ComverseONE™

3.5 RT DAT 6.0

Charging Interfaces Guide

ComONE-3.5-RT-DAT-6.0-CIG-2010-07-23

Notice

This document contains proprietary and confidential material of Comverse, Inc. This document is furnished under and governed by either a license or confidentiality agreement. Any unauthorized reproduction, use, or disclosure of this material, or any part thereof, is strictly prohibited.

The material furnished in this document is believed to be accurate and reliable. However, no responsibility is assumed by Comverse, Inc. for the use of this material. Comverse, Inc. reserves the right to make changes to the material at any time and without notice. This document is intended for information and operational purposes only. No part of this document shall constitute any contractual commitment by Comverse, Inc.

Corporate Headquarters
200 Quannapowitt Parkway
Wakefield, MA 01880 USA
Tel: (781) 246-9000
Fax: (781) 224-8143
www.comverse.com

Revision History

The following table lists the document changes since the initial publication:

Date	Chapter	Description
07/23/2010		Initial publication.

Contents

Chapter 4 OSA Method Descriptions 71

Chapter 5 Diameter Infrastructure in Comverse ONE Solution 115

Figures

Tables

Notational Conventions

NOTE	Useful information appears in this format.
SEE	Provides direction to important information
IMPORTANT	Important information appears in this format.
CAUTION	Indicates possible risk of damage to data, software, or hardware.
WARNING!	Indicates serious risk of damage to data, software, or hardware.

Table 1 Notational Conventions

Notation	Explanation of Convention
References to printed documents	*Helvetica italic* **Example:** See *Database Reference Volume 2*.
<KEYS>	UPPERCASE HELVETICA, in angle brackets **Example:** Press <CTRL><Q><SHIFT><P> to create an em dash.
`User-entered text`	**`Courier bold`** **Example:** Enter **`Total Charges`** in the field.
`Placeholders for user-determined text`	*`Courier italic`*, in angle brackets **Example:** Enter your <*`password`*>.
`Code samples, TABLE_NAMES, field_names, file and directory names, file contents, user names, passwords, UNIX ENVIRONMENT_VARIABLES`	`Courier`
Placeholders for system-generated text	*Helvetica italic* **Example:** Messages appear in this form: *timestamp messageID* >> *text.*
Buttons, Icon Names, and Menu items	**Helvetica bold** **Example:** Choose **Reports** from the main menu.

Special Markers

The Comverse ONE Billing and Active Customer Management solution has the three derivatives shown in Table 2, "Labels in Markers." For user convenience, any content that is specifically included in a derivative is highlighted with special markers so that it can readily be distinguished.

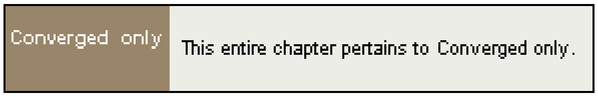
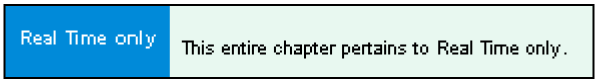
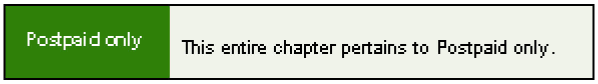
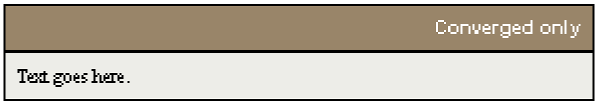
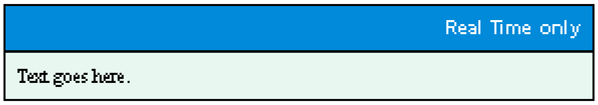
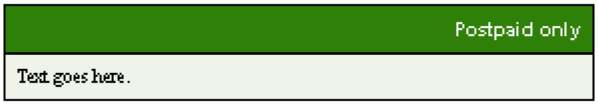
Table 2 Labels in Markers

Derivative	Label Shown in Markers
Comverse ONE Converged Billing derivative	**Converged only**
Comverse ONE Real-Time Charging derivative	**Real Time only**
Comverse ONE Postpaid Billing derivative	**Postpaid only**

Each derivative has a set of three color-coded markers, as shown in Table 3, "Types of Markers." The markers are used individually or in combination to highlight derivative-specific content by:

- Entire chapters
- Selected portions of chapters
- Tables, either entire or partial

Table 3 Types of Markers

Marker	Example	Description
Alert	 Converged only This entire chapter pertains to Converged only.  Real Time only This entire chapter pertains to Real Time only.  Postpaid only This entire chapter pertains to Postpaid only.	■ Placed at the beginning of an entire chapter that pertains only to a specific derivative. ■ Placed just before a table that partially or entirely pertains only to a specific derivative.
Block	 Converged only Text goes here.  Real Time only Text goes here.  Postpaid only Text goes here.	A shaded box that encloses sections of documentation that pertain only to a specific derivative.
Flag	Converged only Real Time only Postpaid only	■ Designates a shaded table row whose contents pertain only to a specific derivative. ■ In a bulleted list, designates an item that pertains only to a specific derivative.

Comverse ONE Documentation List

The documents described below reference the Comverse ONE solution products. All documentation available with the Comverse ONE solution is described in the following pages, organized by the following categories:

- Infrastructure Domain
- Rating, Charging, and Promotions Domain
- Billing and Financials Domain (Converged only)
- Customer and Order Management Domain (Converged only)
- Mediation and Roaming Solutions Domain
- Self-Service Solutions Domain

Infrastructure Domain

Download every document in the Infrastructure domain if you purchase the Comverse ONE solution. Documentation for this domain includes the following (in alphabetical order):

- *Alarms Reference*
  Contains tables of alarm IDs, descriptions, likely causes, and recommended resolutions for systems and components.
- *Back Office Administration GUI Guide*
- *Database Reference*
  Describes all database tables and fields in detail.
- *Glossary*
  Provides a list of terms used specifically for the Comverse ONE solution
- *Operation Reference*
  Describes the processes in the Comverse ONE solution.
- *Platform Operations Guide*
  Describes the back-end operations and maintenance functionality of the core Comverse ONE solution components. Includes AIX/HACMP platform and cluster operations, Linux/Veritas platform and cluster operations, backup/recovery, shared storage and fiber switch operations, and tape backup operations.
- *Product Catalog Overview*
  Provides a high-level description of the Comverse ONE solution Product Catalog, which is the primary mechanism for creating, configuring, managing, and propagating Product Catalog versions.
- *Product Catalog User Guide*
  Instructions on using the Product Catalog application to define and manage all aspects of service provisioning.
- *Schedulable Entity Reference Manual*
  Documents all the jobs, monitors, and workflows, for each component.
- *Security Platform Operations Guide*
  Technical overview of the security platform and information on how to provision and administer the platform.
- *Security Server API Guide*
  Provides an overview of the interfaces exposed by the Java-based Security SDK API, which client

applications can leverage to access various security services, such as authentication, authorization, auditing, key management, and credentials management. Also provides information on the Security Web Services API, which provides interfaces to a subset of Security Server commands (Identity Management commands).

- *Signaling Gateway Unit Guide*
  Describes the hardware, installation, configuration, and maintenance of the Signaling Gateway Unit (SGU) used to connect Comverse real-time systems to the SS7 signaling network using either traditional SS7 protocols or Sigtran (SS7 over IP).
- *System Measurements Guide*
  The Comverse ONE Solution automatically collects statistical data from the Service Logic Unit (SLU) and the Service Gateway Unit (SGU). This includes service statistics on the SLF layer and platform data on the IPF layer.
  This guide describes the format and location of this measurement information and provides a description of the meaning of the data.The measurement data can be used to create reports. It can also be imported into other applications (such as Excel) to be viewed.
- *Unified API Guide*
  General overview of the Unified API, a brief description of its architecture, and information about:
  - Framework classes and the functionality they provide
  - Two standard interfaces provided with the Unified API (client SDK and web services)
  - A subset of Unified API business methods most commonly used
- *Unified Platform Guide*
  Technical overview of the Unified Platform and information on the procedures to manage core systems operations in the Comverse ONE solution.

Rating, Charging, and Promotions Domain

Documentation for this domain includes the following (in alphabetical order):

- *Batch Provisioning Utility Guide*
  - The *CC Batch* utility enables bulk creation of recharge vouchers and subscribers.
  - The *Bulk Provisioning* Utility enables bulk creation of anonymous accounts to support the pre-activation of pre-paid SIM cards.
- *Call Flows Reference*
  Callflows detail the logic flow of specific scenarios. Multiple access numbers can map to the same callflow. Different resellers have the option to publish different numbers but share the same logic.
- *Charging Interfaces Guide*
  Describes the four interfaces that enable external services to support real-time authorization, rating, and charging for transactional usage: (1) the Event Charging Interface, a simple TCP/IP-based interface, (2) Open Services Access (OSA), (3) a Diameter-based interface version enhanced to take advantage of features of the Comverse ONE solution, and (4) a Diameter-based interface packet-switched version.
- *Customer Care Client Provisioning Guide* — Real-Time
  Detailed task-oriented instructions for using Customer Care Client.
- *Diameter Gateway Unit Guide*
  Describes the hardware, installation, configuration and maintenance of the Diameter Gateway Unit (DGU) used to connect Comverse real-time systems to external services, using the diameter protocol over IP.
- *Network Interfaces and Notifications Guide*
  Describes the operation, features, and provisioning of notifications, CAMEL-enabled services, and USSD-enabled services.

- *Network Self-Care Guide*
  Describes the configuration, structure, features, and callflows for the Network Self-Care application.
- *Rating Technical Reference*
  Describes the Unified Rating Engine, which is the subsystem responsible for gathering incoming CDRs and processing them for billing.
- *Reports and Data Extracts Guide* — Real-Time
  Describes the real-time Operational Reports Interface (ORI) and the Data Warehouse Extract Utility.
- *Recurring–Non-Recurring Charges Server Guide*
  Describes all processes commonly available through the Recurring —Non-Recurring Charges Server.
- *Voucher and Recharge Guide*
  Describes the process by which subscribers add funds to accounts using recharge vouchers through IVR, interaction with Customer Service, and other methods. Provides details of the Recharge Control Table, which allows resellers to provision the effects of recharges so that bonuses, discounts, and other changes to offers can result from a successful recharge. Also describes the Card Generator software used to create batches of vouchers and calling cards.

Billing and Financials Domain (*Converged* only)

Documentation for this domain includes the following (in alphabetical order):

- *Advanced Statement Numbering Guide*
  Describes how to configure and use Advanced Statement Numbering.
- *Billing Operations Guide*
  Describes how to schedule and run billing operations modules, which are the modules that retrieve usage, calculate balances, generate invoices, execute collections events, process payments, and perform other tasks.
- *Billing Reports and File Layouts User Guide*
  Describes control reports and other file formats.
- *Billing Technical Reference*
  Describes the inputs, the outputs, and the processing flow of all billing modules.
- *Collections Guide*
  Contains information on configuring Collections database tables, running the Collections module, and using the Collections interface.
- *Configurator Guide*
  Describes how to install and use the Configurator.
- *Invoice Designer Strings and Filters Reference*
  Describes the static strings, dynamic strings, and filters in the Invoice Designer.
- *Invoice Designer Technical Reference*
  Describes how to configure and run Invoice Designer.
- *Invoice Designer User Guide*
  Describes the Invoice Designer and how to perform the tasks needed to create an invoice template.
- *Journals Guide*
  Describes the theory, configuration, and running of Journals processes.
- *Miscellaneous Configurable Entities*
  Instructions for configuring late fees, adjustments, and several other database entities used in postpaid and converged billing.
- *Taxation Guide*
  Describes the configuration, operation, structure, and features of Taxation.

Customer and Order Management Domain (*Converged* only)

Documentation for this domain includes the following (in alphabetical order):

- *Application Integrator Add/Copy Header Adapter User Guide*
  Describes the adapter that adds or copies header information in messages.
- *Application Integrator Aggregator Adapter User Guide*
  Describes the adapter that aggregates multiple input messages as a single composite output message.
- *Application Integrator File Adapter User Guide*
  Describes the configuration process and rules for the file adapter.
- *Application Integrator Generic Services User Guide*
  Describes the Null adapter, Trash adapter, and Initiator adapter generic services.
- *Application Integrator Operator Guide*
  Describes the commands that operate the Application Integrator at creation and runtime.
- *Application Integrator Retry Adapter User Guide*
  Describes the use of the Retry adapter to resend messages in case of failed transmissions.
- *Application Integrator Sequence Adapter User Guide*
  Describes the use of the Sequence adapter to generate unique sequence numbers for messages.
- *Application Integrator System Administrator Guide*
  Outlines installation, sizing, operation, and administration of the Application Integrator and logging. Describes configuration of the user environment and commands for creation and operation of the Application Integrator.
- *Application Integrator Unified API Client Adapter User Guide*
  Describes the Unified API Client Adapter.
- *Application Integrator Unified API Server Adapter User Guide*
  Describes the Unified API Server Adapter.
- *Application Integrator User Guide*
  Describes creating integration specifications, creating instances of the Application Integrator, and commands for operation of the Application Integrator. Provides a complete user guide for the iMaker compiler.
- *Customer Center User Guide*
  Detailed task-oriented instructions for using Customer Center.
- *Inventory Guide*
  Describes the configuration, operation, structure, and features of Inventory.
- *Inventory Replenishment Guide*
  Describes the operation, structure, and features of Inventory Replenishment.
- *Orders Services Guide*
  Describes the structure and features of Orders Services.
- *Request Handling and Tracking and Service Fulfillment User Guide*
  Describes the configuration, operation, structure and features of Request Handling and Tracking and Service Fulfillment.
- *Workflow Developers Guide*
  Helps new users understand the rules-based business process management system so users can create solutions and integrate Workpoint within those solutions.
- *Workflow User Guide*
  Describes the configuration, operation, structure, and features of Workpoint.

