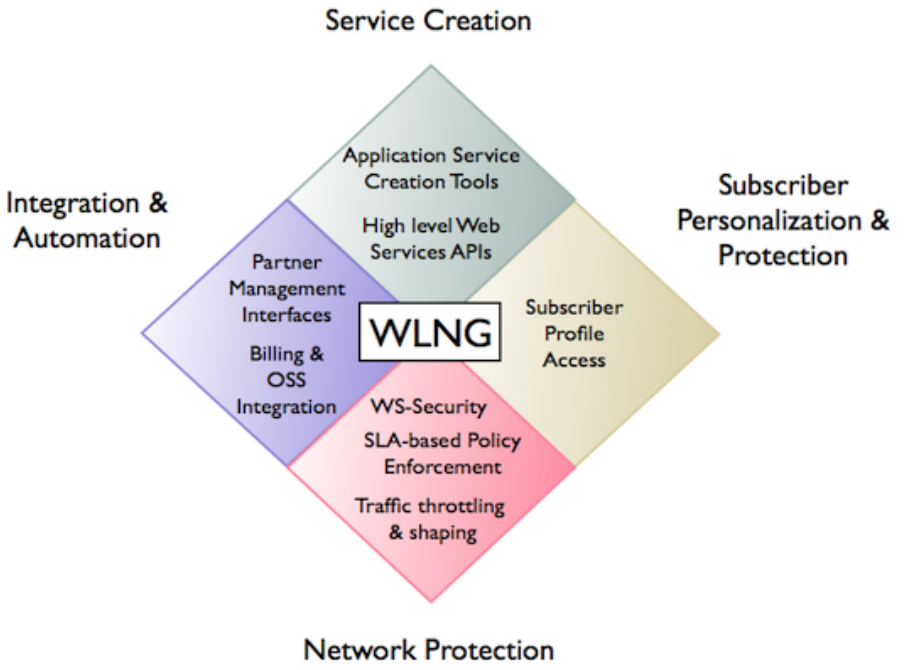
The worlds of TCP/IP applications and of telephony networks continue to converge. Customers want services that provide them with functionality and flexibility that cross the traditional boundaries between the world of the Internet and the world of their phones. Carriers want to be responsive to those desires, and to provide services that will satisfy customer demands, promote subscriber loyalty, increase average revenue per user (ARPU), and drive traffic to their networks.

But developing these services has historically been complex and ungainly. What is needed is a way to reduce the overhead of creating the applications that provide those services, and to make it possible for a wider ranging development community to contribute. To do this, operators need to have a way to:

- Offer simplified access to their network's capabilities, for both internal developers and external partners
- Provide tooling and support for application service development and testing
- Manage external partners efficiently
- Protect the security and stability of the underlying network
- Integrate new services with their existing operations and management facilities
- Protect subscriber privacy and control
- Support flexibility of access as networks change and grow
- Do all this in a way that scales and meets the performance needs subscribers have come to expect

WebLogic Network Gatekeeper has been created specifically to help operators meet these challenges.

Figure 2-1 Network Gatekeeper in Context



What Network Gatekeeper Provides

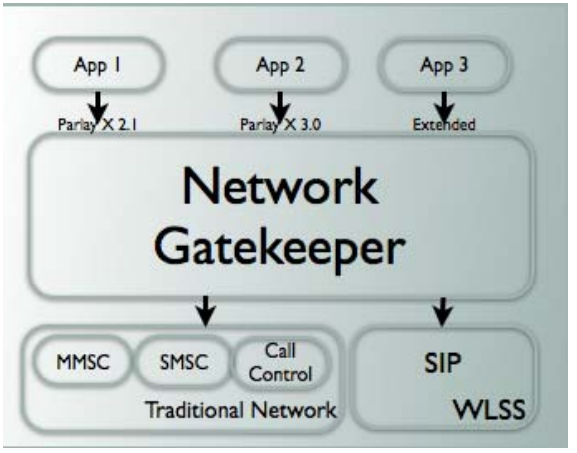
WebLogic Network Gatekeeper, built using version of WebLogic Server 10.0 MP01 that has been hardened and extended to support the specialized needs of telecom networks, offers a host of benefits for both application service developers and network operators.

Access to telecom network service capabilities using APIs based on well-known Web Services standards

The protocols required by underlying telecom network capabilities are often complex, and the learning curve associated with achieving competence in using them is steep. To lower the barriers

to entry for application service developers, out of the box Network Gatekeeper provides access to standard network capabilities such as SMS, MMS or Call Control through a set of easy-to-use Web Services-based interfaces tailored to their needs. These interfaces are largely based on well known standards, both Parlay X 2.1 and 3.0. In cases where access to desired functionality (WAP Push, Binary SMS, and Subscriber Profile) has not yet been incorporated into standardized forms, BEA has created Extended Web Services interfaces. All these interfaces are published in standard WSDL files, so service developers can use their choice of toolsets. Developers can focus on creating compelling and innovative services, while Network Gatekeeper uses its Communication Service components to do the heavy lifting of interacting with the various underlying network elements.

Figure 2-2 Standardized Application Interfaces



Access to WebLogic SIP server for connectivity to SIP network infrastructure

In addition to providing access to traditional telecom network functionality, Network Gatekeeper can also connect application services to SIP-based functionality, using WebLogic SIP Server. Calls set up using the Parlay X 2.1 Third Party Call communication service can be routed through SIP. Parlay X 2.1 Call Notifications can be established using SIP and Parlay X 2.1 Presence *watchers* (consumers of presence information) can be set up.

Application development tools

To assist application service developers, Network Gatekeeper can provide:

- The WebLogic Network Gatekeeper Simulator, which supports early application development cycles without requiring a full-fledged copy of the Gatekeeper
- The WebLogic Network Gatekeeper Simulator GUI, a graphical test and verification environment for SMS, MMS, Terminal Location, and WAP Push traffic that runs on top of the Network Gatekeeper Simulator.
- For developers using WebLogic Workshop, a set of WorkShop Controls,

Network Gatekeeper always provides:

- Web Services WSDL files
- Application Development Guide

Web Service integration for automating partner management

Managing a large number of services, particularly when the providers are third party partners, can be very time and effort intensive. As the market expands, with niche players and short-term services being added to the more mainstream mix, the logistics of on-boarding can become very complex. To assist operators in handling processes such as partner registration, service activation and provisioning, Network Gatekeeper can supply its Partner Relationship Management interfaces. These Web Service interfaces can be used to support the automating of a wide range of partner related tasks, and to provide partners with easily available access to information about their accounts. The interfaces also allow operators to create groups of partners sharing sets of data, which can be used for tiering or segmentation of partners. Operators can then focus their administrative and partner management resources on their most rewarding partners.

Common access control for both internal and 3rd party applications

Network Gatekeeper can function as a single point of contact for access to the functionality of the underlying network, providing common authentication, authorization, and access control procedures for all applications, both internal and third party based. Network Gatekeeper leverages the flexible security framework of WebLogic Server 10 to provide robust system protection. Applications can be authenticated using plaintext or digest passwords, X.509 certificates, or SAML 1.0/1.1 tokens. Service requests can use XML encryption, based on the W3C standard, for either the whole request message or specific parts of it. And, to ensure message integrity, requests can be digitally signed, using the W3C XML digital signature standard.

Flexible authorization control based on fine-grained policy decisions

Network Gatekeeper's powerful and responsive policy enforcement mechanism uses dynamically customizable Service Level Agreements (SLAs) to regulate Service Provider and Application access to particular communication service functionality down to the level of supported operations and parameters. It also supports a range of Quality of Service guarantees that can be modulated by Time of Day/Day of Week, Rates, and Quotas. If desired, further rules covering access can also be added. And both Service Provider and particular Application accounts can be divided into groups to simplify SLA management and maintenance.

In addition, subscriber permissions and preferences can also be reflected in a separate Subscriber SLA, created by the operator or an integrator using tools available in the Platform Development Studio. Subscribers can indicate, for example, that they wish to allow Service Provider X to query for the location of their mobile terminals, but not Service Provider Y.

Enhanced network protection

In addition to the Service Level Agreements that cover access to functionality within Network Gatekeeper itself, further Service Level Agreements explicitly define Service Provider access to underlying network nodes. In conditions of heavy load Network Gatekeeper employs throttling and shaping to protect the underlying network, prioritizing traffic based on these Network SLAs.

Built-in network routing

Network Gatekeeper provides an internal system for the routing of service requests directly to appropriate network nodes, based on address plans and actual destination addresses. Network Gatekeeper supports the as needed in production deployment of multiple instances of most network protocol plug-ins (the module that interacts most directly with the underlying nodes); routing can be managed on a very fine-grained and powerful way.

Carrier grade and fully scalable architecture

Based on WebLogic Server 10 MP01's rock solid performance and superior clustering support, Network Gatekeeper's architecture is designed to support the rigorous demands of telecom operators:

- Tiering:

Network Gatekeeper is deployed in two tiers, which can be separated by a firewall for increased security. State is held only in the Network Tier, and each tier can be built out independently of the other

- High Availability and Failover

  Network Gatekeeper is designed throughout to ensure multi-level protection against single points of failure.

- Geo-Redundancy

  To protect the system in the face of catastrophic failure, geographically distant sites can be set up as site pairs. Service Provider and Application Group SLA enforcement is synchronized across geographic sites and SLAs are enforced between the site pairs.

- Storage Service

  All traffic that passes through Network Gatekeeper is transactionally wrapped. Maintaining state consistently and durably in clustered and high performance environments is traditionally difficult, but Gatekeeper's Storage Service uses a sophisticated strategy of optimizing storage strategies based on state access patterns. An in-memory store distributed among all the nodes serves as the entrance to data access. Reading from disk and its attendant overhead is reduced as the disk-based database functions as an archive rather than as a system of first use. This has two important benefits:

  - Speed: Because the data is available in memory, access is extremely rapid
  - Scalability: As a system scales out, relying exclusively on disk-based database access often becomes a performance bottleneck. Because the data in Network Gatekeeper is distributed among the network tier nodes, adding additional servers to the network tier actually increases data availability.

  In addition, the Storage Service optimizes access to exactly the kinds of data that matter most in telecom traffic processing. Designed around a POJO, java.util.Map-based API, client access is simplified for both storing data and making retrieval queries.

OSS and billing system integration

All or selected parts of the Network Gatekeeper management mechanism can be integrated with an operator's external Operation Support Systems through JMX/JMS or SNMP interfaces. The tasks associated with administering current service providers and adding new ones can be simply folded into existing systems as desired.

Network Gatekeeper's native charging mechanisms can also be integrated with an operator's existing billing systems.

Subscriber personalization and protection

Using Network Gatekeeper, applications can customize their offerings by accessing subscriber profile information stored on network LDAP servers. At the same time, operators can protect subscriber privacy by using filters based on those same profiles to regulate the access that applications have, limiting the information applications can acquire to what the subscriber wants to make available.

In addition, if they choose, operators can define a Subscriber SLA, which creates service provider groupings called service classes that can be associated with individual subscriber URIs. The mechanism to do this is created by the operator or integrator using the Profile Provider SPI provided as part of the Platform Development Studio. This method allows subscribers to customize their interactions with application service providers while keeping all their subscriber data within the confines of the operator's domain.

Extensible architecture

A flexible architecture using the robust capabilities of WebLogic Server 10.0 MP1 means that operators can both extend existing communication services to support new network interfaces, for example Unstructured Supplementary Service Data, and to create entirely new communication services to allow application service developers access to their network's unique features, using Network Gatekeeper's Platform Development Studio.