Customer Relationship Management

- *Administrator’s Guide for SSA Collaboration*
  Describes SSA Open Architecture 6.0 SSA Collaboration.
- *Billing Reports and File Layouts User Guide*
  Describes control reports and other file formats.
- *Customer Center User Guide*
  Detailed task-oriented instructions for using Customer Center.
- *SSA CRM Sales Force Automation and SSA CRM Customer Service Administrator’s Guide*
  Provides supervisors, managers, and executives with the information to use the Customer Service and Sales Force Automation and Admin Console functionality to manage the work of their agents and salespeople. Instructions for day-to-day maintenance of the system are included in this book.
- *SSA CRM Sales Force Automation and SSA CRM Customer Service Application Reference Guide*
  Contains technical reference information relevant to implementers involved in implementing and customizing SSA CRM, powered by Epiphany, Customer Service and Sales Force Automation applications at customer sites. This book provides the reference context for the procedural information available in the Implementation Guide.
- *SSA CRM Sales Force Automation and SSA CRM Customer Service Architecture Reference Guide*
  Provides technical information relevant to individuals involved in implementing the SSA Open Architecture and the applications built on the architecture. The book consists of two parts. Part 1 Developer’s Guide provides an overview of the Open Architecture, application development, and other material relevant to implementation and customization. Part 2 Services describes architecture services.
- *SSA CRM Sales Force Automation and SSA CRM Customer Service Data Dictionary Guide*
  Includes a listing and description of the tables and columns used to store SSA Customer Service operational business data. It also includes a description of the naming conventions for the tables. The target audience includes database administrators, application developers, and implementors.
- *SSA CRM Sales Force Automation and SSA CRM Customer Service Implementation Guide*
  Provides procedural information relevant to individuals involved in implementing and customizing the SSA core and the Customer Service applications built on the core.
- *SSA CRM Sales Force Automation and SSA CRM Customer Service Installation and Configuration Guide*
  Includes all of the SSA Global-specific information required to get the SSA Global Sales and Service applications, Configurator, integration with ERPLN and the underlying SSA Global Open Architecture running. When special configuration is required or recommended for other platform-support software (such as WebLogic, WebSphere, SQL Server, Oracle, etc.), it is also included.
- *SSA CRM Sales Force Automation and SSA CRM Customer Service Integration Guide*
  Provides overview and configuration information for the set of tools used to exchange data with a variety of back-end data sources, including generic SQL sources, Java and EJB-based sources, Web services, and other database types.
- *SSA Open Architecture Administrator’s Guide for Infor Platform Services*
  Provides the administrative aspects of Platform infrastructure.
- *SSA Open Architecture Overview Guide for SSA Collaboration*
  Provides a functional overview of SSA Open Architecture Collaboration — SSA Collaboration and describes the key concepts of SSA Collaboration.

Online Help

Available for users of the SSA Contact Center, SSA Sales, SSA Self-Service, SSA Studio, SSA Workflow Designer, and SSA Dialog Designer

Mediation and Roaming Solutions Domain

Documentation for this domain is subdivided into Mediation/Roaming and Revenue Settlements.

Mediation and Roaming

Mediation and Roaming documentation includes the following (in alphabetical order):

- *Collection API Guide*
  Provides the concepts and functions for the Collection Application Programming Interface (CAPI).
- *Data Manager GUI Reference*
  Contains detailed information about GUI screens and form fields that appear in the Data Manager interface
- *GRID Mapping Language Developer Guide*
  Describes the mediation feature components, semantics, and general syntax of the GRID Mapping Language (GML).
- *Installation Guide for HP*
  Describes how to install and configure the application, components, and some third-party applications associated with the HP platform.
- *Installation Guide for HP Itanium*
  Describes how to install and configure the application, components, and some third-party applications associated with the HP Itanium platform.
- *Installation Guide for HP PA-RISC*
  Describes how to install and configure the application, components, and some third-party applications associated with the HP PA-RISC platform.
- *Installation Guide for IBM*
  Describes how to install and configure the application, components, and some third-party applications associated with the IBM platform.
- *Installation Guide for SUN*
  Describes how to install and configure the application, components, and some third-party applications associated with the SUN platform.
- *Mediation and Roaming User Guide*
  Provides information on how to use the GUI interface, including information on using the Data System Manager application pages.
- *Mediation API Guide*
  Contains reference information on using the Mediation API.
- *Roaming Database Reference*
  Provides reference information on the Roaming database.
- *Roaming Setup Guide*
  Describes how to configure the Roaming Setup application pages. It also provides information on working with TAP, RAP, and CIBER statistics.
- *Scripts Guide*
  Provides information on script files, which contain additional instructions on functions for data collection and transmission.
- *Socket-Based API Guide*
  Explains the building applications using the Socket-Based Record Transmission API. Programmers can use the guide to use the records received from the Data system for their own customized downstream application solutions.
- *System Manager GUI Reference*
  Contains detailed information about GUI screens and form fields that appear in the System Manager interface

- *Variable-Length GRID Guide*
  Provides information on how to configure the control files for variable-length GRID.

Revenue Settlements

Revenue Settlements documentation includes the following (in alphabetical order):

- *Comverse Revenue Settlements Billing System Adapter Guide*
  Describes the configuration, operation, and installation for the Billing System adapter.
- *Comverse Revenue Settlements Data Model Guide*
  Overview of data model entities (such as partners, accounts, revenue sharing, and rate schedules) and how to configure them in the database.
- *Comverse Revenue Settlements Database Reference*
  Detailed descriptions of fields and tables in the database.
- *Comverse Revenue Settlements Technical Reference*
  Instructions for installing and operating Revenue Settlements. Also contains processing descriptions.
- *Comverse Revenue Settlements User Guide*
  Instructions for using the Revenue Settlements GUI.

Self-Service Solutions Domain

The Comverse ONE Self-Service Solutions domain consists of the core products plus the optional separately licensed premium products. The core products consist of the following:

- Self-Service Solutions Platform
- Self-Service Solutions Applications

Self-Service Solutions Platform Documentation

The Self-Service Solutions Platform has a comprehensive set of documentation covering the installation, configuration, and use of our products. The documentation set is divided into the following categories:

- **Manuals**: These manuals cover installing and using the platform.
- **Reference**: These reference documents contain information about APIs, databases, configuration files, and so on. These documents are delivered in HTML.

Self-Service Solutions Platform Manuals

Self-Service Solutions Platform manuals include the following (in alphabetical order):

- *Administration Guide*
  Provides operations and maintenance instructions for Web applications using the Self-Service Solutions Platform.
- *Communications Billing and Usage Reference*
  Provides detailed descriptions of the data models and structure of the Self-Service Solutions Platform Communications Billing and Usage (CBU) database.
- *Connectors Development Guide*
  Provides instructions for developing and customizing Connectors of the Self-Service Solutions Platform.
- *Core Module Development Guide*
  Provides instructions for configuring and developing features of the core module of the Self-Service Solutions Platform.

- *Customer Interaction Datastore Reference*
  Provides detailed descriptions of the data models and the structure of the Self-Service Solutions Platform Customer Interaction Datastore (CID).
- *Database Modules Development Guide*
  Provides instructions for configuring, customizing, and developing features of the database module of the Self-Service Solutions Platform.
- *Platform Installation Guide*
  Provides installation and configuration instructions for the Self-Service Solutions Platform.
- *Platform Services Guide*
  Provides instructions for configuring, customizing, and developing features that use the services provided by the Self-Service Solutions Platform.
- *Processors Development Guide*
  Provides instructions for developing and customizing Processors of the Self-Service Solutions Platform.
- *Reports Development Guide*
  Provides instructions for developing and customizing Reports of the Self-Service Solutions Platform.
- *Self-Service Solutions Overview Guide*
  Provides a high-level architectural and functional description of the Comverse ONE Self-Service Solutions. It also includes a detailed description of the concepts and development process to create and deploy Self-Service Solutions.
- *Web Applications Development Guide*
  Provides instructions for configuring, developing, and deploying Web applications that use the Self-Service Solutions Platform.

Self-Service Solutions Platform Reference

Self-Service Solutions Platform reference documentation includes the following (in alphabetical order):

- *Base Logic Manager Reference*
  Describes usage syntax and configuration files for the Base Logic Manager (BLM) APIs. These APIs are the core services of the Self-Service Solutions Platform.
- *CID2CBU Object Mapping Reference*
  Describes the default mapping of Customer Interaction Datastore (CID) and Communications Billing and Usage (CBU) objects.
- *Communications Billing and Usage Reference*
  Provides detailed descriptions of fields and tables in the Communications Billing and Usage (CBU) database.
- *Customer Interaction Datastore Reference*
  Provides detailed descriptions of fields and tables in the Customer Interaction Datastore (CID).
- *Integration Services Framework API Reference*
  Describes usage syntax of the set of APIs to program connectors and other components of the Intelligent Synchronization Framework (ISF).
- *Integration Services Framework Message Cache Reference*
  Provides detailed descriptions of fields and tables in the Intelligent Synchronization Framework (ISF) Message Cache.
- *Integration Services Framework Script API Reference*
  Describes usage syntax of the Intelligent Synchronization Framework (ISF) script APIs to program the ISF connectors.
- *JavaServer Page Framework for Internet Application API Reference*
  Describes usage syntax for the JavaServer Page Framework for Internet Application (JFN) APIs. These APIs are used to build JSPs using the JFN. This framework provides basic application functions and services as the foundation of user interfaces.

- *Logger Message Reference*
  Provides detailed descriptions of the Self-Service Solutions Platform log messages.
- *QRA API Reference*
  Describes usage syntax for the Query, Reporting, and Analysis (QRA) Engine APIs. These APIs are used to build reports.
- *UTIL API Reference*
  Describes usage syntax for the UTIL package used by different components of the Self-Service Solutions Platform. This package contains a set of utilities including the logger. Self-Service Solutions Applications Documentation

Each Self-Service Solutions Application comes with a comprehensive set of documentation covering the installation, configuration, and use of the product. The application documentation expands and complements the Self-Service Solutions Platform documentation.

The documentation set is divided into the following categories:

- **Manuals**: These manuals cover installing and using the application.
- **Reference**: These reference documents contain information about APIs, databases, configuration files, and so on. These documents are delivered in HTML.

Self-Service Solutions Application Manuals

A full set of these manuals is available for each Self-Service Solutions Application. The documentation set includes the following (in alphabetical order):

- *Business Objects Model Reference*
  Provides a detailed description of the models and entities that make up the Self-Service Solutions Application.
- *Catalog Loader Reference*
  Provides information about the Catalog Loader, including a functional description as well as installation, configuration, and use instructions.
- *Configuration and Development Guide*
  Provides instructions for configuring and developing Self-Service Solutions Application features.
- *Feature Reference*
  Describes the logic and provides use cases for the functional domains of the application.
- *Comverse ONE Installation Guide*
  Provides detailed installation, configuration, and deployment instructions for the Self-Service Solutions Application alongside other elements of the Comverse ONE solution.
- *Installation and Deployment Guide*
  Provides detailed installation, configuration, and deployment instructions for the Self-Service Solutions Application.
- *Introduction*
  Provides a high-level architectural and functional description of the Self-Service Solutions Application. It covers common features, order management, account management, and bill presentment.

Self-Service Solutions Application References

A full set of these references is available for each Self-Service Solutions Application.The reference documentation set includes the following (in alphabetical order):

- *API Reference*
  Describes usage syntax for the Self-Service Solutions Application APIs. These APIs are used to program the user interface and manage data.
- *Invoice Schema Reference*
  Describes the invoice schema reference of the Self-Service Solutions Application.

- *Presentation Layer Page Flow Reference*
  Describes the page flows of the Self-Service Solutions Application.
- *Specification Entity Relationship Diagrams*
  Provides diagrams describing the actors, use cases, user activity, and storyboard in IBM Rational Rose format.

Self-Service Solutions - Separately Licensed Products

Documentation available with optional, separately-licensed premium products in the Comverse Self-Service Solutions is listed below.

Online Catalog Manager

Online Catalog Manager (OCM) documentation includes the following (in alphabetical order):

- *Introduction to the Online Catalog Manager*
  Provides a high-level architectural and functional description of the Online Catalog Manager.
- *Online Catalog Manager Getting Started Guide*
  Describes the best way to build product catalogs in the Online Catalog Manager. This manual is a template for creating end-user documentation.
- *Online Catalog Manager Installation and Configuration Guide*
  Provides installation and configuration instructions for the Online Catalog Manager.
- *Online Catalog Manager User Documentation Template*
  Describes the use of the Online Catalog Manager. This manual is a template for creating end-user documentation. This manual covers many common concepts and procedures of the OCM.
- *Online Catalog Manager User Guide*
  Provides a detailed description of the concepts and use of the Online Catalog Manager. The topics include:
  - Managing Media Files
  - Managing Offers
  - Managing Prices
  - Managing Products
  - Managing Properties
  - Managing Reference Data
  - Publishing

Chapter 1

Charging Interfaces Overview

New Features for This Release

The following is a list of new features in the Comverse ONE 3.5 RT DAT 6.0 release that impact the Charging Interfaces Guide:

- Accumulator Reset on Home Activity

Overview

Comverse ONE solution provides a number of interfaces to the global telecommunications and data network that range from the Event Charging Interface, a generic Transmission Control Protocol / Internet Protocol (TCP/IP) interface that enables a comprehensive set of transaction capabilties from diverse kinds of external clients using a simple set of network queries and responses, through the robust and proven Open Services Access (OSA) interface, to today's most flexible and broadly capable interface paradigm, Diameter.

Event Charging Interface

The Event Charging Interface is a generic TCP/IP-based network interface capable of communicating with an external client and the Comverse ONE solution to apply charges or tariffs to a Mobile Originating subscriber for Short Message Service (SMS) and other transaction activities, and adjust subscriber account balances. The Comverse ONE solution receives a usage amount in units, which must be rated or a currency amount of the charge for services from the Event Charging Interface.

Open Services Access (OSA) Interface

The OSA Charging Interface enables external value-added services such as SMS and Multimedia Message Service (MMS), to interface with the Comverse ONE solution in accordance with the 3GPP OSA/Parlay Charging Application Programming Interface (API), thereby enabling these external services to charge the subscriber's account using standards-based and widely deployed OSA charging methods.

The OSA Charging Interface supports both event- and session-based charging, and implements reservation and debit/credit methods of the Charging Service Capability Feature. Direct charging and split charging (multiple users being charged for using an application such as multi-user games) are not supported.

Diameter

Comverse ONE solution provides three separate implementations of Diameter charging.

Comverse Diameter Charging Interface (DCI) provides value-added services based on the rich set of rating and capabilities of the Comverse ONE solution implementation of the 3GPP Online Charging Server (OCS)

Comverse Packet-Switched Diameter Charging Interface (PS DCI) provides an implementation of the 3GPP Gy interface to enable billing for data transactions that use the General Packet Radio Service (GPRS) capabilities of the Global System for Mobile Communications (GSM).

Comverse IP Multimedia Subsystem Charging Interface (IMS) is an implementation of the IMS Domain charging described in 3GPP specification TS 32.260. This application is used for rating and charging IMS sessions.

Chapter 2

Event Charging Interface

Overview

The Comverse ONE solution Event Charging Interface is a TCP/IP-based interface that enables external services to support real-time authorization, rating, and charging for transactional usage.

The Event Charging Interface is used for charging many kinds of value-added services including sending and receiving SMS and MMS messages, Internet access, and other data content.

The Event Charging Interface supports two types of charges:

- **Direct Charge**: Enables the external service to debit a specific amount. This is used when rating is done by the client and Comverse ONE solution performs balance debit only.
- **Tariff-based Charge**: Uses tariff information stored within the Comverse ONE solution rating engine. This is typical of a Short Message Service Center (SMSC), which does not normally have rating capabilities.