CHAPTER 3

Software Architecture Overview

The following chapter provides an overview of WebLogic Network Gatekeeper's software architecture, including:

Overview

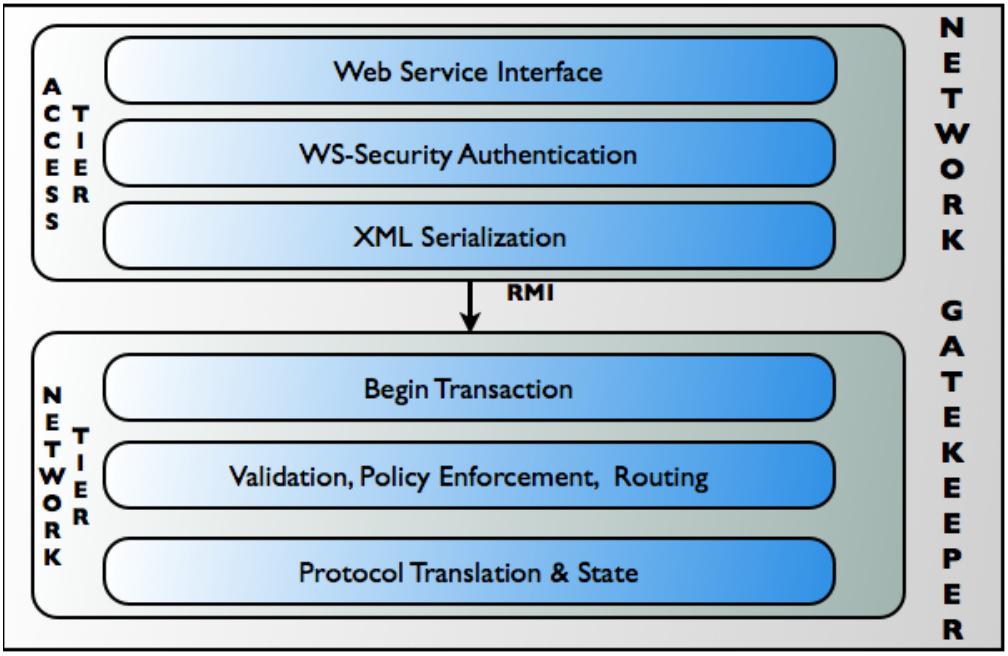
WebLogic Network Gatekeeper provides a robust, secure and highly performant container optimized for the task of running *communication services*. Communication services are specialized components that allow telephony network operators to provide Internet-based application developers with a powerful but simple way to access the operator's network services. Out of the box Network Gatekeeper supplies communication services providing Web Services access to such network capabilities as Messaging, Call Control, Terminal Location, and Presence. Extending the provided communication services or creating entirely new ones, based on the specific needs of the operator's circumstances, is made easy with the supplied Platform Development Studio. Built on WebLogic Server 10.0 MP01, Network Gatekeeper is closely aligned with JEE standards and tightly integrated with WebLogic SIP Server.

- Deployment Model

Communication Services

Communication services are components that run in the Network Gatekeeper container. They provide the main functionality of Network Gatekeeper, exposing network capabilities to Internet based applications in a form that is easy to use and manage. All traffic in Network Gatekeeper is processed through these services. A communication service consists of an application-facing interface, with WS-Security based authentication; a processing layer, where requests are validated, evaluated according to Service Level Agreements, and routed; and a protocol translation layer. A more detailed description of this flow can be found in Chapter 4, "Introducing Communication Services."

Figure 3-1 Communication service structure



Container Services

Network Gatekeeper provides a container that is highly optimized for running communication services, the components of Network Gatekeeper that provide the interface between Internet based applications and the functionality of the underlying telephony network. The container leverages the many standard container services that WebLogic Server 10.0 MP1 provides, but adds a number of services designed for the specialized needs of communication services and Network Gatekeeper generally. See Figure 3-2 and Figure 3-3 below for an example. These services include:

- Core

  Performs initial setup tasks

- Event Channel

  Broadcasts events among modules and servers in the cluster

- Storage

  Provides transparent access to data storage using distributed caching and the database

- Configuration

  Stores largely read-only data, such as configuration information

- CORBA/Orbacus

  Initializes the Orbacus ORB and makes it the default for the system. Can be disabled for systems that do not connect to Parlay Gateways

- EDR

  Broadcasts events and manages their translation into charging data and alarms, as necessary

- Statistics

  Generates system statistics

- Geo-Redundancy

  Provides support for geo-redundant installations

- Budget

Manages cross-cluster bandwidth allocation, and supports geo-redundant installations. In the context of quota and rate SLAs, it also maintains an historical perspective on usage patterns

- Policy

  Wraps the policy engine from older versions, which can still be used

- Plug-in Manager

  Manages the processing layer

- SNMP

  Provides SNMP service for alarms

- Account

  Manages Service Level Agreements and sessions.

Figure 3-2 and Figure 3-3 below show an example of the typical interaction between a communication service (the example uses PX 2.1 Short Messaging to SMPP) and container services.

Figure 3-2 Container Services in Typical Application-Initiated Traffic

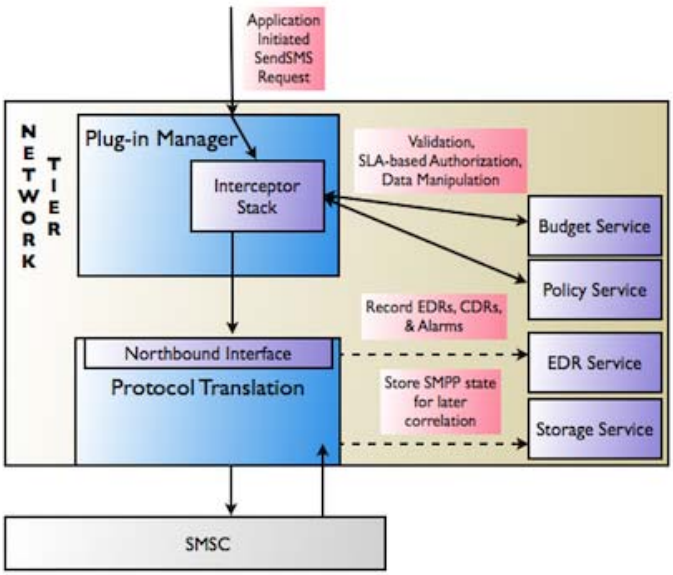
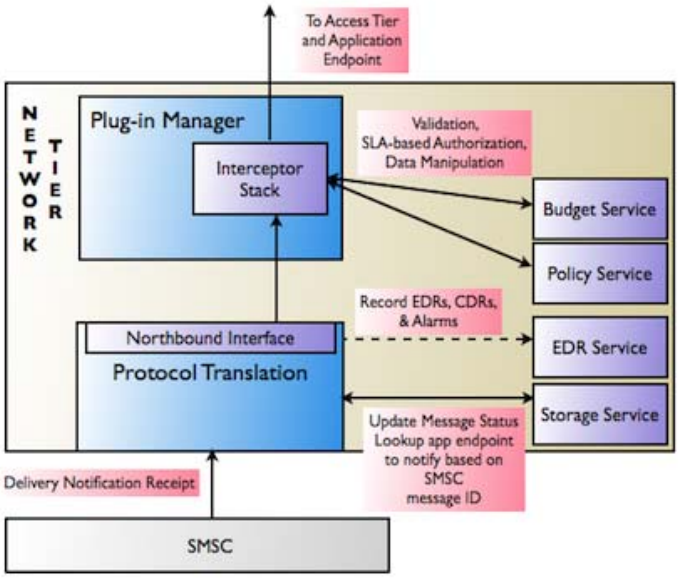


Figure 3-3 Container Services In Typical Network Triggered Traffic



Deployment Model

In production mode, communication services are typically deployed in two clustered tiers, an Access Tier and a Network Tier, separated, if desired, by a firewall. In a single physical site installation, this corresponds to a single WebLogic Server administration domain. Each communication service is deployed in its own EAR file, one per tier.

Note: Some EARs may contain either multiple application-facing interfaces (Parlay X 2.1 Short Messaging and Binary SMS/SMPP) or multiple network plug-ins (Parlay X 2.1 Third Party Call/SIP and INAP) that support the same basic service capability.

Single communication services can be installed or removed without having an impact on other communication services. If no interfaces are changed, existing communication services can be upgraded while traffic is running. This process is called a “hitless upgrade” and tracks traffic so that in-flight requests can be completed before the older version is undeployed. Communication services may be deployed selectively, as needed.

CHAPTER 4

Introducing Communication Services

The following chapter presents a high level introduction to Network Gatekeeper's communication services, including:

- Overview
  - How they work
  - Typical Application-initiated Traffic Flow
  - Typical Network-triggered Traffic Flow
- Platform Features
  - Service Level Agreements and Policy Enforcement
  - Service Level Agreements and Network Protection
  - Events, Alarms, and Charging
  - Statistics and Transaction Units (Licensing)

A separate document, the *Communication Service Reference*, offers a more detailed look at specific services.

Overview

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

How they work

Network Gatekeeper is deployed into two tiers: the Access Tier, which manages interactions with applications, and the Network Tier, which contains the mechanisms necessary for dealing with the underlying network nodes. For increased security, these tiers can be separated by a firewall.

Application-initiated requests (also called mobile terminated, or MT) enter through the Access Tier, where they are processed by the WLS Web Services Container. They are then sent on to the Network Tier using RMI. The Network Tier manages service authorization and policy enforcement, charging, and traffic throttling and shaping. Then the Network Tier translates the request into a form appropriate for the underlying network node.

Note: Although the operator may choose instead to run in a "sessionless" mode, by default, Network Gatekeeper requires that applications acquire a Network Gatekeeper session before beginning to send request traffic. Applications do this using the Session Manager interface, which returns a Session ID. The application then adds this Session ID to the header of all its subsequent SOAP requests. Network Gatekeeper can use this value to keep track of all the traffic that an application sends for the duration of the session.

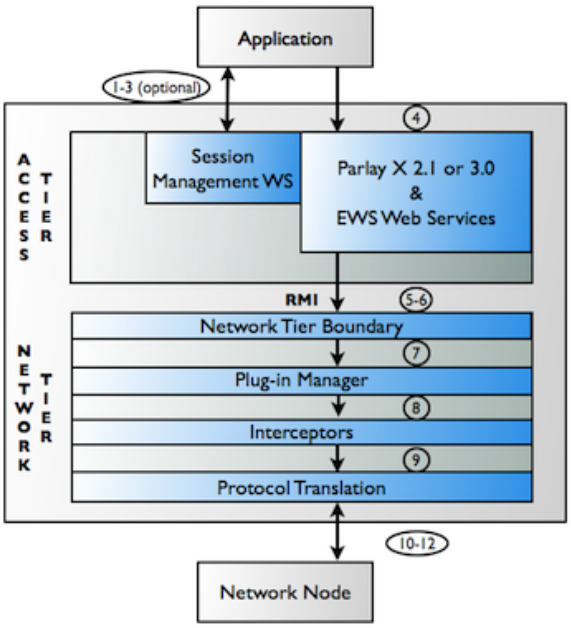
Network-triggered (also called mobile originated, or MO) traffic is also supported by Network Gatekeeper, so that applications can choose to receive data from the telecom network. To do so, the application must first send a request to Network Gatekeeper (or have the operator perform the equivalent task using OAM methods) to register a description of the types of data it is interested in - delivery notifications, incoming messages, etc. - and any criteria that the data must be meet to be acceptable. For example, an application might specify that it is only interested in receiving incoming SMSes that are addressed to the *short code* "12345" and that begin with the string "blue".

Note: For more on short codes, see the "Parlay X 2.1 Short Messaging Communication Service" chapter in the *Communication Services Reference*, another document in this set.

Typical Application-initiated Traffic Flow

Figure 4-1 below illustrates typical application-initiated traffic flow.

Figure 4-1 Typical Application-initiated Traffic Flow



1. *Steps 1-3 are optional and may be turned off.* An application establishes a session by using the Session Management Web Service in the Access Tier.

2. A session is established, and the SessionID is returned to the application. Once the application has been established, it may access multiple communication services across the cluster transparently.

3. The session is valid until the application terminates it or an operator-established time period has elapsed.

Note: Sessions allow correlation among sequences of operations. They are not used for purposes of authentication.

4. A request for a particular operation, usually transported over SSL, enters at the application-facing interface in the Access Tier. This interface is implemented as a a JEE Web

Service. The request is authenticated using WebLogic Server's WS-Security, which supports plaintext or digest passwords, X.509 certificates, or SAML tokens.

Note: All requests are authenticated in this manner, whether the application uses the session mode or not.

In addition, the request may be further secured through encryption using the W3C's standard XML encryption and through digital signatures using the W3C XML digital signature standard. The particular security requirements of the installation are specified in the WS-Policy section of the published WSDL file.

Note: It is possible to simply use the appropriate standard Parlay X 2.1 or 3.0 WSDL to create requests, but the developer would then be required to ascertain the appropriate security type from the operator, and insert the information manually.

5. The request XML is serialized and is passed on to the Network Tier using RMI.
6. The entrance point for the Network Tier marks the beginning of the application-initiated transaction.
7. The request is sent to the Plug-in Manager.
8. The Plug-in Manager invokes the Interceptor Stack to evaluate the request. The Interceptor Stack is a flexible set of chained evaluation steps that:
    - Validates the request
    - Enforces a range of policy decisions based on service level agreements (and, possibly, additional rules)
    - Performs any necessary data manipulation
    - Routes the request to an appropriate protocol translation module

    Note: Should a request fail because of an unavailable module, an interceptor retries the request using one of the remaining eligible modules.
9. The request is sent to the network plug-in to be translated into the protocol suitable for the underlying network node. All state information required by the underlying network node is stored within the network plug-in.
10. The request is passed to the network.
11. When the node acknowledges the request, charging data about the completed request are recorded.
12. The transaction commits.

Typical Network-triggered Traffic Flow

The key difference between application-initiated traffic flow and network-triggered traffic flow (other than the direction) is that the application must first indicate to Network Gatekeeper that it is interested in receiving traffic from the network. It does this by *registering for (*or *subscribing to) notifications*, either by sending a request to Network Gatekeeper or by having the operator set up the notification using OAM methods. For example, the application could send Network Gatekeeper a request to begin receiving SMSes from the network, indicating that it is only interested in SMSes that are sent to the address `12345` and that begin with the string `blue`.It also indicates the URL of the Web Service that the application has implemented to receive these notifications back.

This registration for notifications is stored in the appropriate network plug-in, which in most cases passes it on to the underlying network node itself (in certain cases the Network Gatekeeper operator must do this manually.) When a matching SMS reaches the plug-in from the network, the plug-in sends it to the Plug-in Manager, which invokes the Interceptor stack for evaluation. Then, using RMI, the final Interceptor passes the notification, along with the appropriate URL from the registration, to the Access Tier, which sends it on to the application. See Figure 3-2 and Figure 3-3 for more information on the general flow, although those figures document delivery notifications.

Platform Features

Some functionality is common to all communication services. This functionality includes:

- Service Level Agreements and Policy Enforcement
- Service Level Agreements and Network Protection
- Traffic Security
- Events, Alarms, and Charging
- Statistics and Transaction Units (Licensing)

Service Level Agreements and Policy Enforcement

All application access to WebLogic Network Gatekeeper's communication services is governed by a set of Service Level Agreements (SLAs) between the application service provider and the Network Gatekeeper operator. Network Gatekeeper uses a two tiered account system to group application services and their providers and to simplify the creation and maintenance of SLAs:

- Service Provider Group
- Application Group

For more information on the account system, see The Administration Model.

These SLAs define whether a member of a service provider group or application group:

- Has access to a particular communication service, which can be regulated down to supported methods and parameters
- Participates in any Quality of Service (QoS) agreements such as:
  - Specifying the guaranteed number of requests a service provider may send through a particular communication service in a given period of time. These guarantees may be modulated by:
    - Time of Day/Day of Week
    - Rate (Invocations per time period)
    - Quota (Aggregated number of invocations)

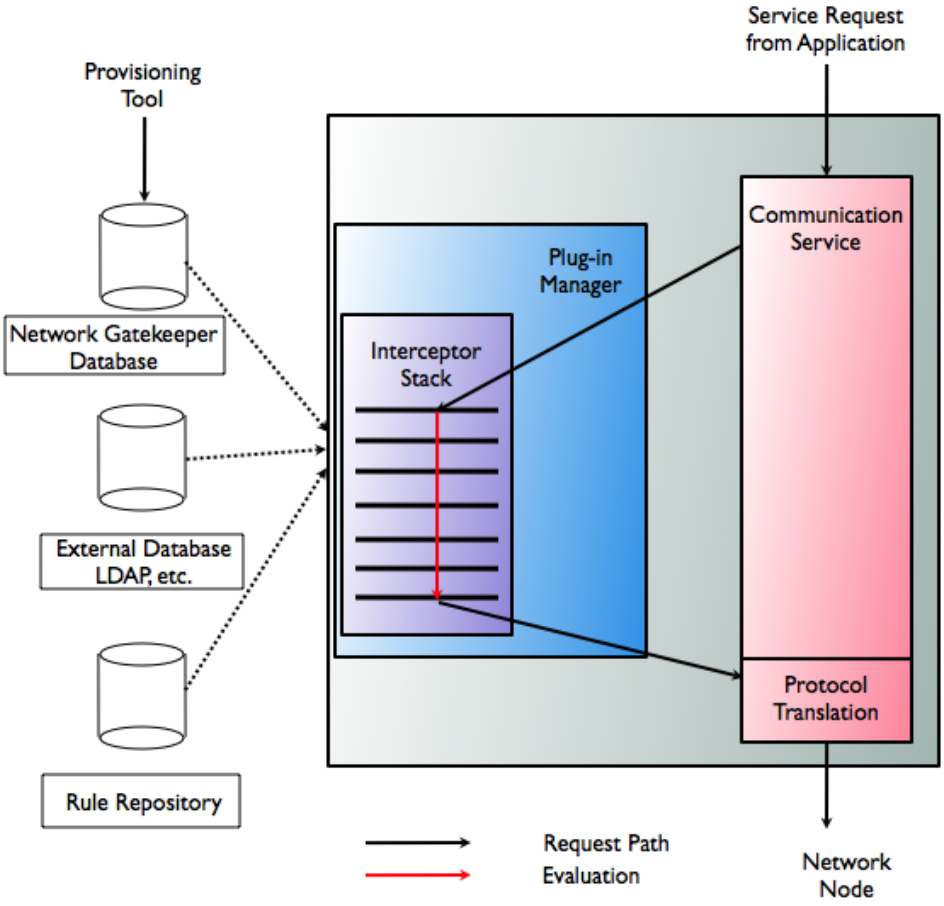
For a more detailed look at Service Provider and Application SLA structure, see the "Defining Service Provider Level and Application Level Service Agreements" chapter in *Managing Accounts and SLAs.* For a communication service-focussed description, see the respective communication service chapters in the *Communication Service Reference.* These books are separate documents in the Network Gatekeeper documentation set.

SLA enforcement for communication services is provided by the Interceptor stack. As in previous versions, it is also possible to create extended *rules* that are evaluated using the external policy engine, which is called from an interceptor.

Note: These rules represent operator specific policies defined by the operator and implemented by BEA Systems or a selected partner.

A simplified version of the flow is illustrated in Figure 4-2 below.

Figure 4-2 Simplified communication service policy execution flow



Network-triggered requests are also evaluated using the Interceptor stack.

Service Level Agreements and Network Protection

There are also SLAs that help protect the underlying network node by setting priorities for sending requests, on the level of a particular service provider group or of Network Gatekeeper as a whole. Depending on the status of the underlying network, traffic can be throttled and shaped. If a particular node is overloaded, lower priority traffic can be rejected all together. For general

information on these Traffic (sometimes called Node) SLAs, see The Administration Model. For more detailed information, see the "Defining Global Node and Service Provider Group Node SLAs" chapter in *Managing Accounts and SLAs.*

Traffic Security

Network Gatekeeper uses special SOAP headers to authenticate service provider applications. These headers are documented in each interface's WSDL file, in the WS-Policy section. Processing is managed by WebLogic Server's WS-Security, which supports plaintext or digest passwords, X.509 certificates, or SAML tokens for authentication. To guarantee the confidentiality of communication between WebLogic Network Gatekeeper and the application, all traffic can be encrypted - fully or partially - using W3C's standard XML encryption. Message integrity can be assured using the W3C XML digital signature standard. Again, the WS-Policy section of the published WSDL for each interface describes if and how either of these standards is being used. For more information on WebLogic Server's capabilities, see the "Configuring Message Level Security" chapter in *WebLogic Web Services: Security*, a separate document in the WLS set. Access to a particular communication service is based on the two types of SLAs discussed in Service Level Agreements and Policy Enforcement.

In addition, if the underlying network node provides an authentication interface, WebLogic Network Gatekeeper protocol plug-ins can register and be authenticated with it, making the request's transfer to the network secure.

Note: This is highly dependent on the protocol and the specific implementation in the node and the plug-in.

Events, Alarms, and Charging

All WebLogic Network Gatekeeper modules can produce general events, alarms and charging events. General events are expected system occurrences that are of importance to the operator, but do not need corrective action. Alarms are system occurrences that are unexpected and may require corrective action. Charging events are the basis for CDRs, the records that provide the information needed to charge for services. CDRs are written only when the transaction that brackets the request's flow through the Network Tier commits. For more information on events and charging, see "Events, Alarms, and Charging" in the *Communication Service Reference*. For more information on alarms, see *Handling Alarms*.

Statistics and Transaction Units (Licensing)

Licensing for WebLogic Network Gatekeeper is based on a maximum allowed rate (measured in *transaction units per second* or TUPS) during a specific time period per 24-hour interval. Two TUPS rates are measured: Base Platform - the more general rate - and BEA Module - which covers only Network Gatekeeper-supplied communication services. For more information on how these rates are calculated, see *Licensing*, a separate document in this set.

CHAPTER 5

Developing and Testing Applications

The following sections introduce developing client applications to interact with WebLogic Network Gatekeeper:

- Overview
- References
- SDK

Overview

Network Gatekeeper provides client application developers with two types of easy to use Web Services APIs: those based on the Parlay X 2.1 and 3.0 standards and three additional ones to cover Binary SMS, Subscriber Profile, and WAP Push functionality, which are not supported by Parlay X. These interfaces include:

- Third Party Call (Parlay X 2.1 and 3.0)

  Using the communication services based on these interfaces, an application can set up a call between two parties (the caller and the callee), poll for the status of the call, and end the call. In addition, applications that use the Parlay X 3.0 based communication service can also add or delete additional parties, transfer call participants, get the call session information associated with a call participant, and interact with certain other communication service functionality, using Audio Call (play media to one or more parties and/or collect data from them) and Call Notification (respond to call event notifications previously established).

- Audio Call (Parlay X 3.0)

Using the communication service based on this interface, an application can play audio to one or more call participants in a call session that was set up using the Parlay X 3.0 Third Party Call service. It is also possible to collect digits from the participant in response to the audio, which can be delivered to the application using the Parlay X 3.0 Call Notification communication service.

- Call Notification (Parlay X 2.1 and 3.0)
  Using the communication services based on these interfaces, an application can set up and end notifications on call events, such as a callee in a third party call attempt being busy. If desired, the application can then reroute the call to another party. In addition, the Parlay X 3.0 communication service can work in concert with certain other communication service functionality, using Third Party Call or Audio Call.

- Short Messaging (Parlay X 2.1)
  Using the communication service based on this interface, an application can send SMS text messages, ringtones, or logos to one or multiple addresses, set up and receive notifications for final delivery receipts of those sent items, and arrange to receive SMSes meeting particular criteria from the network.

- Binary SMS (EWS)
  Using the communication service based on this interface, an application can send a generic binary object (for example, a vCard) using SMS mechanisms. This interface is not based on the Parlay X standards, but instead belongs to the BEA Extended Web Services set. This interface is used only for sending the object. To receive delivery receipts, an application uses the Parlay X 2.1 Short Messaging interface.

- Multimedia Messaging (Parlay X 2.1)
  Using the communication service based on this interface, an application can send Multimedia Messages to one or multiple addresses, set up and receive notifications for final delivery receipts of those sent items, and arrange to receive MMSes meeting particular criteria from the network.