Any charge can be reversed, if appropriate (for example, an SMS message that was not successfully delivered).

Event Charging Interface Specifications

The Event Charging Interface enables simple, network-based communications with external applications. A single Event Charging Interface Service Logic Unit (SLU) supports up to 20 simultaneous sessions from one or more external services.

The Event Charging Interface uses TCP/IP as the transport protocol. The configuration files /etc/hosts and /etc/services, which contain addresses and port numbers, must match the equivalent configuration files on the external service side.

Two TCP/IP ports (6200 and 6300) are reserved. Data Content Providers must always use port number 50013.

Each external service can maintain a permanent connection to the Event Charging Interface, or connect only when there is an action to perform.

Table 4, “Event Charging Interface Messages” shows the 19 messages the interface uses.

Table 4 Event Charging Interface Messages

External Service Originated	Event Charging Interface Originated
Heartbeat Request	Heartbeat Response
Validate Subscriber Request	Validate Subscriber Response
Apply Charge Request	Apply Charge Response
Apply Currency Charge Request	Apply Currency Charge Response
Apply Tariff Request	Apply Tariff Response
Transaction ID Acknowledge	
Reverse Charge	Reverse Charge Response
Confirmation Request	Confirmation Response

Table 4 Event Charging Interface Messages (Continued)

External Service Originated	Event Charging Interface Originated
Apply Tariff with Volume	Apply Tariff with Volume Response
	Invalid Message Response
Disconnect Session	

The external service initiates most actions, including establishing the TCP/IP session. Upon receipt of an action request from the external service, the Event Charging Interface attempts to perform the requested action, and sends a response message indicating the success or failure of the action.

Message sequences can be intermixed. The external service need not wait until one transaction has been completed before starting another transaction.

All data is left-aligned. Any place within a field that does not contain a data character must be filled with the space character (hex 020). The length field contains the length of the message excluding the length field itself.

Event Charging Interface and Unit Types

Each Event Charging Interface transaction maps three inputs acquired from the external service (Bearer / Discount / External Unit Type Description) to an initial Application / Unit Type (AUT) that is delivered to the Comverse ONE solution.

On the input side, Bearer and Discount must always have values, while External Unit Type Description could be a value passed from the Apply Tariff Volume message, or None.

For Apply Charge and Apply Currency Charge, Bearer and Discount are None (0) and External Unit Type Description is always mapped to a prerated unit type (currency).

The Apply Charge method, which does not support a Unit Type, always has an External Unit Type Description of None.

If the External Unit Type Description is not None, then the Unit Type associated with the External Unit Type Description must be the same as the Unit Type for the output. That is, the output associated with the Comverse ONE solution initial AUT.

There is a one-to-one mapping relationship between a Bearer Capability / Discount pair and an initial AUT. A Bearer Capability and Discount pair can never map to more than one initial AUT.

However, the relation between Bearer Capability / Discount pairs and initial AUT is many to one. More than one Bearer Capability and Discount pair can map to the same initial AUT.

Service Logic Unit Redundancy

The Event Charging Interface application resides on the Comverse ONE solution Service Logic Unit (SLU), which operates in an N+1 redundancy mode. Minimally, two SLUs are designated to support the Event Charging Interface application.

If a particular SLU is unavailable, the external service application must contain logic to establish TCP/IP sessions with another SLU.

Load Sharing

The Event Charging Interface does not contain any load sharing capabilities. If the load is greater than that which is handled by a single Event Charging Interface, it is up to the external service(s) to divide the load between multiple Event Charging Interface units.

Heartbeats

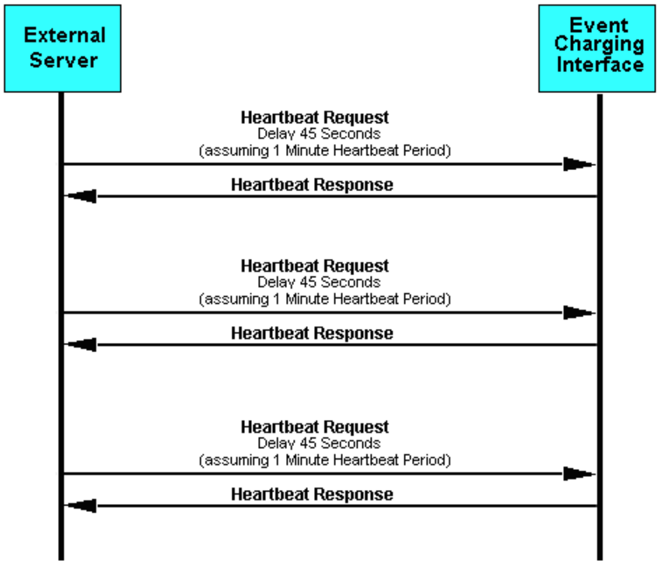
Heartbeat messages are sent between the external service and the Event Charging Interface to keep the session active when there is no other message activity from the external service.

The heartbeat period is configurable at the Event Charging Interface. The default value is one minute, and the maximum value is 30 minutes.

If the Event Charging Interface does not detect any external service messages for two heartbeat periods, it disconnects the session. The external service must implement similar functionality.

Figure 1, “Typical Heartbeat Message Sequence” shows the heartbeat message exchange sequence diagram.

Figure 1 Typical Heartbeat Message Sequence



Heartbeat Request

The external service sends a Heartbeat Request message to the Event Charging Interface to keep the session active when there is no other message activity. The Event Charging Interface always responds to a Heartbeat Request with a Heartbeat Response. Table 5, “Heartbeat Request Message” shows the contents of the Heartbeat Request message.

Table 5 Heartbeat Request Message

Field	Size (bytes)	Type	Value	Hexidecimal Value
Length	2	16 bit binary	2	0002x
Message ID	1	8 bit binary	1 = Request	01x
Message Name	1	8 bit binary	1 = Heartbeat	01x

Heartbeat Response

The Event Charging Interface sends a Heartbeat Response message to the external service in response to the Heartbeat Request message. Table 6, “Heartbeat Response Message” shows the contents of the Heartbeat Response message.

Table 6 Heartbeat Response Message

Field	Size(bytes)	Type	Value	Hexidecimal Value
Length	2	16 bit binary	2	0002x
Message ID	1	8 bit binary	2 = Response	02x
Message Name	1	8 bit binary	1 = Heartbeat	01x

Validate Subscriber

The Validate Subscriber sequence is a prequalification the external service uses to validate a Subscriber ID, or to determine if the subscriber has a certain amount of money available.

Using the Subscriber ID from the request message, the Event Charging Interface responds with a status indicating the subscriber's validity (valid Subscriber ID, valid subscriber state), or with an indication that the subscriber currently has sufficient balance to cover the charge.

This sequence returns information only and does not affect the subscriber's account.

Validate Subscriber Request

The external service sends a Validate Subscriber Request message to the Event Charging Interface to validate a subscriber's status and balance. Table 7, "Validate Subscriber Request Message" shows the contents of the Validate Subscriber Request message.

NOTE Validate Subscriber Request message does not actually debit the subscriber's account.

Table 7 Validate Subscriber Request Message

Field	Size (bytes)	Type	Value	Hexidecimal Value
Length	2	16 bit binary	53	
Message ID	1	8 bit binary	1 = Request	
Message Type	1	8 bit binary	2 = Validate Subscriber	
Associated Number	4	ASCII/binary	A value generated by the external service to uniquely identify this request	
Subscriber ID	30	ASCII	ID of subscriber being validated	
Charge Amount	17	ASCII	Amount of charge being validated	

Validate Subscriber Response

The Event Charging Interface sends the Validate Subscriber Response message to the external service in response to the Validate Subscriber Request message.

The response indicates whether the Subscriber ID is valid, if the subscriber is in a valid state, and if the subscriber has sufficient balance.

Table 8, "Validate Subscriber Response Message" shows the contents of the Validate Subscriber Response message.

Table 8 Validate Subscriber Response Message

Field	Size (bytes)	Type	Value	Hexidecimal Value
Length	2	16 bit binary	15	
Message ID	1	8 bit binary	2 = Response	
Message Type	1	8 bit binary	2 = Validate Subscriber	
Associated Number	4	ASCII/binary	The Associated Number received in the request	
Server ID	4	ASCII/binary	NA - filled with zeros	
Transaction ID	4	ASCII/binary	NA - filled with zeros	
Status ID	1	8 bit binary		00x =Subscriber is valid, active, and has sufficient balance 01x = Invalid Subscriber ID 02x = Subscriber not active 03x = Service unavailable 04x = Subscriber has insufficient balance

Apply Charge

The external service uses the Apply Charge sequence to debit a subscriber’s account by a specific amount.

The external service initiates the Apply Charge sequence by sending the Apply Charge Request message to the Event Charging Interface to debit a subscriber’s account. The Event Charging Interface confirms that the Subscriber ID is valid, the subscriber is in the Active state, and that the subscriber has enough balance to afford the Charge Amount. The Event Charging Interface deducts the charge amount from the subscriber, creates appropriate transaction records on the Comverse ONE solution system, and sends back a positive response message to the external service. It then starts a timer (with a default value of five minutes) to wait for the Transaction ID Acknowledge message from the external service.

If the the Event Charging Interface does not receive the Transaction ID Acknowledge within the defined time period, it reverses the subscriber charge at the Comverse ONE solution system.

If the subscriber is unavailable or has an insufficient balance, the Event Charging Interface sends a negative response to the external client, indicating the status of the subscriber, and does not deduct the charge amount from the subscriber's account.

Apply Charge Request

To debit a subscriber’s account by a specific amount, the external service sends an Apply Charge request message to the Event Charging Interface. If the subscriber is active with a sufficient balance, the charge amount is deducted from the subscriber's account.

If any of the validation steps fail, the Event Charging Interface sends a negative response message to the external service, no debit is applied to the subscriber’s account, and no call records are created on the Comverse ONE solution. No Transaction ID Acknowledge message is required when a negative response is received.

The apply charge sequence includes an Associated Number field. This field, supplied by the external service, links the request and response messages.

Table 9, “Apply Charge Request Message” shows the contents of the Apply Charge Request message.

Table 9 Apply Charge Request Message

Field	Size (bytes)	Type	Mandatory / Optional	Value	Hexidecimal Value
Length	2	16 bit binary	M	85	0055x
Message ID	1	8 bit binary	M	1 = Request	01x
Message Type	1	8 bit binary	M	3 = Apply Charge	03x
Associated Number	4	ASCII / binary	M	A value generated by the external service to uniquely identify this request	00000001x – This number is generated by the external service and used as a reference number for all messages related to this transaction.
Subscriber ID	30	ASCII	M	ID of subscriber to be validated	393939313233343536 20….20x that is, 999123456 Subscriber ID in the format that is stored in the Comverse ONE solution database. Field is left-justified, padded with spaces to fill in the 30 bytes.
Charge Amount	17	ASCII	M	Amount to be charged to subscriber's account. Valid amounts are –999999999999.999 to 0.00	31302E3030202020…20 that is, 10.00 dollars Field is left-justified, padded with spaces to fill in the 17 bytes.
Type of Charge	32	ASCII	M	A character string indicating the transaction type (for example, train ticket)	Note that if the first characters are MMS 72696E67746F6E6523312020…...20x that is, ringtone#1 Field is left-justified, padded with spaces to fill in the 32 bytes.

Apply Charge Response

The Event Charging Interface sends an Apply Charge Response message to the external service in response to the Apply Charge Request message.

A Status ID of 0 indicates that the account has been debited. Any other Status ID value indicates that the charge has not been applied.

Table 10, “Apply Charge Response Message” shows the contents of the Apply Charge Response message.

Table 10 Apply Charge Response Message

Field	Size (bytes)	Type	Value	Hexidecimal Value
Length	2	16 bit binary	15	000Fx
Message ID	1	8 bit binary	2 = Response	02x
Message Type	1	8 bit binary	3 = Apply Charge	03x

Table 10 Apply Charge Response Message (Continued)

Field	Size (bytes)	Type	Value	Hexidecimal Value
Associated Number	4	ASCII / binary	The Associated Number that was received in the request	00000001x – This is the number that was received in the Apply Charge request message to which this is the response.
Transaction ID	8	64 bit binary	This number is generated by the Event Charging Interface to uniquely identify this transaction. The external service must use this number in any follow-on messages such as Transaction Acknowledge of Reverse Charge.	0123456789ABCDEFx
Status ID	1	8 bit binary	00 = Prepaid Charge Applied 01 = Invalid Subscriber ID 02 = Prepaid subscriber not active 03 = Service unavailable 04 = Prepaid subscriber has insufficient balance 10 = Postpaid Charge Applied 12 = Postpaid subscriber not active 13 = Service unavailable 14 = Postpaid subscriber has insufficient balance	00x or 10x Indicates that the charge was applied to the subscriber. If the charge could not be applied, a non-zero Status ID is returned, and the value indicates why the charge could not be applied.

Transaction ID Acknowledgement

The external service sends the Transaction ID Acknowledge message to the Event Charging Interface in response to either the Apply Charge Response or Apply Tariff messages.

This message is sent to acknowledge that the charge or tariff should or should not be reversed.

NOTE

The Transaction ID Acknowledge message must be received within the defined time period or the Event Charging Interface automatically reverses the charge.

A Status ID of 0 indicates that the charge should remain. A Status ID of 1 indicates that the charge should be reversed.

Table 11, “Transaction ID Acknowledge Message” shows the contents of the Transaction ID Acknowledge message.

NOTE

The Transaction ID Acknowledge message also enables the external service to request that the charge be reversed.

Table 11 Transaction ID Acknowledge Message

Field	Size (bytes)	Type	Value	Hexidecimal Value
Length	2	16 bit binary	15	000Fx
Message ID	1	8 bit binary	1 = Request	01x
Message Type	1	8 bit binary	6 = Transaction ID Acknowledge	06x
Associated Number	4	ASCII / binary	The Associated Number that was received in the charge or tariff request	00000001x – This is the number that was received in the original Apply Charge request message to which this is the acknowledgement.
Transaction ID	8	64 bit binary	Transaction ID as supplied by the Event Charging Interface in the response message	0123456789ABCDEFx This is the number that the Event Charging Interface returned as part of the Apply Charge response.
Status ID	1	8 bit binary	0 = Apply 1 = Reverse	00x This message acknowledges the Apply Charge/Tariff response. Reverse (01x) is only used if the external service must immediately reverse the charge. This is an unlikely scenario.

Limitations

Because Apply Charge requests do not access Comverse ONE solution tariffs, these transactions are treated as balance adjustments, and have the following limitations:

- Only currency charges are permitted. Charging in other units is not supported.
- Apply Charge requests do not specify a currency type for the charge. When a client supplies a charge using ApplyCharge, the amount is deducted from the subscriber's currency balance regardless of the specific currency that subscriber uses. If the currency type is important to a particular kind of transaction, Apply Currency Charge must be used. Refer to the section “Apply Currency Charge,” on page 14.