- Terminal Location (Parlay X 2.1)
  Using the communication services based on this interface, an application can request the position of one or more terminals or the distance between a given position and a terminal. It can also set up and receive notifications based on geographic location or time intervals.

- Presence (Parlay X 2.1)
  Using the communication service based on this interface, an application can be a *watcher* for presence information published by a *presentity*, an end user who has agreed to have certain data, such as current activity, available communication means, and contact addresses, made available to others. So a presentity might say that at this moment he is in the office and prefers to be contacted by SMS at this number. Before the watcher can

receive this information, it must subscribe and be approved by the presentity. Once this is done, the watcher can either poll for specific presentity information, or set up status notifications based on a wide range of criteria published by the presentity.

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

- Subscriber Profile (EWS)
  The application-facing interface of this communication service is based on a subset of that in a proposed Parlay X version. Using the communication service based on this interface, an application can retrieve either individual properties associated with a subscriber profile record stored in an LDAP data source in the underlying network or entire profiles from that data source.

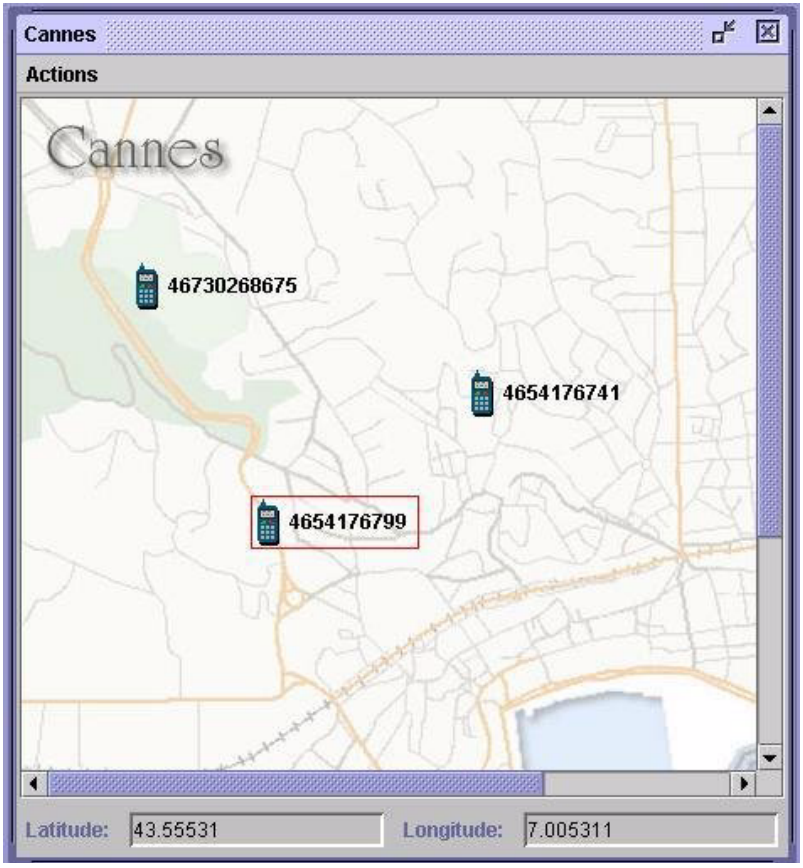
References

WebLogic Network Gatekeeper ships with the *Application Development Guide*, which covers both the APIs themselves and some additional information an application developer needs to create applications that work with Network Gatekeeper. Because all of the APIs are Web Services based and WSDLs are supplied, applications can be developed using any environment that the developer chooses.

SDK

As an option, application developers for WebLogic Network Gatekeeper can also access a set of tools created to ease the development process, the WebLogic Network Gatekeeper SDK, including Network Gatekeeper Simulator and a network simulator. Applications use the interfaces on the Simulator just as they would on a Network Gatekeeper instance running on a telecom network. In addition, there is a GUI-based testing environment that runs on the Simulator for applications that are developing MMS, SMS, WAP Push, and Terminal Location functionality. See Figure 5-1 below.

Figure 5-1 WebLogic Network Gatekeeper Simulator Messaging and Location GUI



The Messaging and Location testing interface of the Simulator consists of a GUI which displays a map. The map can be changed to represent different geographical areas. Mobile terminals representing the application's end users are added to the map and given a phone number. These terminals can then be used as testing targets, sending and receiving messages, and querying for location. Once the terminals have been defined, they can be moved to different locations on the map.

Also included with the Simulator is a developer's copy of WebLogic Server, the environment in which the Simulator runs. For more on what is needed to use the SDK and run the Simulator, see the *SDK User Guide*, a separate volume in this document set. For those developers who are using

WebLogic Workshop as their development environment, the SDK also ships with a set of pre-defined controls.

CHAPTER 6

Managing Application Service Providers

The following sections describe the framework for managing service providers and applications:

- Overview
- The Administration Model
- Partner Relationship Management Interfaces
- Other Tasks Associated with Administering Service Providers

Overview

Managing partner relationships is key to the successful convergence of third-party application services and telecom network operations. WebLogic Network Gatekeeper provides a partner administration model to help operators handle the needs and demands of their partners in a flexible and powerful way:

- Application service providers are registered with WebLogic Network Gatekeeper, by service provider account and application account.
- Each account type is associated with a group that is tied to a Service Level Agreement that defines its access to both Network Gatekeeper and underlying network nodes.

The service provider and application registration are performed either internally through the WebLogic Network Gatekeeper Management Console or through external management systems integrated with WebLogic Network Gatekeeper using the Network Gatekeeper Partner Relationship Management Interfaces.

The Administration Model

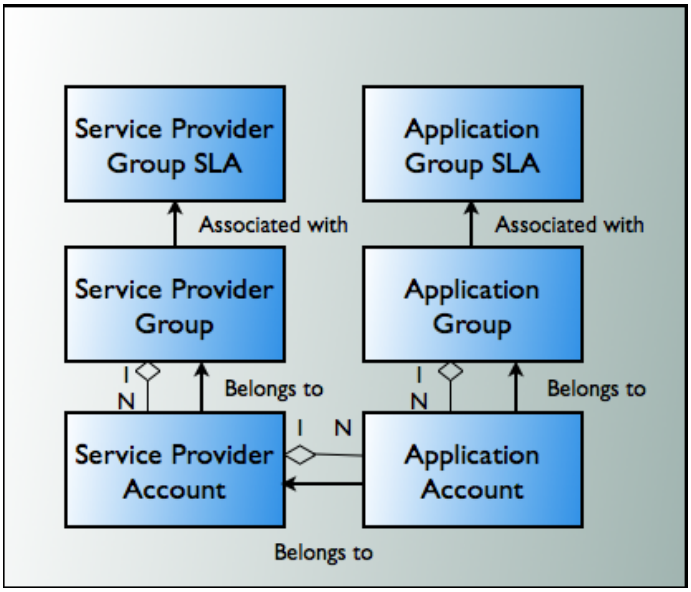
The WebLogic Network Gatekeeper administration model allows operators to manage application service provider access at increasingly granular levels of control. An application service provider registers with WebLogic Network Gatekeeper and is given a service provider account. To support tiering, service provider accounts are associated together into account groups. These groups are then associated with their own Network Gatekeeper Service Level Agreements.

Within a service provider account are individual application accounts, registered on their respective service provider accounts. As in the case of service provider accounts, these application accounts are grouped together into account groups, each of which is associated with its own SLA.

Network Gatekeeper SLAs on the service provider and application level regulate, for example, the type of service capability made available and the maximum bandwidth use allowed. They may also specify access to charging capabilities and revenue sharing schemas. See Figure 6-1 for more information.

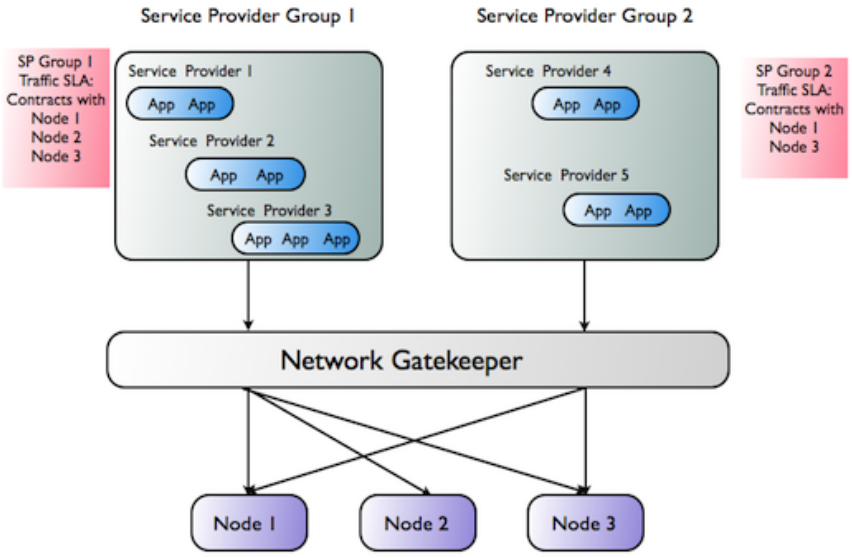
Note: Using the Platform Development Studio, integrators can extend this model to include subscribers as well. For more information, see the *Platform Development Studio - Developer's Guide*, a separate document in this set.

Figure 6-1 Service Provider and Application Administration Model



In addition to Network Gatekeeper SLAs, WebLogic Network Gatekeeper supports two types of traffic SLAs, contracts designed to protect the underlying telecom network. Service provider traffic SLAs regulate the relationship between a service provider group and the network nodes to which it has access. See Figure 6-2 for more information.

Figure 6-2 Service Provider Traffic SLAs



In Figure 6-2 above, service providers in service provider group 1 are allowed to access all network nodes, since their service provider traffic SLA (valid for all service providers within the group) contains node contracts for all nodes.

Service providers in service provider group 2 are only allowed to access network node 1 and 3, because their service provider traffic SLA only contains node contracts for node 1 and 3.

The second type of traffic SLA, the total traffic SLA, regulates the relationship between the Network Gatekeeper itself and the underlying nodes.

Partner Relationship Management Interfaces

The WebLogic Network Gatekeeper Partner Relationship Management Interfaces provide support for the automation of the traditionally work intensive tasks related to service provider and application administration, including supporting on-boarding workflows, using request/approve. Most of the work of registration can be shifted to the service provider, as the operator's role changes from that of entering registration data to that of approving registration data. Large numbers of service provider and application accounts can be managed without increasing

administration overhead. Service providers are also provided with a defined and structured channel to communicate desired account changes and to retrieve usage statistics for the accounts.

For a detailed description of the Partner Relationship Management Interfaces, see the document *Integration Guidelines for Partner Relationship Management for WebLogic Network Gatekeeper*.

As a part of an integration project, the Partner Relationship Management Interfaces can also be integrated with back end systems and network nodes, such as SMSCs, MMSCs, and pre-paid systems for creating and updating accounts.

Other Tasks Associated with Administering Service Providers

For an application to use Audio Call-based services, announcements must be recorded and installed in the network. For more information on these areas, see the *System Administrator's Guide*.

CHAPTER 7

Managing Network Gatekeeper: OAM

The following sections describe Operation, Administration, and Maintenance (OA&M) functionality for WebLogic Network Gatekeeper

- Overview
- The WLS and Network Gatekeeper Management Console
- OAM Tasks Overview
- OSS Integration

Overview

WebLogic Network Gatekeeper is usually controlled through the WebLogic Network Gatekeeper Management Console, a specialized extension of the general WebLogic Server Console. The Console is a web-based tool, and can be run in any environment that supports appropriate web browsers. For general information on the WLS Console, see the "Overview of the Administration Console" chapter of the *Introduction to BEA WebLogic Server and BEA WebLogic Express*™. For some tasks, you can also use scripts that run in the WebLogic Scripting Tool. For general information, see *WebLogic Scripting Tool.* In addition, all or selected parts of the management application can be integrated with external Operation Support Systems (OSS) using JMX/JMS and alarms can be distributed using SNMP traps. Finally, the application service provider management tool functionality can be integrated with PRM and CRM systems using the Network Gatekeeper Partner Relationship Management Interfaces.