Apply Currency Charge

The the external service uses the Apply Currency Charge sequence to debit a subscriber’s account by a specific amount of a specific currency type.

The external service sends the Apply Currency Charge Request message to the Event Charging Interface to initiate the sequence.

Apply Currency Charge Request is identical to the Apply Charge Request except for the addition of a Currency Type field, which contains the ISO Currency Code for the charge being applied.

Apply Currency Charge operates identically to the Apply Charge Request and Apply Charge Response scenarios except in the case where the charging currency does not match the subscriber’s Primary Offer currency.

- If the currency mismatch flag is set, the charge is converted to the subscriber’s Primary Offer currency, using the applicable conversion rate based on the subscriber’s primary home time zone.
- If currency mismatch is not set, the call is rejected. The Apply Currency Charge Response indicates a failure status of Currency Mismatch.

Apply Currency Charge Request

The external service sends the Apply Currency Charge Request message to the Event Charging Interface to debit a subscriber's account by a specific amount.

Table 12, "Apply Currency Charge Request Message" shows the contents of the Apply Currency Charge Request message.

Table 12 Apply Currency Charge Request Message

Field	Size (bytes)	Type	Mandatory / Optional	Value	Hexidecimal Value
Length	2	16 bit binary	M	88	0058x – hex equivalent of 88
Message ID	1	8 bit binary	M	1=Request	01x
Message Type	1	8 bit binary	M	10=Apply Currency Charge	0Ax
Associated Number	4	ASCII / binary	M	A value generated by the external service to uniquely identify this request	00000001x – This number is generated by the external service to be used as a reference number for all messages related to this transaction.
Subscriber ID	30	ASCII	M	ID of subscriber to be validated	393939313233334353620….20x that is, 999123456 Subscriber ID in the format that is stored in the Comverse ONE solution database. Field is left-justified, padded with spaces to fill in the 30 bytes.
Charge Amount	17	ASCII	M	Amount to be charged to subscriber's account. Valid amounts are 0.00 to 999999999999.999	31302E3030202020…20 that is, 10.00 dollars Field is left-justified, padded with spaces to fill in the 17 bytes.
Type of Charge	32	ASCII	M	A character string indicating the transaction type (for example, train ticket)	Note that if the first characters are MMS 72696E67746F6E6523312020…...20x that is, ringtone#1 Field is left-justified, padded with spaces to fill in the 32 bytes.
Currency Type	3	ASCII	M	The ISO Currency Code for the charge being applied.	

The Event Charging Interface sends the Apply Currency Charge Response message to the external service in response to the Apply Currency Charge Request message.

A Status ID of 0 indicates that the account has been debited. Any other Status ID value indicates that the charge has not been applied

Table 13, "Apply Currency Charge Response Message" shows the contents of the Apply Currency Charge Response message.

Table 13 Apply Currency Charge Response Message

Field	Size (bytes)	Type	Value	Hexidecimal Value
Length	2	16 bit binary	1	000Fx
Message ID	1	8 bit binary	2=Response	02x
Message Type	1	8 bit binary	10=Apply Currency Charge	0Ax
Associated Number	4	ASCII / binary	The Associated Number that was received in the request	00000001x – This is the number received in the Apply Currency Charge request message to which this is the response.
Transaction ID	8	64 bit binary	This number is generated by the Event Charging Interface to uniquely identify this transaction. The external service must use this number in any follow-on messages such as Transaction Acknowledge of Reverse Charge.	0123456789ABCDEFx
Status ID	1	8 bit binary	00=Charge Applied 01=Invalid Subscriber ID 02= Subscriber not active 03=Service unavailable 04= Subscriber has insufficient balance 10= Currency Mismatch	(0Ax) 00x Indicates that the charge was applied to the subscriber. If the charge could not be applied, then a non-zero Status ID is returned, and the value indicates why the charge could not be applied.

Transaction ID Acknowledgement

The external service sends the Transaction ID Acknowledge message to the Event Charging Interface in response to the Apply Currency Charge Response message.

This message is sent to either acknowledge that the charge or tariff should be taken, or to notify the Event Charging Interface that the charge/tariff should be reversed.

This message is a protocol acknowledgement that the client has received the successful Apply Currency Charge Response.

This message must be received within the agreed upon time period, (five minutes or less), or the Event Charging Interface automatically reverses the charge.

A Status ID of 0 indicates that the charge should remain. A Status ID of 1 indicates that the charge should be reversed.

Table 14, “Transaction ID Acknowledge Message” shows the contents of the Transaction ID Acknowledge message.

Table 14 Transaction ID Acknowledge Message

Field	Size (bytes)	Type	Value	Hexidecimal Value
Length	2	16 bit binary	15	000Fx
Message ID	1	8 bit binary	1=Request	01x
Message Type	1	8 bit binary	6=Transaction ID Acknowledge	06x

Table 14 Transaction ID Acknowledge Message (Continued)

Associated Number	4	ASCII / binary	The Associated Number that was received in the charge or tariff request	00000001x – This is the number that was received in the original Apply Currency Charge request message to which this is the acknowledgement.
Transaction ID	8	64 bit binary	Transaction ID as supplied by the Event Charging Interface in the response message	0123456789ABCDEFx This is the number that the Event Charging Interface returned as part of the Apply Currency Charge response.
Status ID	1	8 bit binary	0=Apply 1 = Reverse	00x This message acknowledges the Apply Charge/Tariff response. Reverse (01x) is only used if the external service must immediately reverse the charge. This is an unlikely scenario.

Apply Currency Charge Message Sequences

Figure 2, “Typical Apply Currency Charge Message Sequence – Charge Not Reversed” through Figure 4, “Typical Apply Currency Charge Message Sequence – Charge Reversed Due To Timeout” show typical Apply Currency Charge message sequences.

Figure 2 Typical Apply Currency Charge Message Sequence – Charge Not Reversed

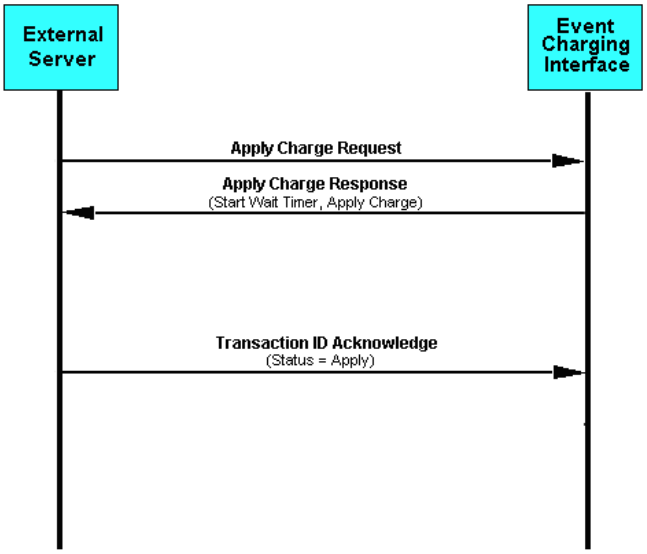


Figure 3 Typical Apply Currency Charge Message Sequence – Charge Reversed by External Service

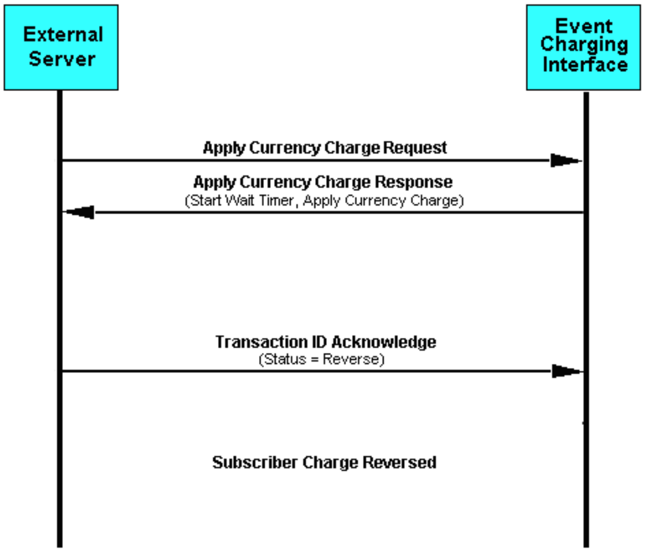
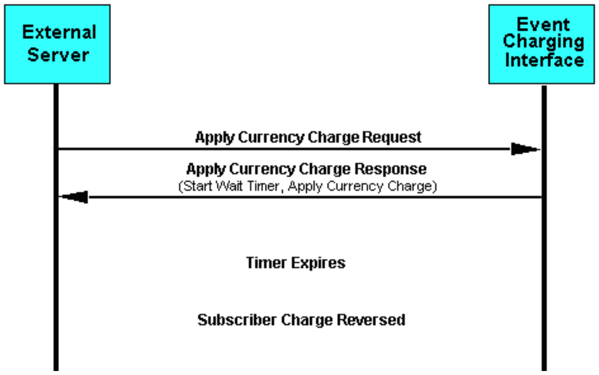

External Server
Event Charging Interface
Apply Currency Charge Request
Apply Currency Charge Response
(Start Wait Timer, Apply Currency Charge)
Transaction ID Acknowledge
(Status = Reverse)
Subscriber Charge Reversed

Figure 4 Typical Apply Currency Charge Message Sequence – Charge Reversed Due To Timeout


External Server
Event Charging Interface
Apply Currency Charge Request
Apply Currency Charge Response
(Start Wait Timer, Apply Currency Charge)
Timer Expires
Subscriber Charge Reversed

Apply Tariff

The external service sends the Apply Tariff request message to the Event Charging Interface to charge a subscriber using the Comverse ONE solution tariff and rating engine. If the transaction is authenticated, the Comverse ONE solution determines the actual cost and enables the Event Charging Interface to deduct the derived charges tariff from the appropriate party.

There are three transaction scenarios in which a tariff is applied: Mobile Originating (MO) to application, application to Mobile Terminating (MT), or MO to MT.

- **Mobile Originating to Application (MO to App)**: Deducts the tariff from the originating subscriber's account.
- **Application to Mobile Terminating (App to MT)**: Deducts the tariff from the terminating subscriber's account.
- **Mobile Originating to Mobile Terminating (MO to MT)**: Deducts the tariff from the originating subscriber's account.

Charges are calculated based on a variety of factors, which are described in the section “Calculate the Charge,” on page 23.

If the subscriber to be charged can pay for the transaction (has enough funds, and is in a state that permits the transaction), the subscriber’s account is debited, and the Event Charging Interface responds with an Apply Tariff response message, with a status indicating the transaction is permitted and has been charged.

Additionally, the Event Charging Interface starts a timer (default value five minutes) to wait for the Transaction ID Acknowledge message from the external service. If the Transaction ID Acknowledge is not received within the defined time period, the subscriber charges are automatically reversed.

Each time the Event Charging Interface updates the Comverse ONE solution database of account balances, adds a record to the Event Charging Interface Transaction history table, and, optionally, generates a usage record.

If the subscriber is unavailable or has insufficient balance, the Event Charging Interface sends a negative response message to the external client, the charge amount is not deducted from the subscriber's account, the transaction is recorded in the Event Charging Interface history and a usage record is generated in Comverse ONE solution.

NOTE No Transaction ID Acknowledge message is required when a negative response is received.

The Apply Tariff sequence also includes an Associated Number field. This field, supplied by the external service, links the request and response messages.

NOTE The Unified Rating Engine (URE) has the capability to apply a negative tariff to reduce an existing negative charge. For example, a 10% discount to a -$0.40 charge results in a charge of -$0.36. Refer to the Rating Technical Reference for more details.

Apply Tariff Request

The external service sends the Apply Tariff Request message to the Event Charging Interface to charge a subscriber using the Comverse ONE solution tariff and rating engine.

Table 15, “Apply Tariff Request Message” shows the contents of the Apply Tariff Request message.

Table 15 Apply Tariff Request Message

Field	Size (bytes)	Type	Mandatory / Optional	Value	Hexidecimal/Value
Length	2	16 bit binary	M	135	0087x
Message ID	1	8 bit binary	M	1 = Request	01x
Message Type	1	8 bit binary	M	4 = Apply Tariff	04x
Associated Number	4	ASCII / binary	M	A value generated by the external service to uniquely identify this request	00000002x – This is the number that was received in the Apply Charge request message to which this is the response.
Originating Caller ID	30	ASCII	M	Originating Subscriber ID or Application ID	39393939393132332020….20x that is, 99991234 This represents the Application ID, which is some unique way to identify the service.
Originating Subscriber MSC Address	30	ASCII	O	MSC Address / Cell ID of originating subscriber	20202020….20x
Subscriber Type	1	8 bit binary	M	1 = Mobile Originating to Mobile Terminating (MO to MT) 2 = Mobile Originating to Application 3 = Application to Mobile Terminating 4 = Data	01x or 02x or 03x or 04x
Terminating Caller ID	30	ASCII	M	Terminating Caller ID or Application ID If blank, and the Terminating Subscriber MSC Address is also blank, the terminating party is assumed to be anywhere.	393939313233343536 20….20x that is, 999123456 Subscriber ID in the format that is stored in the Comverse ONE solution database. In this case, this is the subscriber receiving the item (ringtone). Field is left-justified, padded with spaces to fill in the 30 bytes.

Table 15 Apply Tariff Request Message (Continued)

Field	Size (bytes)	Type	Mandatory / Optional	Value	Hexidecimal/Value
Terminating Subscriber MSC Address	30	ASCII	O	MSC Address / Cell ID of terminating subscriber	20202020….20x This field is only used for location-based charges, If not used, it is filled with all spaces.
Bearer Capability	4	ASCII	M	2 = WAP/SMS Transaction 3 = Web Transaction 11 = Data-Services	32202020x that is, 2
Discount	4	ASCII	M	Allowable values: 0–255 This field is used by the tariff engine to determine the charge for the service.	31323520x that is, 125 A combination of the Application ID and Discount fields ultimately determines which tariff is applied to calculate the charge.

NOTE For content providers, the Subscriber Type must be 4 (Data), and the Bearer Capability must be 11 (Data-Services).

Apply Tariff Response

Table 16, “Apply Tariff Response Message” shows the contents of the Apply Tariff Response message.