Administrative users can be divided into user groups with access to different aspects of the administrative functionality. Within user groups, individual users can have differing levels of access. See the *System Administrator's Guide* for more information.

The WLS and Network Gatekeeper Management Console

The BEA WebLogic Network Gatekeeper Management Console is a Web browser-based, graphical user interface that you use to manage a WebLogic Server domain. A standard production installation for WebLogic Network Gatekeeper consists of at least one WebLogic Server domain.

One instance of WebLogic Server in each domain is configured as an Administration Server. The Administration Server provides a central point for managing a WebLogic Network Gatekeeper domain. All other server instances in the domain are called Managed Servers. In Network Gatekeeper, they are divided into Access Tiers and Network Tiers. In a domain with only a single WebLogic Server instance, such as a development environment, that server functions both as Administration Server and both Managed Servers. The Administration Server hosts the Administration Console, which is a Web application accessible from any supported Web browser with network access to the Administration Server. To access the console, use the following URL:

```
http://hostname:port/console
```

where `hostname` is the DNS name or IP address of the Administration Server and `port` is the listen port on which the Administration Server is listening for requests.

OAM Tasks Overview

Use the Administration Console to:

- Configure, start, and stop Network Gatekeeper instances
- Configure Network Gatekeeper clusters
- Configure Network Gatekeeper services, such as database connectivity (JDBC) and messaging (JMS)
- Monitor server and application performance
- View server and domain log files
- View application deployment descriptors
- Edit selected runtime application deployment descriptor elements

- Upgrade communication services
- Configure security parameters and roles

Use the WebLogic Network Gatekeeper specific section (accessed through the Domain Structure tree on the left side of the Administration Console) to:

- Configure Network Gatekeeper communication services
- Manage administrative users and groups
- Provision Application Service Providers, Applications, and Application Instances, and related SLAs.
- Monitor alarms, CDRs, and EDRs
-
- Create multiple plug-in instances, set up plug-in routing, etc.

Tasks performed outside the Console

- Extend Network Gatekeeper's functionality
- Backup and restore the system
- Upgrade the system

Complete information about WebLogic Network Gatekeeper OAM can be found in the *WebLogic Network Gatekeeper - System Administrator's Guide*.

OSS Integration

All or selected parts of the management application can also be integrated with external Operation Support Systems (OSS) through secured JMX/JMS interfaces. For more information on working with JMX, see *Developing Manageable Applications* and *Configuring and Using the WebLogic Diagnostic Framework.* Alarm supervision systems can set up external JMS listeners to receive user definable types of event-based data, including standard alarms. SNMP traps are sent to any registered SNMP managers.

CHAPTER 8

Charging and Billing Integration

The following describes WebLogic Network Gatekeeper's charging functionality:

- Overview
- CDRs
- Billing System Integration

Overview

WebLogic Network Gatekeeper supports charging based on time used or per-use services.

CDRs

CDRs are used for charging based either on time used or on access to certain per-use services. Charging based on time used is typically employed for calls. Per-use might be employed, for example, to charge for a positioning service.

CDR data can be stored in WebLogic Network Gatekeeper's internal charging database or retrieved in real-time by billing and post processing systems through a billing gateway (this requires integration with the billing gateway: see Billing System Integration).

Data Generation

Charging data is generated every time an application uses a communication service. The charging data is recorded by the communication service during the period the application interacts with the

network. When the interaction is closed, the communication service stores the charging data as a CDR in the Network Gatekeeper's database. (If WebLogic Network Gatekeeper is integrated with a billing gateway, the charging data is sent directly to the billing gateway.)

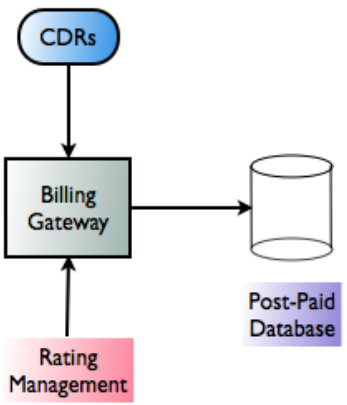
Billing System Integration

Network Gatekeeper can be integrated with external billing systems, either those that receive charging data directly or those that automatically retrieve information from Network Gatekeeper's database. CDRs can be customized to fit the requirements of these systems, both in terms of format and behavior.

Billing gateways

Real-time settlement of pre-paid accounts using CDR based charging requires integration through a billing gateway. This method can also be used to support post-paid services.

Figure 8-1 Billing integration through billing gateway

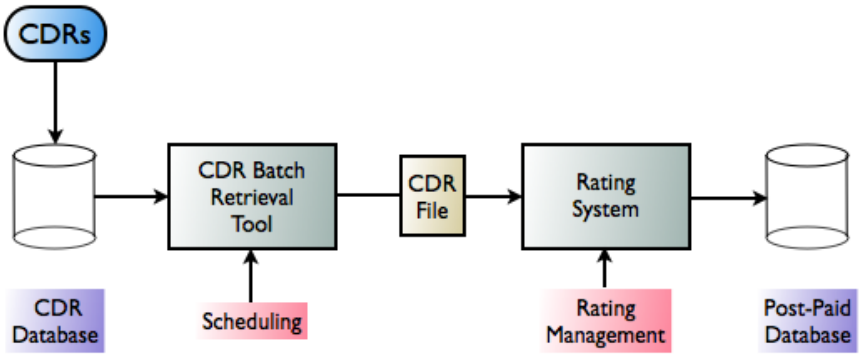


When integrating through a billing gateway, the billing gateway retrieves the CDRs in real-time through an external JMS-based charging listener. Rating, rating management, billing information storage, and pre-paid accounts settlement are handled by the billing gateway. The flow is shown above in Figure 8-1.

CDR database

In the case of applications that use post-paid accounts, it is possible to integrate billing by retrieving CDRs that have been stored in the Network Gatekeeper database.

Figure 8-2 Billing integration using the database



When integrating using this method, a CDR batch retrieval tool retrieves the CDRs from the database and stores them in a file format. The CDR file is processed by a rating system that transforms it into billing information and then stores it in a post-paid accounts database. The flow is shown above in Figure 8-2.

CHAPTER 9

Redundancy, Load Balancing, and High Availability

Redundancy, load balancing and high availability are essential for true carrier grade performance. WebLogic Network Gatekeeper uses both software and hardware components to achieve these important ends:

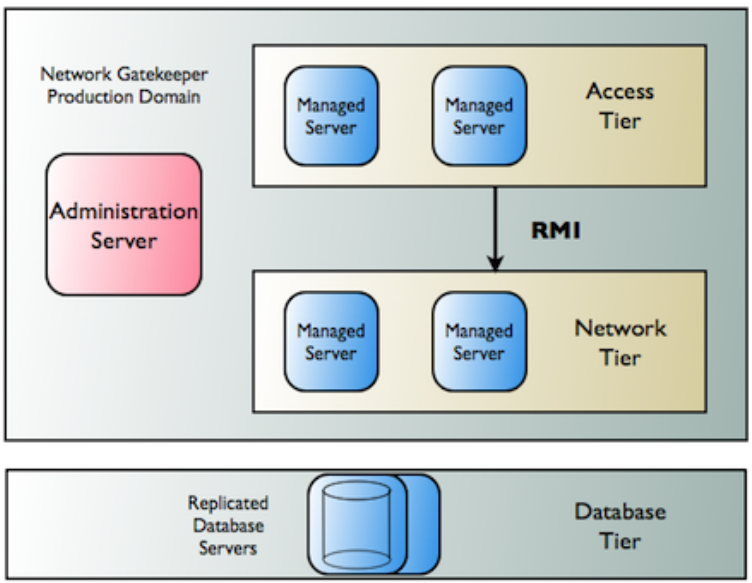
- Tiering
- Traffic Management Inside Network Gatekeeper
  - Application-initiated Traffic
  - Network-triggered Traffic
- Registering Notifications with Network Nodes
  - When the Network Node Supports Primary and Secondary Notification
  - When the Network Node Supports Only Single Notification
- Network Configuration
- Geographic Redundancy

WebLogic Network Gatekeeper's high availability mechanisms are supported by the clustering mechanisms made available by WebLogic Server 10.0 MP01. For general information about WebLogic Server and clustering, see *Using WebLogic Server Clusters*.

Tiering

For both high availability and security reasons, Network Gatekeeper is split into two tiers: the Access Tier and the Network Tier. Each tier consists of a cluster, with at least two server instances per cluster, and all server instances run in active mode, independently of each other. The servers in both clusters are, in the context of WebLogic Server, *managed* servers. Together the clusters make up a single WebLogic Server administrative domain, controlled through an administration server.

Figure 9-1 Sample Production Domain



Communication between the Access Tier and the Network Tier takes place using Java RMI. Application requests are load-balanced between the Access Tier and the Network Tier and failover mechanisms are present between the two. See Traffic Management Inside Network Gatekeeper for more information on these mechanisms in application-initiated and network-triggered traffic flows.

There is an additional tier containing the database. Within the cluster, data is made highly available using a cluster-aware storage service which ensures that all state data is made available across all Network Tier instances.

Traffic Management Inside Network Gatekeeper

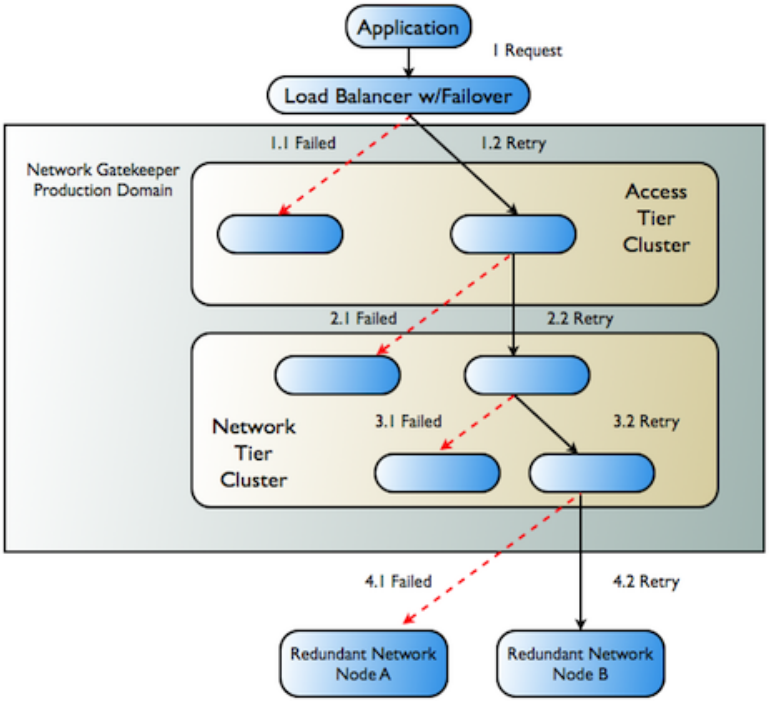
Potential failure is possible at many stages in traffic workflow in Network Gatekeeper. The following sections detail, tier by tier, how Network Gatekeeper deals with problems that might arise in both application-initiated and network-triggered traffic.

Application-initiated Traffic

Application-initiated traffic consists of all requests that travel from applications through Network Gatekeeper to underlying network nodes.

The example below follows the worst-case scenario for application-initiated traffic as it passes through Network Gatekeeper, and the failover mechanisms that attempt to keep the request alive.