Table 16 Apply Tariff Response Message

Field	Size (bytes)	Type	Value	Hexidecimal Value
Length	2	16 bit binary	15	000Fx
Message ID	1	8 bit binary	2 = Response	02x
Message Type	1	8 bit binary	4 = Apply Tariff	04x

Table 16 Apply Tariff Response Message (Continued)

Field	Size (bytes)	Type	Value	Hexidecimal Value
Associated Number	4	ASCII / binary	The Associated Number that was received in the request	00000002x – This is the number that was received in the Apply Charge request message to which this is the response.
Transaction ID	8	64 bit binary	This 8 byte number is generated by the Event Charging Interface to uniquely identify this transaction. The external service must use this number in any follow-on messages such as Transaction Acknowledge of Reverse Charge.	0123456789ABCDFFx
Status ID	1	8 bit binary		00x = Charge successfully applied to prepaid subscriber 01x = Invalid Subscriber ID (subscriber not found) 02x = Prepaid subscriber not active or not data enabled, charge not applied 03x = Service unavailable 04x = Prepaid subscriber has insufficient balance, partial charge taken 05x = Tariff engine error 06x = No tariffs found 10x = Charge successfully applied to postpaid subscriber 12x = Postpaid subscriber not active or not data enabled, charge not applied 14x = Postpaid subscriber has insufficient balance, partial charge taken For purposes of this field, a Toggle subscriber is considered Prepaid when in Prepaid mode and Postpaid when in Postpaid mode.

Transaction ID Acknowledgement

Table 17, “Transaction ID Acknowledge Message” shows the contents of the Transaction ID Acknowledge message.

Table 17 Transaction ID Acknowledge Message

Field	Size (bytes)	Type	Value	Hexidecimal Value
Length	2	16 bit binary	15	000Fx
Message ID	1	8 bit binary	1 = Request	01x
Message Type	1	8 bit binary	6 = Transaction ID Acknowledge	06x
Associated Number	4	ASCII / binary	The Associated Number that was received in the charge or tariff request	00000002x – This is the number that was received in the original Apply Tariff request message to which this is the acknowledgement.
Transaction ID	8	64 bit binary	Transaction ID as supplied by the Event Charging Interface in the response message	0123456789ABCDFFx This is the number that the Event Charging Interface returned as part of the Apply Tariff response.
Status ID	1	8 bit binary	0 = Apply 1 = Reverse	00x This message acknowledges the Apply Charge/Tariff response. Reverse (01x) is only used if the external service must immediately reverse the charge. This is an unlikely scenario.

Calculate the Charge

The unified rater in the Comverse ONE solution billing model calculates the charge for the service. It first maps the received Bearer and Discount values into an Initial AUT via the Event Charging Interface Translation table (shown in the section “Event Charging Interface Provisioning in Product Catalog,” on page 33).

The initial AUT is then mapped to a final AUT based on segmentation keys such as Location Relationships, using the location information provided in the Originating and Terminating MSC Address fields in the Apply Tariff message, or special features such as Friends and Family, Calling Circles, and so on.

The Final AUT is then used to determine the tariff plan (and associated tariffs) based on the subscriber’s offer, as well as the date and time of the usage. Charges are then calculated based on the tariffs within the tariff plan.

Apply Tariff Message Sequences

Figure 5, “Typical Apply Tariff Message Sequence – Charge Not Reversed” through Figure 7, “Typical Apply Tariff Message Sequence – Charge Reversed Due To Timeout” show typical Apply Tariff message sequences.

Figure 5 Typical Apply Tariff Message Sequence – Charge Not Reversed

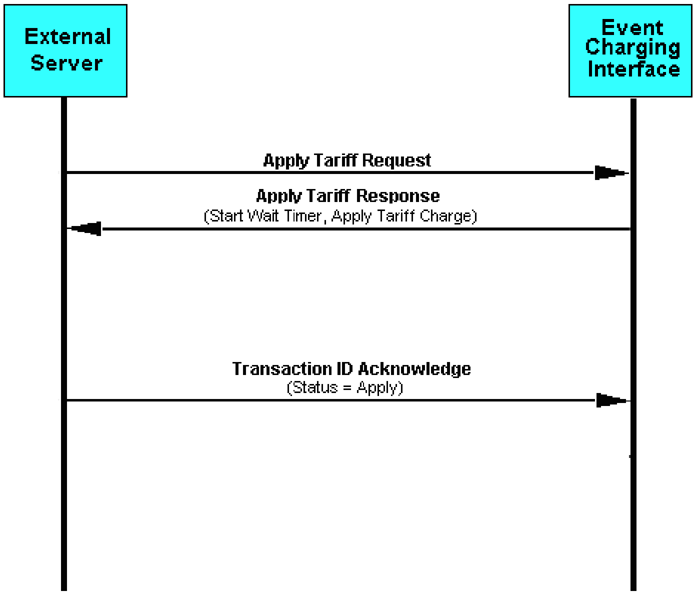


Figure 6 Typical Apply Tariff Message Sequence – Charge Reversed by External Service

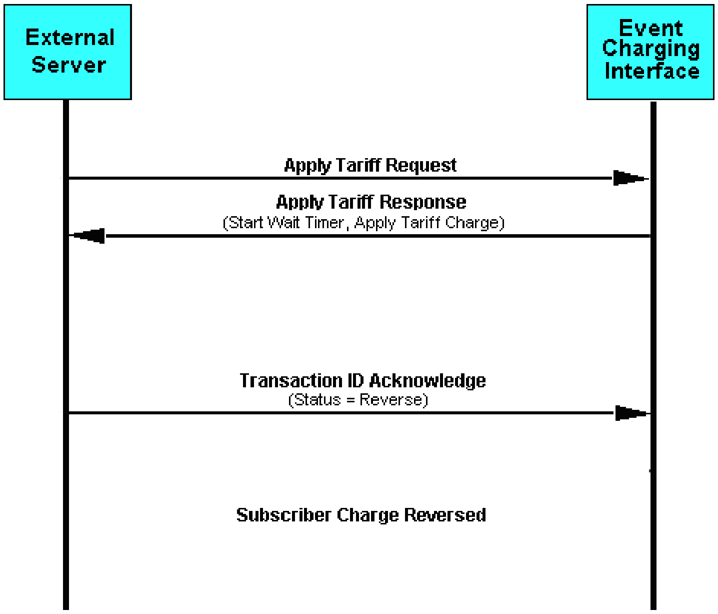
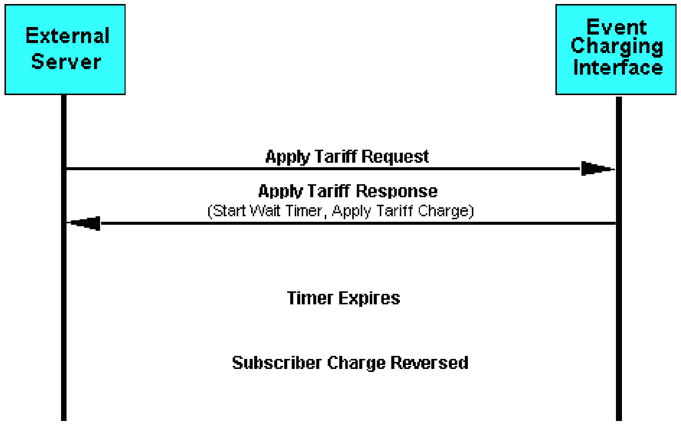


Figure 7 Typical Apply Tariff Message Sequence – Charge Reversed Due To Timeout



Apply Tariff Volume

The Apply Tariff Volume method enables an external system to apply a tariff to a specified volume of units for MO-MT, App-MT, and MO-App transactions. This method is similar to the Apply Tariff method with the addition of Volume, Unit Type, and Param1 and Param2 fields.

Apply Tariff Volume Request

The external service sends the Apply Tariff Volume Request message to the Event Charging Interface to charge a subscriber using the Comverse ONE solution tariff and rating engine, based on usage volume.

Table 18, “Apply Tariff with Volume Request Message” shows the contents of the Apply Tariff Volume Request message.

Table 18 Apply Tariff with Volume Request Message

Field	Size Format	Type	Mandatory / Optional	Value	Comment
Length	2 char	16 bits binary	M	178	This is the message size.
Message ID	1 char	8 bits binary	M	1	Request (01x)
Message Type	1 char	8 bits binary	M	11	APPLYTARIFF_VOLUME (05x)
Associated Number	4 chars	ASCII / binary	M		Value generated by the external service to uniquely identify this request
Originating Caller ID	30 chars	ASCII	M	12345678	MSISDN of Originating Subscriber (no porting prefix)
Originating Subscriber MSC Address	30 chars	ASCII	O		A-number location.
Subscriber Type	1 char	8 bits binary	M	1	1=MO-MT. 2=MO-App. 3=App-MT.
Terminating Caller ID	30 chars	ASCII	M	1234568	MSISDN of Terminating Party

Table 18 Apply Tariff with Volume Request Message (Continued)

Field	Size Format	Type	Mandatory / Optional	Value	Comment
Terminating Subscriber MSC Address	30 chars	ASCII	O		B-number location.
Bearer Capability	4 chars	ASCII	M	101	Values 0000–9999. Used with Discount and Unit Type to determine Application and Sub Type. Examples: HS to HS: 101 HS to VAS: 102 HS to Email: 103 SoB to HS: 104 SoB to Email: 105 SoB to VAS (future): 106 VAS to HS: 107 Email to HS:108
Discount	4 chars	ASCII	M	102	Values 0000–9999. Used with Discount and Unit Type to determine Application and Sub Type. Examples: Picture=101 ; Audio=102 ; Video=103 ; Text=104 ; Ringtone=105 ; Multi=106 ; Empty = 107; Other = 108
Volume	10 chars	ASCII	M	3000000	Volume (typically bytes used). Values 0–999999999 permitted. This is a required field for this method.
Unit Type	4 chars	ASCII	M	2	Unit of volume. This is mapped to a Comverse ONE solution Unit via the ECI Unit Type Mapping table. Allowable values are 0–9999. This is a required field for this method.
InfoParam1	32 chars	ASCII	O	ABCD	Information passed to the usage record but not used in rating. Left blank when not used. For Comverse MMSC, this field is the 1st 32 characters of the destination email address in this field.
InfoParam2	32 chars	ASCII	O	EFGH	Information passed to the usage record but not used in rating. Left blank when not used. For Comverse MMSC, the VAS ID and VAS Name is in this field.

Apply Tariff Volume Response

The Event Charging Interface sends the Apply Tariff Volume Response message to the external service in response to the Apply Tariff Request message.

A Status ID of 0 indicates that the account has been debited. Any other Status ID value indicates that no charge has been applied.

Table 19, “Apply Tariff with Volume Response Message” shows the contents of the Apply Tariff Volume Response message.

Table 19 Apply Tariff with Volume Response Message

Field	Size	Type	Value
Length	2 bytes	16 bits binary	15
Message ID	1 byte	8 bits binary	2=Response
Message Type	1 byte	8 bits binary	11=Apply Tariff Volume
Associated Number	4 bytes	ASCII/binary	The Associated Number received in the request.
Transaction ID	8 bytes	64 bits binary	This number is generated by the Event Charging Interface to uniquely identify this transaction. The external service must use this number in any follow-on messages such as Transaction Acknowledge of Reverse Charge.
Status ID	1 byte	8 bits binary	00x=Charge successfully applied to subscriber 01x=Invalid Subscriber ID (subscriber not found) 02x=Subscriber not active or not data enabled, charge not applied 03x=Service unavailable 04x=Subscriber has insufficient balance, partial charge taken 05x=Tariff engine error 06x=No tariffs found

Transaction ID Acknowledgement

Table 20, “Transaction ID Acknowledge Message” shows the contents of the Transaction ID Acknowedgement message.

Table 20 Transaction ID Acknowledge Message

Field	Size (bytes)	Type	Value	Hexidecimal Value
Length	2	16 bit binary	15	000Fx
Message ID	1	8 bit binary	1 = Request	01x
Message Type	1	8 bit binary	6 = Transaction ID Acknowledge	06x
Associated Number	4	ASCII / binary	The Associated Number received in the charge or tariff request.	00000002x – This is the number received in the original Apply Tariff Volume request message to which this is the acknowledgement.
Transaction ID	8	64 bit binary	Transaction ID as supplied by the Event Charging Interface in the response message.	0123456789ABCDFFx This is the number that the Event Charging Interface returned as part of the Apply Tariff response.
Status ID	1	8 bit binary	0 = Apply 1 = Reverse	00x This message acknowledges the Apply Charge/Tariff response. Reverse (01x) is only used if the external service must immediately reverse the charge. This is an unlikely scenario.

Calculate the Charge

The unified rater in the Comverse ONE solution billing model calculates the charge for the service. It first maps the received Bearer and Discount values into an Initial AUT via the Event Charging Interface Translation table (shown in the section "Event Charging Interface Provisioning in Product Catalog," on page 33).

The initial AUT is then mapped to a final AUT based on segmentation keys such as Location Relationships, using the location information provided in the Originating and Terminating MSC Address fields in the Apply Tariff message, or special features such as Friends and Family, Calling Circles, and so on.

The Final AUT is then used to determine the tariff plan (and associated tariffs) based on the subscriber's offer, as well as the date and time of the usage. Charges are then calculated based on the tariffs within the tariff plan.

Apply Tariff with Volume Message Sequences

Figure 8, "Typical Apply Tariff with Volume Message Sequence – Charge Not Reversed" through Figure 10, "Typical Apply Tariff with Volume Message Sequence – Charge Reversed Due To Timeout" show typical Apply Tariff with Volume message sequences.

Figure 8 Typical Apply Tariff with Volume Message Sequence – Charge Not Reversed

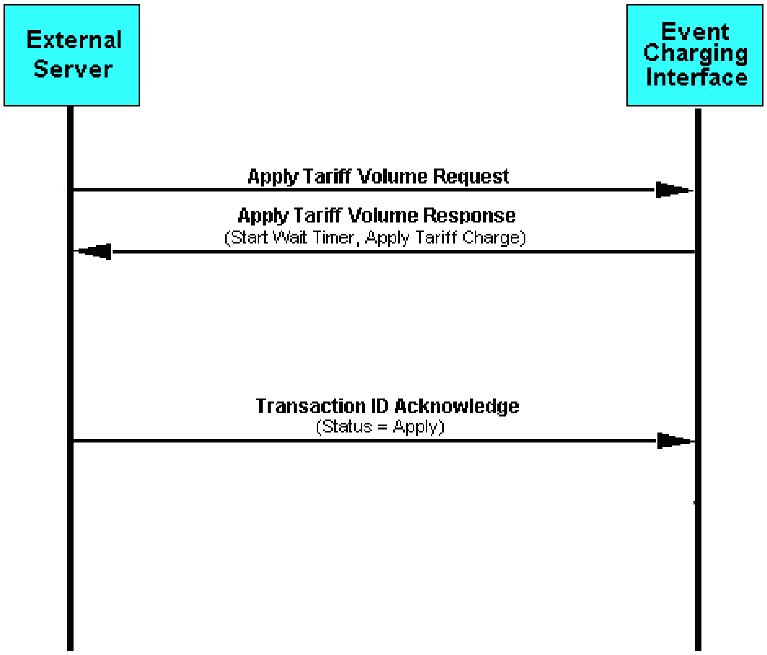

External Server
Event Charging Interface
Apply Tariff Volume Request
Apply Tariff Volume Response
(Start Wait Timer, Apply Tariff Charge)
Transaction ID Acknowledge
(Status = Apply)

Figure 9 Typical Apply Tariff with Volume Message Sequence – Charge Reversed by External Service

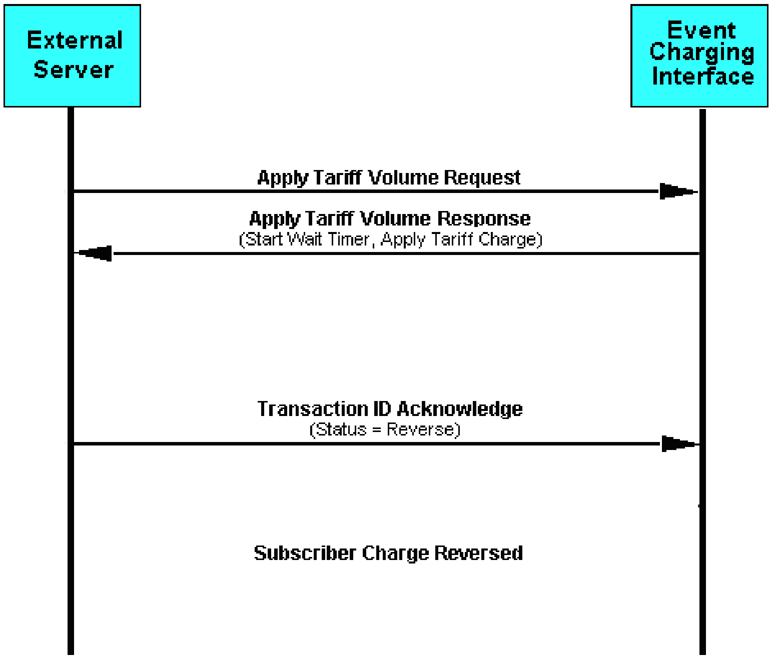
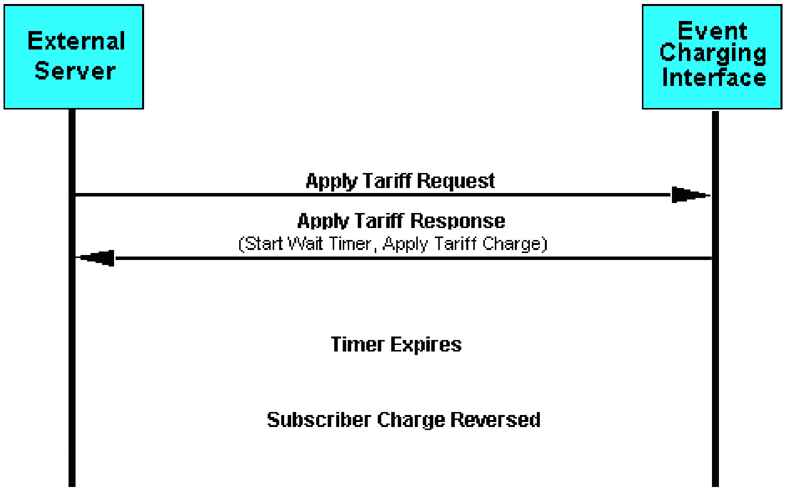

External Server
Event Charging Interface
Apply Tariff Volume Request
Apply Tariff Volume Response
(Start Wait Timer, Apply Tariff Charge)
Transaction ID Acknowledge
(Status = Reverse)
Subscriber Charge Reversed

Figure 10 Typical Apply Tariff with Volume Message Sequence – Charge Reversed Due To Timeout



Reverse Charges

The external service can request a reversal of a previous transaction, such as when an SMS message delivery fails.

Apart from the Transaction ID Acknowledge, (discussed in the section “Apply Charge,” on page 11), which can only be used to reverse the charge within a short time of the original charge (typically five minutes), the Reverse Charge method is used to reverse any previous transaction, based on the Transaction ID.

A transaction reversal cannot be undone. The Event Charging Interface only permits valid transactions to be reversed, and each transaction can only be reversed once.

NOTE Reverse Charge depends on the existence of transaction histories. If transaction histories are purged after five days, a transaction that is six days old cannot be reversed.

Reverse Charge Request

The external service sends the Reverse Charge Request message to the Event Charging Interface to reverse a previous transaction.

Table 21, “Reverse Charge Request Message” shows the contents of the Reverse Charge Request message.

Table 21 Reverse Charge Request Message

Field	Size (bytes)	Type	Mandatory / Optional	Value	Hexidecimal Value
Length	2	16 bit binary	M	14	000Ex
Message ID	1	8 bit binary	M	1 = Request	01x
Message Type	1	8 bit binary	M	5 = Reverse Charge	05x
Associated Number	4	ASCII / binary	M	A value generated by the external service to uniquely identify this request	00000003x
Transaction ID	8	64 bit binary	M	Transaction ID as supplied by the Event Charging Interface in the response to the charge/tariff request	0123456789ABCDFFx This is the number that the Event Charging Interface returned as part of the Apply Tariff response.

Reverse Charge Response

The Event Charging Interface sends a Reverse Charge Response message to the external service in response to the Reverse Charge Request.

The Event Charging Interface looks for an Event Charging Interface transaction history record with a matching Transaction ID.

If a match is found, the Event Charging Interface reverses the transaction, (meaning it refunds the charge amounts to the subscriber), creates an Event Charging Interface history record of the reversal, sends a positive response, and, optionally, creates a usage record.

If a match is not found, for example, if a transaction’s history record has been purged, a negative response is sent.

Table 22, “Reverse Charge Response Message” shows the contents of the Reverse Charge Response message.

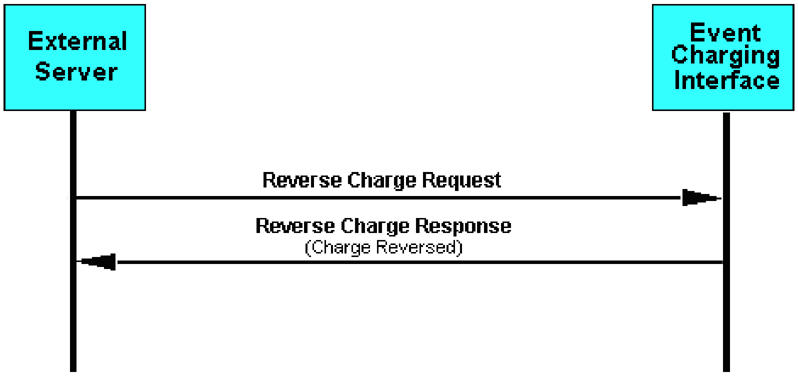
Table 22 Reverse Charge Response Message

Field	Size (bytes)	Type	Value	Hexidecimal Value
Length	2	16 bit binary	7	0007x
Message ID	1	8 bit binary	2 = Response	02x
Message Type	1	8 bit binary	5 = Reverse Charge	05x
Associated Number	4	ASCII / binary	The Associated Number that was received in the reverse request	00000003x – This is the number that was used in the Reverse Charge request message.
Status	1	8 bit binary	0 = Charge Reversed 7 = Transaction Not Found	00x 0 indicates that the transaction was reversed. 7 indicates that the Transaction ID was not found, so the transaction could not be reversed.

NOTE If a subscriber's currency is found to be different from the currency in use at the time of a transaction, the Comverse ONE solution uses the currency conversion rate that was in Apply Tariff.

Figure 11, "Typical Reverse Charge Message Sequence – Reverse Charge Request" shows a typical Reverse Charge messaging sequence:

Figure 11 Typical Reverse Charge Message Sequence – Reverse Charge Request



Disconnect Connection

Either side can send a Disconnect message to terminate the session.

- The Event Charging Interface sends the Disconnect message when a switchover from Active to Standby occurs.
- The external service can send the Disconnect message at any time.

The only response to this message is to tear down the TCP/IP session.

Table 23, "Disconnect Request Message" shows the contents of the Disconnect Request message.

Table 23 Disconnect Request Message

Field	Size (bytes)	Type	Value	Hexidecimal Value
Length	2	16 bit binary	2	0002x
Message ID	1	8 bit binary	1 = Request From external service 2 = Request From Event Charging Interface	01x or 02x depending on which side is sending the message
Message Type	1	8 bit binary	8 = Disconnect	08x

Invalid Message

If the Event Charging Interface receives an invalid message type, it replies with an Invalid Response message.

Table 24, "Invalid Response Message" shows the contents of the Invalid Response message.

Table 24 Invalid Response Message

Field	Size (bytes)	Type	Value	Hexidecimal Value
Length	2	16 bit binary	7	0007x
Message ID	1	8 bit binary	2=Response	02x
Message Type	1	8 bit binary	The Message Type that was received	The content of the fourth byte of the invalid message received by the Event Charging Interface.
Associated Number	4	ASCII / binary	Associated Number that was received	The content of the Associated Number field of the invalid message received by the Event Charging Interface.
Status ID	1	8 bit binary	8 = Invalid message Type	08x

Flow Control / Throttling

If the Event Charging Interface cannot handle requests because the Comverse ONE solution is too busy, it ignores Apply Charge, Apply Tariff, Validate Subscriber, and Reverse Charge requests to attempt to throttle interactions with the Comverse ONE solution application. The external service must resend these requests after the appropriate timeout period.

It is expected that overload conditions are transient and occur very infrequently.

Configuration Parameters

Table 25, "Event Charging Interface-Related Fields in the Service Parameters Table" shows the Service Parameters Table fields that must be configured.

Table 25 Event Charging Interface-Related Fields in the Service Parameters Table

Service Parameters Table Field	Default	Description
PSVR_AC_SV_CD	NULL	Event Charging Interface Apply Charge Service Code
PSVRCP_AC_SV_CD	NULL	Content Provider Apply Charge Service Code
PSVR_AC_CH_CD	NULL	Event Charging Interface Apply Charge Charge Code
PSVRCP_AC_CH_CD	NULL	Content Provider Apply Charge Charge Code

Table 25 Event Charging Interface-Related Fields in the Service Parameters Table (Continued)

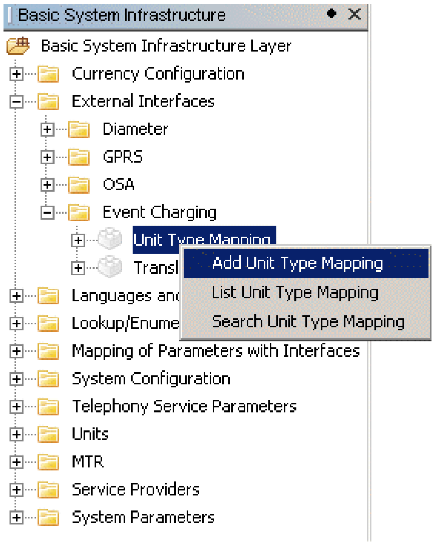
Service Parameters Table Field	Default	Description
PSVR_AC_CG_TP	1	Event Charging Interface Apply Charge Charge Type
PSVRCP_AC_CG_TP	1	Content Provider Apply Charge Charge Type
PSVRCP_AC_BR_TP	11	PMTSVRCP Apply Charge Bearer Type
PMT_CDR_UNSUCC	0	0=OFF 1=ON, Generate usage records for unsuccessful attempts

Event Charging Interface Provisioning in Product Catalog

To provision the event changing interface in the Product Catalog (PC), take the following steps:

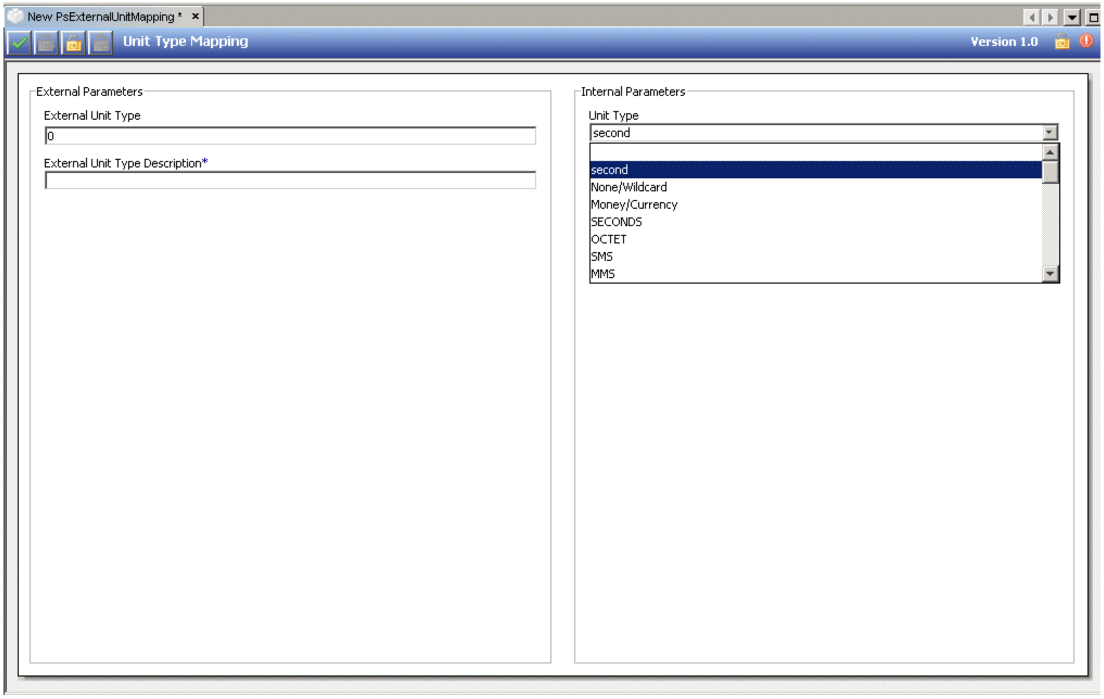
1. Log in to the Product Catalog.
2. In the TreeView area of the PC, click **Basic System Infrastructure tab**. The Basic System Infrastructure TreeView appears.
3. Basic System Infrastructure TreeView, click **External Interfaces**, then click **Event Charging**.
4. Right click Unit Type Mapping. A drop-down menu appears (Figure 12, “Unit Type Mapping Right Click Menu”).
5. Click Add Unit Type Mapping.

Figure 12 Unit Type Mapping Right Click Menu



6. The Event Charging Interface Unit Type Mapping window (Figure 13, “Event Charging Interface Unit Type Mapping Window”) appears.