Figure 9-2 Failover mechanisms in application-initiated traffic



1. The application sends a request to Network Gatekeeper. In a production environment, this request is routed through a hardware load balancer, usually protocol-aware. If the request towards the initial Access Tier server fails (1.1), either a time-out or a failure is reported. The load-balancer, or the application itself, is responsible for retrying the request.
2. The request is retried on a second server in the cluster (1.2) and it succeeds. It then attempts to send the request on to the Network Tier.
3. The request either fails to reach the Network Tier or fails during the process of marshalling/unmarshalling the request as it travels to the Network Tier server (2.1).
4. A fail-over mechanism in the Access Tier sends the request to a different server in the Network Tier cluster and it succeeds (2.2). It then attempts to send the request on to the network node.
5. The request is sent to a plug-in in the Network Tier that is unavailable (3.1). An Interceptor from the stack retries the remaining eligible plug-ins in the same server and succeeds (3.2).
6. The attempt to send the request to the telecom network node fails (4.1).
7. If a redundant pair of network nodes exists, the request is forwarded to the redundant node (4.2). If this request fails, the failure is reported to the application.

Network-triggered Traffic

Network-triggered traffic can consist of the following:

- Requests that contain a payload, such as terminal location or an SMS
- Acknowledgements from the underlying network node that an application-initiated request has been processed by the network node itself. A typical example might indicate that an SMS has reached the SMSC. From an application's perspective, this is normally processed as part of a synchronous request, although it may be asynchronous from the point of view of the network
- Acknowledgements from the underlying network node that the request has been processed by the destination end-user terminal; for example, an SMS delivery receipt indicating that the SMS has been delivered to the end-user terminal. From an application's perspective, this is normally handled as a incoming notification

For network-triggered traffic, Network Gatekeeper relies heavily on the telecom network node, or other external artifacts such as load-balancers with failover capabilities, to do failover.

Some network nodes can handle the registration of multiple callback interfaces. In such cases, Network Gatekeeper registers one primary and one secondary callback interface. If the node is

unable to send a request to the network plug-in registered as the primary callback interface, it is responsible for retrying the request, sending it to the plug-in that is registered as the secondary callback interface. This plug-in resides in another Network Tier instance. The plug-ins themselves are responsible for communicating with each other and making sure that both callback interfaces are registered. See When the Network Node Supports Primary and Secondary Notification below for more information.

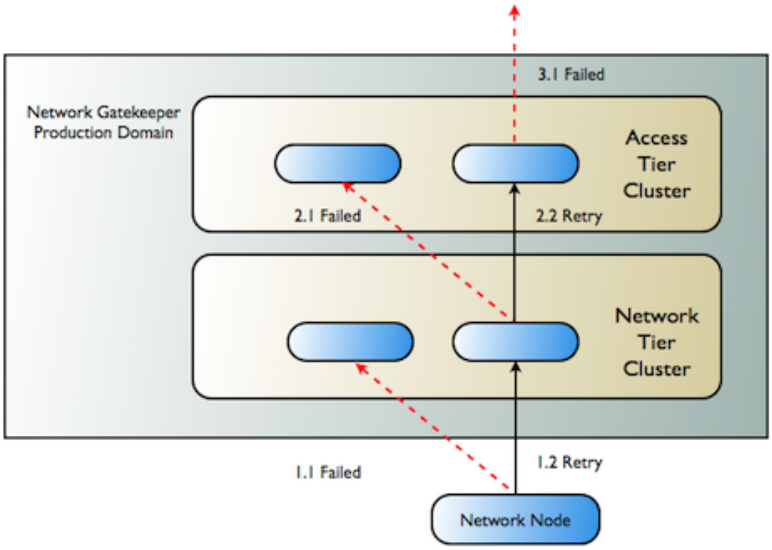
When the communication service is using SMPP, on the other hand, all Network Gatekeeper plug-ins can function equally as receivers for any transmission from the network node.

Finally, for HTTP-based protocols, such as MM7, MLP, and PAP, Network Gatekeeper relies on an HTTP load balancer with failover functionality between the telecom network node and Network Gatekeeper. See When the Network Node Supports Only Single Notification below for more information.

If a telecom network protocol does not support load balancing and high availability, a single point of failure is unavoidable. In this case, all traffic associated with a specific application is routed through the same Network Tier server and each plug-in has one single connection to one telecom network node.

The worst-case scenario for network triggered traffic for medium life span notifications using a network node that supports primary and secondary callback interfaces is described below..

Figure 9-3 Failover mechanisms in network-triggered traffic



1. A telecom network node sends a request to the Network Gatekeeper network plug-in that has been registered as the primary. It fails (1.1) due to either a communication or server failure.
2. The telecom network node resends the request, this time to the plug-in that is registered as the secondary call-back interface (1.2). This plug-in is in a different server instance within the Network Tier cluster. It succeeds.
3. The Network Tier attempts to send the message to the callback mechanism in the Access Tier. It fails (2.1).
4. If the request fails to reach the Access Tier, or failure occurs during the marshalling/unmarshalling process (2.1), the Network Tier retries, targeting another server in the Access Tier. It succeeds (2.2).

   Note: If, however, the failure occurs after processing has begun in the Access Tier, failover does not occur and an error is reported to the network node.

5. The callback mechanism in the Access Tier attempts to send the request to the application (3.1). If the application is unreachable or does not respond, the request is considered as having failed, and an error is reported to the network node.

Registering Notifications with Network Nodes

Before applications can receive network-triggered traffic, or notifications, they must register their interest in doing so with Network Gatekeeper, either by sending a request or having the operator set the notification up using OAM methods. In turn these notifications must be registered with the underlying network node that will be supplying them. The form of this registration is dependent on the capabilities of that node.

If registration for notifications is supported by the underlying network node protocol, the communication service's network plug-in is responsible for performing it, whether the registration is the result of an application-initiated registration request or an on-line provisioning step in Network Gatekeeper. For example, all OSA/Parlay Gateway interfaces support such registration for notifications.

Note: Some network protocols support some, but not all registration types. For example, in MM7 an application can register to receive notifications for delivery reports on messages sent *from* the application, but not to receive notifications on messages sent *to* the application from the network. In this case, registration for such notifications can be done as an off-line provisioning step in the MMSC.

Whether the plug-in sets up the notification in the network or it is done using OAM, Network Gatekeeper is responsible for correlating all network-triggered traffic with its corresponding application.

Notification Life Span

Notifications are placed into three categories, based on the expected life span of the notification. These categories determine the failover strategies used:

- Short life span

  These notifications are very short-lived, with an expected life span of a few seconds. Typically these are delivery acknowledgements for hand-off of the request to the network node, where the response to the request is reported asynchronously. For this category, a single plug-in, the originating one, is deemed sufficient to handle the response from the network node.

- Medium life span

These notifications are neither short- nor long-lived, with an expected life span of minutes up to a few days. Typically these are delivery acknowledgements for message delivery to an end-user terminal. For this category, the delivery notification criteria that have been registered are replicated to exactly one additional instance of the network protocol plug-in. The plug-in that receives the notification is responsible for registering a secondary notification with the network node, if possible.

- Long life span

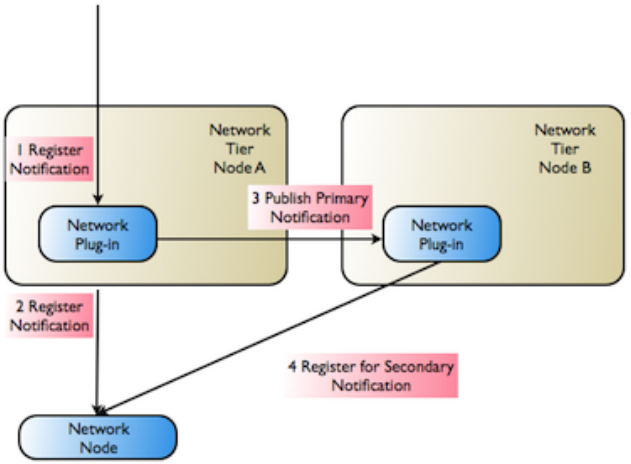
  These notifications are long-lived, with an expected life span of more than a a few days. Typically these are registrations for notifications for network-triggered SMS and MMS messages or calls that need to be handled by an application. For this category, the delivery notification criteria are replicated to all instances of the network plug-in. Each plug-in that receives the notification is responsible for registering an interface with the network node.

When the Network Node Supports Primary and Secondary Notification

Figure 9-4 below illustrates how Network Gatekeeper registers both primary and secondary notifications with network nodes that support it. This capability must be supported both by the network protocol in the abstract, and by the implementation of the protocol as it exists in both the network node and the communication service's network plug-in.

Note: The scenario assumes that the network node supports registration for notifications with overlapping criteria (primary/secondary).

Figure 9-4 Network node supports primary/secondary notifications



1. The request to register for notifications enters the network protocol plug-in from the application.
2. The primary notification is registered with the telecom network node.
3. The notification information is propagated to another instance of the network protocol plug-in.
4. The secondary notification is registered with the telecom network node.

Note: The concept of primary/secondary notification is not necessarily ordered. The most recently registered notification may, for example, be designated the primary notification.

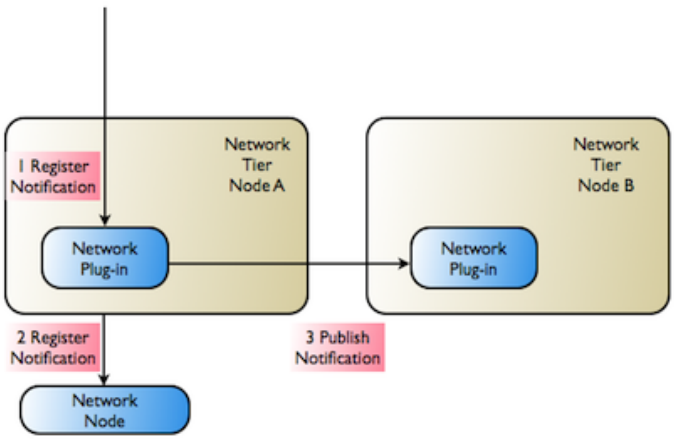
When a network-triggered request that matches the criteria in a previously registered notification reaches the telecom network node, the node first tries the network plug-in that registered the primary notification. If that request fails, the network node has the responsibility of retrying, using the plug-in that registered the secondary notification. The secondary plug-in will have all necessary information to propagate the request through Network Gatekeeper and on to the correct application.

When the Network Node Supports Only Single Notification

Figure 9-5 below illustrates the registration step in Network Gatekeeper if the underlying network node does not support primary/secondary notification registration.

Note: The scenario assumes that the network node does not support registration for notifications with overlapping criteria. Only one notification for a given criteria is allowed.

Figure 9-5 Network node supports only single notification



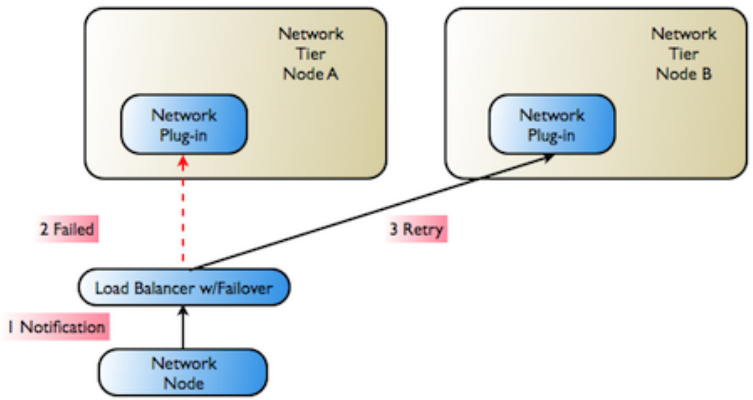
1. The request to register for notifications enters the network protocol plug-in from the application.
2. The notification is registered with the telecom network node.
3. The notification information (matching criteria, target URL, etc.) is propagated to another instance of the network protocol plug-in. The plug-in makes the necessary arrangements to be able to receive notifications.

There are two possibilities for high-availability and failover support in this case:

- All plug-ins can receive notifications from the network node. This is the case with SMPP, in which all plug-ins can function as receivers for any transmission from the network node
- A load balancer with failover support is introduced between the network protocol plug-in and the network node. This is the case with HTTP based protocols, as in Figure 9-6 below.