Figure 13 Event Charging Interface Unit Type Mapping Window



External Parameters

External Unit Type: 0

External Unit Type Description*:

Internal Parameters

Unit Type: second

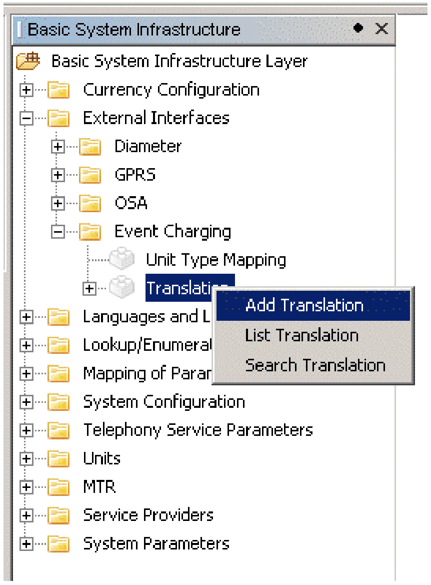
7. In the **External Unit Type** field, in the External Parameters area, enter a number that designates this unit type.
8. In the **External Unit Type Description** field, enter a brief description of this unit type.
9. In the **Unit Type** field in the Internal Parameters area, select a unit type from the drop-down list of defined unit types within the Comverse ONE solution. This internal unit type is then associated with the external unit type.

NOTE

It is possible to assign more than one External Unit Type to a single Internal Unit Type. For example, the External Unit Types ‘SMS MESSAGE FOR WAKE UP’ and ‘P_CMVRS_UNIT_SMS’ can both be assigned to a single Comverse ONE solution Unit Type, in this case, SMS.

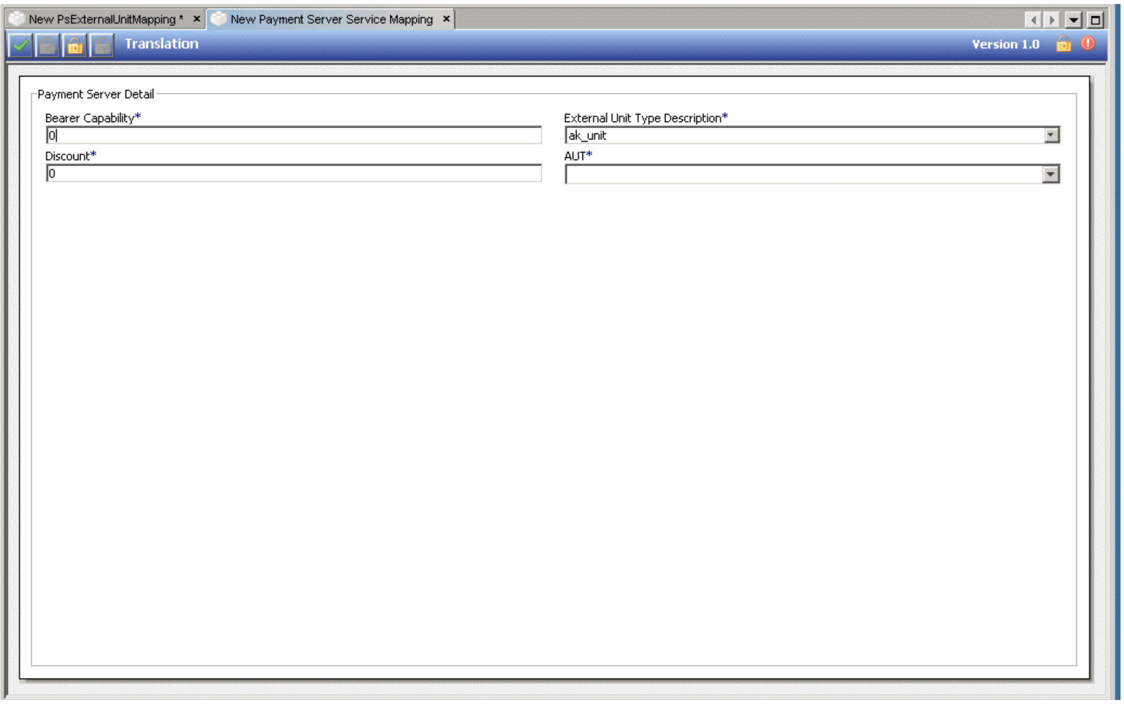
10. Click the Save icon on the toolbar or select Save from the Edit menu to save the new External Unit Type mapping.
11. In the Basic System Infrastructure TreeView, under **Event Charging Interface,** right click **Translation**. A dropdown menu appears (Figure 14, “Translation Right Click Menu”).

Figure 14 Translation Right Click Menu



12. Click Add Translation.
13. The Event Charging Interface Unit Translation window (Figure 15, “Event Charging Interface Translation Window”) appears.

Figure 15 Event Charging Interface Translation Window


New PsExternalUnitMapping *
New Payment Server Service Mapping
Translation
Version 1.0

Payment Server Detail

Bearer Capability* 0

External Unit Type Description* ak_unit

Discount* 0

AUT*

14. To define a new translation of Bearer Capability and Discount into an Initial AUT via the Event Charging Interface Translation table, complete the following fields:
    - **Bearer Capability**: numeric value, 0–9999
    - **Discount**: numeric value, 0–9999

- **External Unit Type Description:** select an External Unit Type from the drop-down list.
- **AUT:** select an Application / Unit Type from the drop-down list.

15. Click the Save icon on the toolbar or select Save from the Edit menu to save the new External Unit Type translation for billing in the selection list in the Event Charging Interface List Translation window.

Chapter 3

Open Services Access (OSA)

Overview

This chapter describes the specifications, interface details, and operation of the Comverse Open Services Access (OSA) Interface for Part 12 Charging as supported in Comverse ONE solution.

NOTE OSA is a premium option available with the Comverse ONE solution.

Comverse ONE solution supports the industry standard OSA charging interface, as defined by 3GPP and Parlay.

Target Audience

This chapter is intended primarily for programmers who must develop client programs that communicate with Comverse ONE solution through the OSA API. A brief overview of the Parlay/OSA model is provided, and some of the relevant explanatory texts, diagrams, and tables from the OSA specifications are reproduced. Refer to the actual standards and specifications from 3GPP and Parlay for the complete specifications.

Feature Description

The Parlay/OSA Charging API implements a strong set of features associated with the Comverse ONE solution, along with some additional rating capabilities. The operator has a wide range of flexibility to make charges based on simple to sophisticated algorithms using currency- or unit-based rating.

Comverse ONE solution does not support the Parlay/OSA Framework entity. This eases the requirement on application developers by avoiding the Framework interaction before interfacing to the Comverse Charging API. To protect the operator’s sensitive subscriber information, this release is restricted to work only within the trusted domain of the network operator’s IP network. There is no exposure of the charging API outside of the scope of the operator’s trusted systems.

References

1. ETSI TS 122.127 Universal Mobile Telecommunications Group (UMTS); Open Service Access (OSA); Stage 1 (Release 5)
2. ETSI TS 123.127 Universal Mobile Telecommunications Group (UMTS); Virtual Home Environment (VME)/Open Service Access (OSA); Stage 2 (Release 5)
3. ETSI TS 129.198-1 Universal Mobile Telecommunications Group (UMTS); Open Service Access (OSA) Application Programming Interface (API); Part 1: Overview (Release 5)
4. ETSI TS 129.198-2 Universal Mobile Telecommunications Group (UMTS); Open Service Access (OSA) Application Programming Interface (API); Part 2: Common Data (Release 5)
5. ETSI TS 129.198-12 Universal Mobile Telecommunications Group (UMTS); Open Service Access (OSA) Application Programming Interface (API); Part 12: Charging (Release 5)
6. The Unified Modeling Language User Guide – Grady Booch, James Rumbaugh, Ivar Jacobson
7. ISO 4217 (1995): Codes for the representation of currencies and funds http://www.iso.org/iso/support/faqs/faqs_widely_used_standards/widely_used_standards_other/currency_codes/currency_codes_list-1.htm
8. Web Services Description Language (WSDL) 1.1 http://www.w3.org/TR/wsdl
9. Simple Object Access Protocol (SOAP) 1.1 http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

Acronyms

API – Application Programmers Interface

CA – Client Application

CDR – Call Detail Record

HTTP – Hypertext Transfer Protocol

IP – Internet Protocol

OSA – Open Service Access

RPC – Remote Procedure Call

SOAP – Simple Object Access Protocol

SCF – Service Capability Feature

SCS – Service Capability Server

UML – Unified Modeling Language

URL – Universal Resource Locator

WSDL – Web Service Description Language

XML – Extensible Markup Language

Parlay/OSA

The Parlay/OSA service integrates telecom network capabilities with Internet Protocol (IP) applications via a secure, measured, and billable interface. The methodology for this interface was originally developed in the Parlay working group during the late 1990s. The 3GPP working group drew heavily on the Parlay 1.2 and Parlay 2.1 releases for the initial OSA Release 99 and subsequent OSA Release 4. Since that time, the various working groups have agreed to align all further releases of the specifications from Parlay 3.0 and OSA Release 5 and beyond. The ETSI specifications are aligned with the 3GPP versions and are also aligned with Parlay.

Parlay/OSA services enable third party applications to be hosted within a telecom operator's network and enable applications running on external application servers to offer their services to the operator's subscriber base via a gateway. In contrast to many earlier telecom models where the communication and service interaction between systems and applications were specified based on protocols, Parlay/OSA standards are based on an Application Programming Interface (API) model. Support for 2G, 2.5G and 3G networks is provided through the same API, enabling development of applications that are unaware of the underlying protocol.To transfer information between the peer entities of this model, Parlay/OSA is based on a set of open interface standards such as CORBA, IDL, Java, UML, and Web Services (SOAP, XML and WSDL). Comverse ONE solution supports the OSA/Parlay implementation of the Web Services Description Language (WSDL).

Parlay/OSA assumes that applications run on an application server and connect to the Parlay/OSA Gateway over a TCP/IP network. The application server invokes functionality through an API, which generates a Remote Procedure Call (RPC) interaction with a network service in the gateway. The network service operates on the request, evaluates the arguments, performs the requested operation, and returns a value to the invoking client in an RPC response.

In certain cases the processing of the request is not completed in the network service before the response is returned to the client application. The API request is evaluated for correct parameters and the network service responds with a void value to enable the client application to continue. When the processing of underlying requests is complete, the network service invokes an API request to the client application to indicate the result of the operation. The client application expects to receive this indication as confirmation that the operation processing has completed.

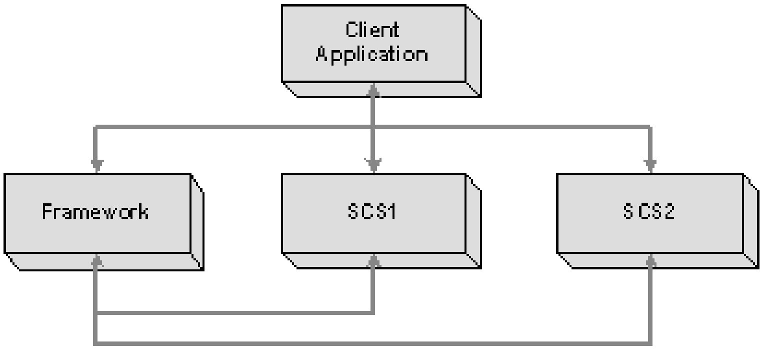
The Parlay/OSA model diverges from the standard client/server model defined in the RPC parlance. The interaction between client application and the network service is more of a peer-to-peer relationship.

The term used to refer to a program requesting network services from the gateway is application or Client Application (CA). A network service in the gateway consists of two components defined by the Parlay/OSA specifications:

- **Framework**: This entity is responsible for initial interaction with the Client Application, authorization and authentication, providing service descriptions, and other functions.
- **Service Capability Servers (SCS)**: The SCS is responsible to provide the actual service requested by the client application.

Figure 16, “Client Application, Framework, and SCS Relationship” shows the components of the Parlay/OSA model.

Figure 16 Client Application, Framework, and SCS Relationship



NOTE Comverse ONE solution does not support the Parlay/OSA Framework entity.

The requested service is referred to as the Service Capability Feature (SCF). Service Capability Features are defined by broad sets of functionality.

Parlay/OSA contains a wide range of SCFs, defined as APIs for client applications, which include:

- Call Control
- Charging
- Mobility
- Location
- Presence
- Availability Management
- User Interaction
- Messaging
- Policy Management

Each of these capabilities is defined as an individual SCF. The OSA Framework SCF is intended to provide a normalized, secure access to the rest of the SCFs from the client applications. These are logical abstractions and they do not necessarily imply separate processors, processes or infrastructures.

For a complete discussion of the Parlay/OSA interface please refer to references [1], [2], [3].

OSA Charging API

Client applications use the OSA Charging SCF to charge for the usage of an application service. The charging is either amount-based or unit-based depending on the operations invoked by the client application. Amount-based charging is always associated with currency and unit-based charging is associated with arbitrary unit types. In addition, unit-based charging implies that the Charging SCS applies a rating mechanism to determine the actual cost of the item(s) being charged (for example, one originated SMS message is rated at $0.10).

To provide the service, the OSA Charging specification [5] defines a set of methods and parameters that support both the amount- and unit-based charging types and are available to the client application. For each type of charging, two sets of charging methods are also defined. A charging method either directly updates the subscriber balance (Event-based charging) or makes a reservation of a certain amount and at some later time debits the amount by the actual consumption (Reservation-based charging).

As an example of Event-based charging, consider a client application associated with an SMS switching system. The system operator wishes to debit $0.10 from the subscribers account on the attempt to originate an SMS message. The client application invokes an OSA charging API method to directly debit the subscriber balance.

An example of Reservation-based charging is provided by a client application for a Wi-Fi Internet service configured to charge based on volume of bytes used. Since the duration of the connected session cannot be known ahead of time, the client application reserves one megabyte of usage for the subscriber at the start of the session. Once the subscriber session is established, the client application must invoke a debit method for the actual number of bytes consumed within the lifetime of the reservation, that is, before the reservation expires. There are no other restrictions on when the reserved amount is consumed. A debit could come in the middle of the session or at the end. If the client is not ready to send a debit, it can invoke the extendLifeTimeReq() method, which extends the lifetime of the reservation. For details on the extendLifeTimeReq() method refer to “Reservation Lifetime Extension Method: extendLifeTimeReq(),” on page 75. The actual decision to apply a charge is taken by the client application running on the external Wi-Fi server.

Parlay/OSA defines the API in an object-oriented manner. The Unified Modeling Language (UML) describes the relationships between the different entities of the API. UML defines a convention for communicating the type of functionality that is available from elements of a system and how these elements relate to one another. Drawings are typically used to express these ideas, where each shape represents some type of functional element and lines connecting the shapes represent interfaces and relationships. For a complete discussion of this subject please refer to [6].

The Parlay/OSA specifications refer to objects that are instantiated in either the client application or the SCF server side. Relationships are defined between these objects in how they interact with each other. The typical model is that a manager object has a relationship with a peer manager object on the remote system. Also, the manager object has relationships to many session objects within its own domain. Finally, each session object has a peer relationship to a session object on the remote system.