Note: Whether or not this is possible depends on the network protocol, as the load-balancer must be protocol-aware.

Figure 9-6 Traffic with a single notification only node: load-balancer with failover support



Network Configuration

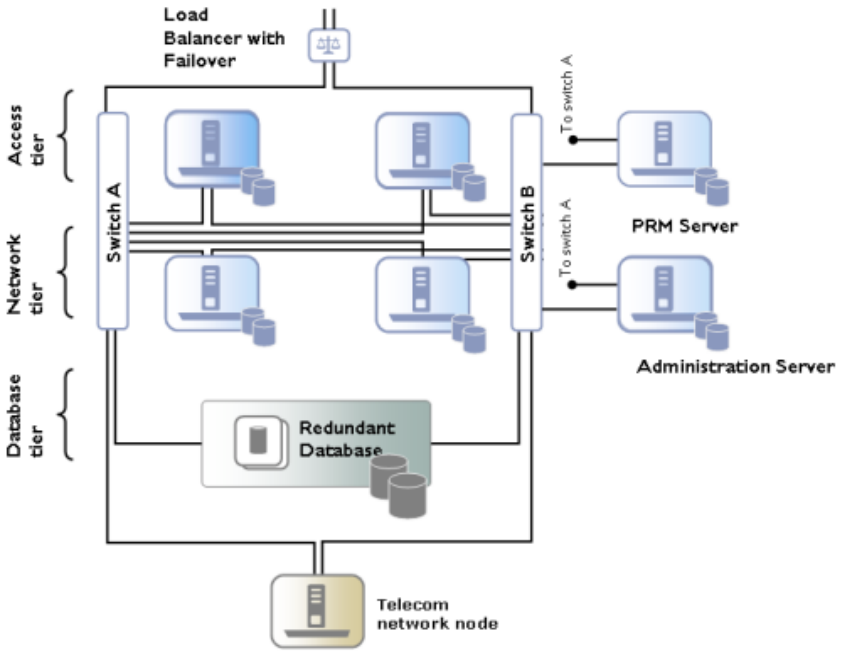
In addition to the specific hardware components listed above, the general structure of a production Network Gatekeeper installation is designed to support redundancy and high availability. A typical installation consists of a number of UNIX/Linux servers connected through duplicated switches. Each server has redundant network cards connected to separate switches. The servers are organized into clusters, with the number of servers in the cluster determined by the needed capacity.

As described previously, Network Gatekeeper is divided into an Access Tier, which manages connections to applications and a Network Tier, which manages connections to the underlying telecom network. For security, the Network Tier is usually connected only to Access Tier servers, the appropriate underlying network nodes, and the WebLogic Server administration server, which manages the domain. A third tier hosts the database. This tier should be hosted on dedicated, redundant servers. For physical storage, a Network Attached Storage via fibre channel controller cards is an option.

Because the different tiers perform different tasks, their servers should be optimized with different physical profiles, including amount of RAM, disk-types, and CPUs. Each tier scales individually, so the number of servers in a specific tier can be increased without affecting the other tiers.

A sample configuration is shown in Figure 9-7. Smaller systems in which the Access Tier and the Network Tier are co-located in the same physical servers are possible, but only for non-production systems,. Particular hardware configurations depend on the specific deployment requirements, and are worked out in the dimensioning and capacity planning stage.

Figure 9-7 Sample hardware configuration



In high availability mode, all hardware components are duplicated, eliminating single point of failure. This means that there are at least two servers executing the same software modules, that

each server has two network cards, and that each server has a fault-tolerant disk system, as, for example, RAID.

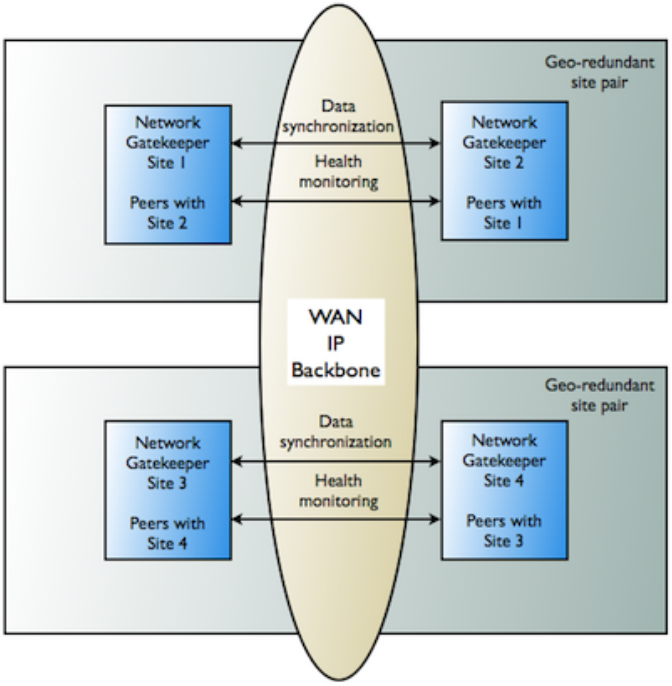
The administration server may have duplicate network cards, connected to each switch. The optional PRM servers should run on separate, dedicated servers.

For security reasons, the servers used for the Access Tier can be separated from the Network Tier servers using firewalls. The Access Tier servers reside in a Demilitarized Zone (DMZ) while the Network Tier servers are in a trusted environment.

Geographic Redundancy

All Network Gatekeeper modules in production systems are deployed in clusters to ensure high availability. This prevents single points of failure in general usage. To prevent service failure in the face of catastrophic events - natural disasters or massive system outages like power failures - Network Gatekeeper can also be deployed at two geographically distant sites as site pairs. Each site, which is a Network Gatekeeper domain, has a site peer. See Figure 9-8 for more information.

Figure 9-8 Overview of geographically redundant site pairs



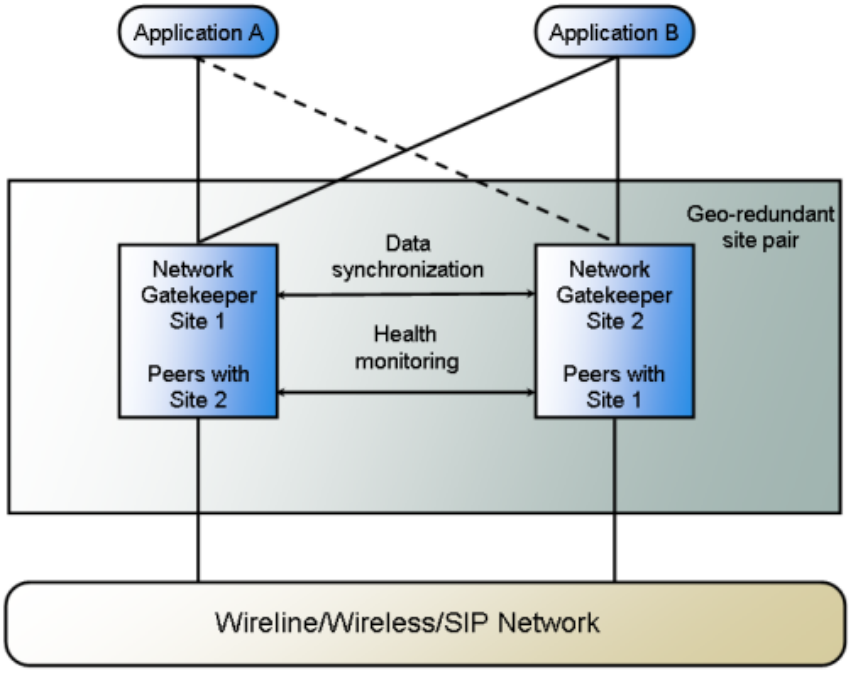
Note: The geographic distribution of the sites is *not* transparent to the applications accessing Network Gatekeeper. There is no single sign-on mechanism across sites and an application must establish a session with each site it intends to use. In case of site failure, an application must manually failover to a different site. Provisioning for each site must be performed individually.

SLA enforcement is synchronized across geographic sites and SLAs are enforced across predefined pairs. Each site is configured to have a reference to its peer site. A subset of all SLAs for a given site is designated as being enforceable across sites. Exactly which parts are selected depends on particular applications and their usage patterns.

Each site maintains a designated hub node that responsible for accounting and the enforcement of SLAs at that site. The service executing on the hub node is highly available and is migrated to another server should server failure occur. Cross-site enforcement is accomplished through hub-to-hub synchronization of global usage counts. The accuracy of enforcement across site pairs is configurable through an accuracy factor, which is translated into a synchronization interval based on, among other settings, the number of servers.

Applications that normally use only a single site for their traffic can failover to their peer site while maintaining ongoing SLA enforcement. This scenario is particularly relevant for SLA aspects that have longer term impact such as quotas.

Figure 9-9 Geographically redundant site pairs and applications



The geographic redundancy design does not maintain state for *ongoing conversations*. Conversations in this sense are defined in terms of the correlation identifiers that are returned to the applications by Network Gatekeeper or passed into Network Gatekeeper from the applications. Any state associated with a correlation identifier exists on only a single geographic site and is lost in the event of a site-wide disaster. Conversational state includes, but is not limited to, call state and registration for network triggered notifications. This type of state is considered volatile, or transient, and is not replicated at the site level.

By implication, therefore, conversations must be conducted and complete on their site of origin. If an application wishes to maintain conversational state cross-site - for example, to maintain a registration for network-triggered traffic - it must register with each site individually.

On the other hand, this type of affinity does not prevent load balancing between sites for different or new conversations. For example, because each request to send an SMS message constitutes a new conversation, sending SMS messages could be balanced between the sites.

Below is a high-level outline of the redundancy functionality:

- The SLA format allows contracts to be enforced across geographic site domains. It can be set up per SLA on both the service provider group and application group level. By default, SLAs are not enforced across sites.
- Connection lost alarms will be raised whenever the peer sites fail to establish connection a certain number of times. The number of retries is configurable.
- The Network Gatekeeper Budget Service is used to enforce SLA request and quota limits. The budget state is distributed to the other geographic sites. Network Gatekeeper automatically determines the optimal synchronization interval. For more information on the Budget Service, see "Managing and Configuring Budgets" in the *System Administrator's Guide*.
- The following categories of data are candidates for replication to other sites, depending on the restrictions on application load balancing and failover between sites you choose to support.
  - SLA quota counters

    Quotas that span over longer period of time are stored in the database to avoid losing state information during server or site failures. The replication is performed at the level of the Network Gatekeeper as a whole as opposed to relying on the databases to do the replication.
  - SLA request limits

    Request limits that span over longer period of time are persisted, in a manner similar to that of quota counters.
- Alarms are generated if a site does not have identical service provider group and application group level SLA configuration between peer sites.
- Alarms are generated if site A treats site B as a peer, but site B does not recognize site A as a peer.

CHAPTER 10

Service Extensibility

The following sections describe how to extend the WebLogic Network Gatekeeper functionality, including:

Overview

Networks change. Existing functionality is parsed in new ways to support new features. New nodes with new or modified abilities are added. Because of WebLogic Network Gatekeeper's highly modular design, exposing these new features to partners is a straightforward proposition. There are several ways to extend Network Gatekeeper:

- Entirely new communication services
- New network plug-ins that can work with existing application facing interfaces
- New or re-ordered interceptors or policy rules
- Integration with the existing network mechanisms with EDR listeners and SMNP MIBs.

The Platform Development Studio

To help operators and systems integrators, WebLogic Network Gatekeeper ships with the Network Gatekeeper Platform Development Studio. The PDS comprises the following features:

- *Platform Development Studio - Developer's Guide*

  A detailed guide covering how to:

  – Install the PDS

  – Use the Eclipse Wizard to generate a project

  – Understand the example communication service

  – Use Network Gatekeeper's container services

  – Add Policy Rules

  – Create Subscriber SLAs for subscriber-based policy

  – Update EDR filters and add JMS-based listeners

  – Re-order the Interceptor Stack or create new interceptors

  – Use the Platform Testing Environment, for functional and duration tests, and for unit tests

- Example Communication Service, including:

  – Source code

  – Wsdls

  – Build files

  – Example unit test

  – Example Subscriber SLA/Profile Provider

- The Eclipse Wizard

  The developer supplies information to an Eclipse plug-in wizard, which automatically sets up an Extension Project. Included within this project can be a substantial amount of generated code, including:

  – The entire Access Tier, with the Web Service implementation and any callback modules (EJBs) that are necessary

  Note: The Eclipse Wizard only supports building communication services based on Web Services application-facing interfaces.

  – Most of the code for the common services code in the Network Tier

  – A skeleton of the code required for the network plug-in layer of the Network Tier

- A complete Javadoc reference

- Specialized templates and Ant tasks
- The Platform Test Environment, launched using an Ant task at runtime from a flexible set of modules including:
  - Application Service Clients
  - Network Simulators
  - A complete example module including client and simulator
  - Utilities such as an MBean browser and JMS-based EDR listeners

  The PTE supports two modes of running:

    - Standalone with a Java Swing-based GUI
    - Console, particularly for use with the Unit Test Framework
- The Unit Test Framework, providing:
    - A base test class, derived from JUnit
    - Simple to use mechanisms for connecting to the Platform Test Environment
    - An example unit test bundled with the Communication Service Example
- The Profile Provider SPI, which allows operators and integrators to create subscriber-centric policy, associating subscribers to Service Provider and Application Groups based on individualized subscriber preferences and permissions.

APPENDIX A

Standards and Specifications

The following appendix provides a description of the specific standards that WebLogic Network Gatekeeper supports, along with, where possible, links to the actual specifications. A detailed statement of compliance is also available.

Application-facing interfaces

Parlay X 2.1

The Network Gatekeeper application-facing interfaces support the following parts of the Parlay X 2.1 specification.

Note: See http://parlay.org/en/specifications/pxws.asp for links to the specifications.

- **Common**, ETSI ES 202 391-1 V1.2.1 (2006-12) Open Service Access (OSA); Parlay X Web Services; Part 1: Common (Parlay X 2).
- **Third Party Call**, ETSI ES 202 391-2 V1.2.1 (2006-12) Open Service Access (OSA); Parlay X Web Services; Part 2: Third Party Call (Parlay X 2).
- **Call Notification**, ETSI ES 202 391-3 V1.2.1 (2006-12) Open Service Access (OSA); Parlay X Web Services; Part 3: Call Notification (Parlay X 2).
- **Short Messaging**, ETSI ES 202 391-4 V1.2.1 (2006-12) Open Service Access (OSA); Parlay X Web Services; Part 4: Short Messaging (Parlay X 2).
- **Multimedia Messaging**, ETSI ES 202 391-5 V1.2.1 (2006-12) Open Service Access (OSA); Parlay X Web Services; Part 5: Multimedia Messaging (Parlay X 2).

- **Terminal Location**, ETSI ES 202 391-9 V1.2.1 (2006-12), Open Service Access (OSA); Parlay X Web Services; Part 9: Terminal Location (Parlay X 2).
- **Presence**, ETSI ES 202 391-14 V1.2.1 (2006-12), Open Service Access (OSA); Parlay X Web Services; Part 14: Presence (Parlay X 2).

Parlay X 3.0

The Network Gatekeeper application-facing interfaces support the following parts of the Draft Parlay X 3.0 specification.

- **Common**, Draft ETSI ES 202 504-1 v.0.0.3 (2007-06), Open Service Access (OSA); Parlay X Web Services; Part 1: Common (Parlay X 3).
- **Third Party Call**, Draft ETSI ES 202 504-2 v0.0.5 (2007-06), Open Service Access (OSA); Parlay X Web Services; Part 2; Third Party Call (Parlay X 3).
- **Call Notification**, Draft ETSI ES 202 504-3 v0.0.3 (2007-06), Open Service Access (OSA); Parlay X Web Services; Part 3: Call Notification (Parlay X 3).
- **Audio Call**, Draft ETSI ES 202 504-11 v.0.0.3 (2007-06), Open Service Access (OSA); Parlay X Web Services; Part 11: Audio Call (Parlay X 3).

Extended Web Services

The Extended Web Services are Network Gatekeeper's proprietary application-facing interfaces. These interfaces are implementations of commonly requested functionality, including, in this release, WAP Push, Binary SMS, and Subscriber Profile. Although the interfaces themselves are not standardized, they use standardized elements.

Binary SMS

Note: See http://www.3gpp.org/ftp/Specs/html-info/23040.htm for links to the specification.

The payload, protocol identifier, and validity period rely on:

- 3GPP TS 23.040 version 6.5.0, Technical realization of Short Message Service (SMS)()

WAP Push

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

The payload of a WAP Push message shall adhere to:

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

Note: The Extended Web Services WAP Push communication service does not verify the payload. It simply passes it on to the underlying network node.

Subscriber Profile LDAP

There current specifications covering subscriber profile LDAP access, although a draft version exists. Gatekeeper's implementation is based on that draft.

Network protocol plug-ins

Off-the shelf, Network Gatekeeper supports the network protocols listed in Table 10-1 through the use of network plug-ins. Although each plug-in is a part of a given communication service, certain protocols can be used by multiple communication services for different purposes. In these cases there may be multiple implementations of the same protocol for use in different communication services.

Below is a list of supported network protocols organized per communication service.

Table 10-1 Network plug-ins organized per communication service.

Communication service	Network protocol plug-in	Specification
Parlay X 2.1 Third Party Call	SIP	RFC 3261. http://www.ietf.org/rfc/rfc3261.txt
	INAP/SS7	ETSI 94 INAP CS1, ETS 300 374-1, Intelligent Network (IN); Intelligent Network Capability Set 1 (CS1);Core Intelligent Network Application Protocol (INAP) http://pda.etsi.org/pda/queryform.asp

Table 10-1 Network plug-ins organized per communication service.

Communication service	Network protocol plug-in	Specification
Parlay X 3.0 Third Party Call	Parlay 3.3 MultiParty Call Control l	ETSI ES 201 915-4 V1.4.1 (2003-07), Open Service Access (OSA); Application Programming Interface (API); Part 4: Call Control SCF (Parlay 3), part MultiParty Call Control Service. Section MultiParty Call Control Service http://parlay.org/en/specifications/apis_archives.asp
Parlay X 2.1 Call Notification	SIP	RFC 3261. http://www.ietf.org/rfc/rfc3261.txt
Parlay X 3.0 Call Notification	Parlay 3.3 MultiParty Call Control	ETSI ES 201 915-4 V1.4.1 (2003-07), Open Service Access (OSA); Application Programming Interface (API); Part 4: Call Control SCF (Parlay 3), part MultiParty Call Control Service. Section MultiParty Call Control Service. http://parlay.org/en/specifications/apis_archives.asp
Parlay X 2.1 Short Messaging	SMPP v3.4	Short Message Peer to Peer, Protocol Specification v3.4, Document Version:- 12-Oct-1999 Issue 1.2. http://smsforum.net/
Parlay X 2.1 Multimedia Messaging	MM7 v 5.3.0	3rd Generation Partnership Project; Technical Specification Group Terminals; Multimedia Messaging Service (MMS); Functional description; Stage 2 (Release 5), 3GPP TS 23.140 V5.3.0. Messages are compliant with the schema defined by REL-5-MM7-1-2.xsd, http://www.3gpp.org/ftp/Specs/html-info/23140.htm

Table 10-1 Network plug-ins organized per communication service.

Communication service	Network protocol plug-in	Specification
Parlay X 2.1 Terminal Location	MLP 3.0 MLP 3.2 **Note:** Only one of the above listed protocols can be used at the same moment for a given node in a domain.	Location Inter-operability Forum (LIF) Mobile Location Protocol, LIF TS 101 Specification Version 3.0.0 and Mobile Location Protocol 3.2 Candidate Version 3.2 Open Mobile Alliance, OMA-TS-MLP-V3_2-20051124-C. MLP 3.0: http://www.openmobilealliance.org/tech/affiliates/lif/lifindex.html MLP 3.2: http://www.openmobilealliance.org
Parlay X 3.0 Audio Call	Parlay 3.3 Call User Interaction and Parlay 3.3 MultiParty Call Control	ETSI ES 201 915-5 V1.4.1 (2003-07), Open Service Access (OSA); Application Programming Interface (API); Part 5: User Interaction SCF (Parlay 3). Call user interaction parts. ETSI ES 201 915-4 V1.4.1 (2003-07), Open Service Access (OSA); Application Programming Interface (API); Part 4: Call Control SCF (Parlay 3). Section MultiParty Call Control Service http://parlay.org/en/specifications/apis_archives.asp
Parlay X 2.1 Presence	SIP	RFC 3261. http://www.ietf.org/rfc/rfc3261.txt
Extended Web Services WAP Push	PAP 2.0	Push Access Protocol, WAP Forum™, WAP-247-PAP-20010429-a. http://www.openmobilealliance.org
Extended Web Services Binary SMS	SMPP v3.4	Short Message Peer to Peer, Protocol Specification v3.4, Document Version:- 12-Oct-1999 Issue 1.2. http://smsforum.net/

Table 10-1 Network plug-ins organized per communication service.

Communication service	Network protocol plug-in	Specification
Extended Web Services Subscriber Profile	LDAPv3	Lightweight Directory Access Protocol, RFC 4510: June 2006 http://tools.ietf.org/html/rfc4510
Not applicable	Parlay 3.3 Framework **Note:** This is a network-facing protocol that does not belong to a certain communication service. It is used by all Parlay plug-ins.	ETSI ES 201 915-3 V1.4.1 (2003-07), Open Service Access (OSA); Application Programming Interface (API); Part 3: Framework (Parlay 3). http://parlay.org/en/specifications/apis_archives.asp

Security

Network Gatekeeper supports the security standards listed below. The security standards are applicable for the application-facing interfaces. Network Gatekeeper leverage Web Services Security mechanisms provided by WebLogic Server. For more information, see *Understanding WebLogic Security* and *WebLogic Web Services: Security*.

Note: See http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss for links to the specifications.

- WS-Security Core Specification 1.1
- WS-Security 1.0 and 1.1
- UsernameToken Profile 1.1
- X.509 Certificate Token Profile 1.1
- SAML Token Profile 1.1

- SOAP Message Security 1.0
- SOAP with Attachments (SWA) 1.1

In addition, the following standards are also supported:

- WS-Addressing 1.0, http://www.w3.org/2002/ws/addr/
- WS-Policy 1.1, http://ftpna2.bea.com/pub/downloads/WS-Policy.pdf
- WS-SecurityPolicy 1.2, http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702
- WS-Trust 1.3, http://docs.oasis-open.org/ws-sx/ws-trust/200512/ws-trust-1.3-os.html
- WS-SecureConversation 1.3, http://docs.oasis-open.org/ws-sx/ws-secureconversation/200512/ws-secureconversation-1.3-os.html
- WS-ReliableMessaging 1.0, http://ftpna2.bea.com/pub/downloads/wsrm-feb-14-2005.zip
- WS-PolicyAttachment 1.0, http://ftpna2.bea.com/pub/downloads/WS-PolicyAttachment.pdf

Transport-level security mechanisms such as 1 or 2-way SSL or VPN tunneling can be used for the PRM interfaces.

Identity and Trust

Network Gatekeeper leverages the robust identity management capabilities of Web Logic Server, including:

- Private Keys
- X.509 v3 Digital Certificates
- Symmetric & Asymmetric Key Algorithms
  - DES-CBC
  - Two-Key Triple DES
  - RC4
  - RSA
- Message Digest:
  - MD5

  - SHA
- JEE 5 & Weblogic Security Packages
  - Java Secure Socket Extension (JSSE)
  - Java Authentication & Authorization Services (JAAS)
  - Java Security Manager
  - Java Cryptography Architecture and Java Cryptography Extensions (JCE)
  - Java Authorization Contract for Containers (JACC)
  - Common Secure Interoperability Version 2 (CSIv2)