OSA Charging Classes and Interfaces

The manager object types, or classes, for the charging interface are the IpAppChargingManager in the client application and the IpChargingManager in the SCS. The IpAppChargingManager manages a set of session objects on the client application known as IpAppChargingSession. On the SCS, the corresponding session objects are the IpChargingSession, which are managed by the IpChargingManager. Each IpAppChargingSession object in the client application has a relationship to a corresponding IpChargingSession object on the SCS.

Convention in Parlay/OSA plays a part in the naming of the various object types, or classes.

- Classes that are prefixed with Ip are interface classes, meaning that there is an interface to another peer class on a remote system.
- Classes that start with the IpApp prefix are interface classes that reside in the client application.

Services are requested from the SCS by invoking methods in the IpChargingSession object. These methods interface through an RPC mechanism to the remote IpChargingSession object to process the request at the SCS. In quite a few instances, the request is not fully serviced by the SCS at the time the IpChargingSession method is invoked. In these cases, the IpChargingSession at the SCS evaluates the arguments to the call, responds to the invoke method with a null value, and continues processing the request. When the SCS has completed the operation the IpChargingSession object invokes a callback method through the RPC mechanism to the IpAppChargingSession.

Many of the operations on the SCFs are asynchronous in nature. As a result, client side applications are required to implement peer level callback interfaces and register these callback elements with SCF server side interfaces.

Request Number

When invoking methods that operate on the user's account (either directly or indirectly via reservations), a Request Number synchronizes the activity between the client application and IpChargingSession. This mechanism forces the client application to wait for a callback from the IpChargingSession object because the next Request Number is always provided as part of this response. It also makes retries possible after client application, service or communication errors. A retry of these operations is achieved by invoking the same operation with the same Request Number after the client application has timed out waiting for a response.

In the callback to the application, the Request Number to be used for the next request operation is returned. This is the only Request Number besides the one in the last request operation that is honored by the SCS. The use of the Request Number ensures that there is only one outstanding request per Charging Session. The client application issues the next request only after a response (result or error) that includes the Request Number for the next request is received. Only asynchronous operations that could lead to over- or under-charging of the user require a Request Number. Refer to the section “Request Number in Method Callbacks,” on page 57 for detailed information about request number usage.

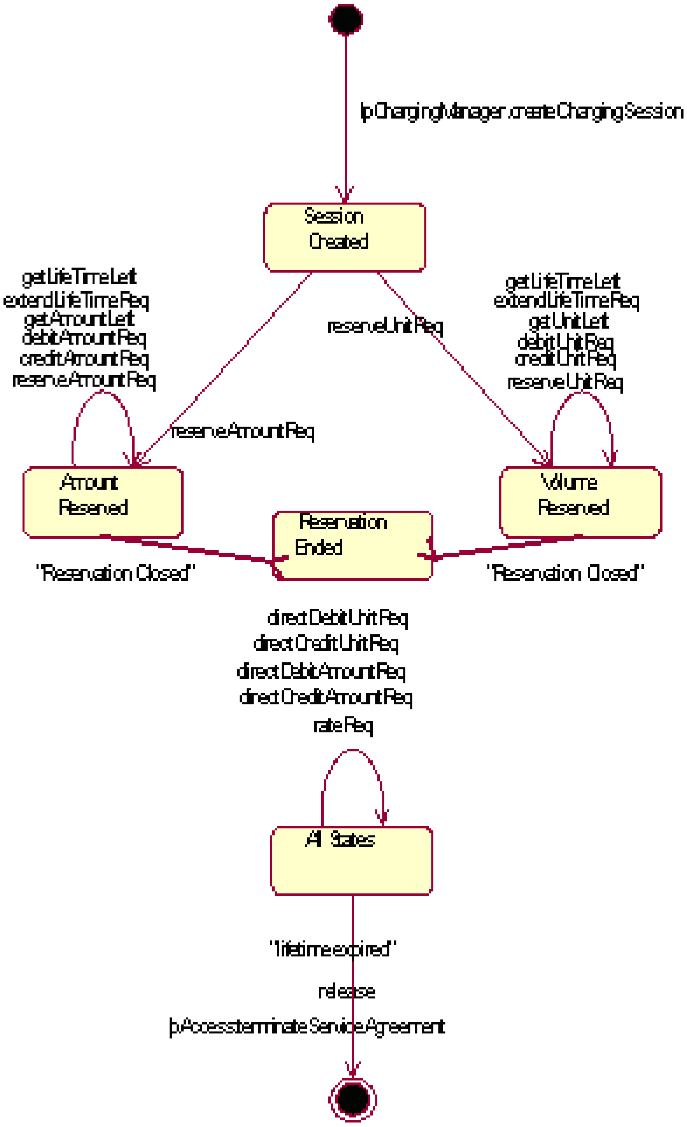
OSA Charging Session States

The Parlay/3GPP OSA specifications for charging APIs define four specific states for a charging session:

- Session Created state
- Amount Reserved state
- Volume Reserved state
- Reservation Ended state

Figure 17, “OSA Charging Session State Transitions” illustrates the state transitions with a UML state machine diagram. It shows the relationship between several of the available methods and the defined states.

Figure 17 OSA Charging Session State Transitions



Important features of this diagram include:

- The direct operations are permitted in all states.
- At a single point in time, a given charging session cannot have both an amount-based reservation and a unit-based reservation existing.
- Not illustrated, but documented in the method specification, is that the closeReservation flag in the reservation-based debit and credit operations indicates whether the reservation must be closed after the handling of the method. The client application can specify a True value for this flag to close an existing reservation. The state of the session returns to Session Created in this case.
- When the reservation lifetime expires, the session exits. The Comverse OSA API implementation provides notification to the IpAppChargingSession with the sessionEnded() method in this case.

Comverse OSA Charging SCS

The Comverse ONE solution implements, as a feature, the Charging Service Capability Server (SCS) to support the OSA Charging SCF.

The Comverse SCS provides three charging modes to offer variety in the ways a client application uses the Comverse ONE solution. The charging modes enable three levels of provisioning of the charges and associated tariffs. This feature provides significant flexibility in determining the tariff plan and charging modes for any given application. Refer to the section “Comverse OSA Charging Modes,” on page 48 for complete details.

The Comverse SCS deviates from the Parlay/OSA specification to only provide deployment without the Framework SCS. Security for this feature is maintained by only providing support to trusted systems in the domain of a private network or intranet.

The initial contact address is provisioned into the client application and the client application uses the services of the Comverse SCS without going through the authentication, authorization, and discovery process associated with the Framework SCS.

The initial release of the Comverse SCS provides support for many of the defined methods in the Parlay/OSA specification, however some methods are partially supported or not supported. The complete list of supported functionality is found in the section “Supported Methods,” on page 46.

SOAP/XML as the Protocol

The Comverse OSA Charging SCS supports a SOAP/XML-based RPC mechanism to communicate between the client application and the SCS. SOAP/XML requests and responses are transferred using the HTTP protocol over standard TCP/IP network connections.

The SOAP/XML payloads are defined in the Web Services Description Language (WSDL) definition provided in the section “Exceptions and Errors,” on page 64. The Comverse WSDL is based on the Parlay/OSA charging specifications and contains Comverse extensions and corrections to certain syntax errors in the 3GPP version of the WSDL. There are four WSDL files provided in the install base that are required for supporting OSA Charging:

- common_cc_data.wsdl
- cs.wsdl
- osa.wsdl
- ui_data.wsdl

These files are located in the SLU directory: **/home/apache/httpd-2.0.55/htdocs/wsdl**.

NOTE

When developing applications for use with the Comverse OSA Charging SCF, developers must use the Comverse WSDL. Applications built with the 3GPP version do not work with the Comverse SCF.

Developers of client applications use the WSDL to create an interface library in the development language of choice and can now build real time billing applications with Comverse ONE solution.

The WSDL and SOAP/XML support makes it possible to build client systems on a variety of platforms, running diverse operating systems, and using different programming languages. The advantage to the application developer is that it is now possible to leverage a vast set of programming talents and tools to quickly build new services.

Supported Methods

Figure 18, “IpChargingManager Supported Methods” shows the UML diagram for IpChargingManager, which supports two methods: createChargingSession() and setCallback(). It is recommended that client applications invoke the setCallback() method at the start of interaction with the Comverse SCS to register the application reference address with the Comverse SCS.

Figure 18 IpChargingManager Supported Methods

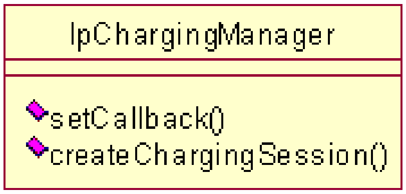


Figure 19, “IpChargingSession Supported Methods” shows the supported IpChargingSession methods. These methods support reservation-based charging and amount-based direct charging activities. For unit-based charging, only one unit type is supported for a specific charging session.

Figure 19 IpChargingSession Supported Methods

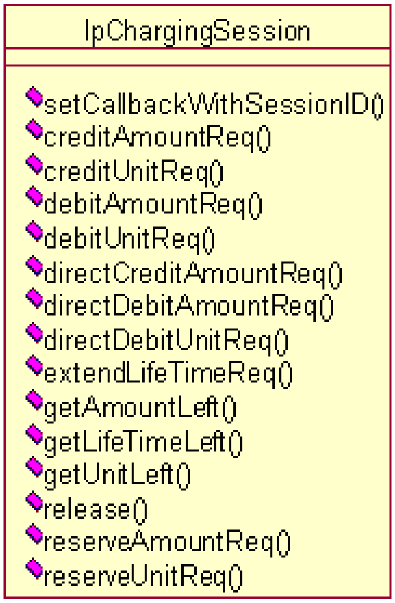


Figure 20, “IpAppChargingManager Required Methods”and Figure 21, “IpAppChargingSession Required Methods” list the required methods from the IpAppChargingManager and IpAppChargingSession respectively. The client application implements the IpAppChargingManager and IpAppChargingSession as part of conformance to the Parlay/OSA specification. These methods are invoked from the SCS by the objects instantiated from the IpChargingManager and IpChargingSession classes.

Figure 20 IpAppChargingManager Required Methods

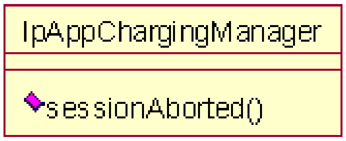
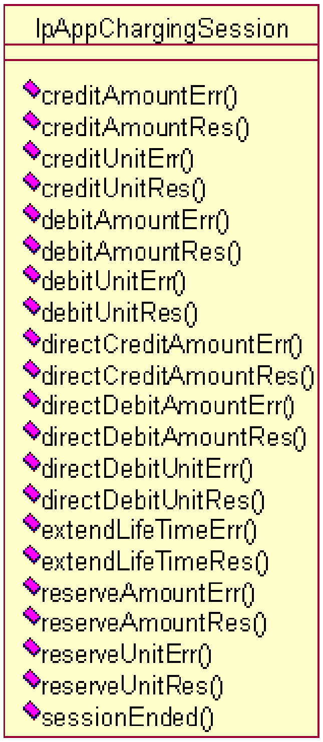


Figure 21 IpAppChargingSession Required Methods



IpAppChargingSession

- creditAmountErr()
- creditAmountRes()
- creditUnitErr()
- creditUnitRes()
- debitAmountErr()
- debitAmountRes()
- debitUnitErr()
- debitUnitRes()
- directCreditAmountErr()
- directCreditAmountRes()
- directDebitAmountErr()
- directDebitAmountRes()
- directDebitUnitErr()
- directDebitUnitRes()
- extendLifeTimeErr()
- extendLifeTimeRes()
- reserveAmountErr()
- reserveAmountRes()
- reserveUnitErr()
- reserveUnitRes()
- sessionEnded()

Unsupported Methods

Several Parlay/OSA methods are not supported in the Comverse SCS. The complete list is:

- In the IpChargingManager class
  - createSplitChargingSession()
- In the IpChargingSession class
  - directCreditUnitReq()
  - rateReq()

Reservation and Session Lifetime

In the Parlay/OSA Charging specification, reservations are specified to have a lifetime assigned by the SCS at the start of the reservation. The duration of the lifetime is returned to the client application in the reservation response message. The Comverse SCS maintains a timer for the reservation lifetime and closes a session when the timer has expired. When the timer expires, the SCS invokes the IpAppChargingSession method sessionEnded() to notify the client application that the session is closed. The client application can always extend the lifetime of a valid reservation through the IpChargingSession method extendLifeTimeReq().

Reservation lifetime is provisioned as part of the tariffing structure in the Comverse ONE solution.

NOTE Refer to Product Catalog User Guide for a description of the provisioning of the Comverse ONE solution.

The Comverse SCS also implements a session lifetime that is applicable when no reservation exists for the session. The session lifetime begins once the session is created and is intended to ensure that sessions are always closed at the SCS when no activity is received. The default session lifetime is set to two minutes. This value can be provisioned at the platform. Unlike the reservation lifetime, this timer cannot be extended by the client application explicitly. It is reset after all operations for a session.

Comverse OSA Charging Modes

Comverse OSA Charging SCS supports a concept known as a charging mode. The charging mode is not defined as an OSA data type, rather, it is a term used to convey the different levels of sophistication and customization possible with Comverse OSA Charging SCS. A Charging Mode represents a set of capabilities available to the client application and provisioning system to define different levels of rating complexity. The Basic mode enables charges to be made directly without any rating whatsoever and the most sophisticated mode supports location-based billing for the location of one or two parties involved in the transaction.

NOTE

The charging modes defined by Comverse use extension parameters to the TpChargingParameterSet. Detailed descriptions of these parameters are provided in the section “Comverse Charging Mode Parameters,” on page 60.

A charging mode is invoked by the client application based on the initial URL used to contact the Comverse Charging SCS to create the charging session. Based on the charging mode, the set of charging parameters in the TpChargingParameterSet used in the argument is evaluated to provide enhanced tariff selection. The Comverse Charging SCS defines several extensions to the base charging parameter set provided by Parlay/OSA. The Basic charging mode uses the OSA-defined charging parameters, while other charging modes are designated as increasingly sophisticated supersets, with additional charging parameters defined by Comverse.

The Comverse OSA Charging SCS supports three charging modes:

- Basic
- Extended
- Mobility Plus

The purpose of the charging mode is to provide the most appropriate set of functionality to the client application and operator as determined by the real world requirements of the application. For some applications, the Basic charging mode is perfect for their needs, while other more sophisticated applications require the additional features offered by the Extended or Mobility Plus modes. The type of charges applied and the flexibility to define the various charging types is available to the operator and to client applications.

Basic Charging Mode

The Basic charging mode provides the simplest interface to the Comverse Charging SCS and enables the client application to make reservation-based charges using the units or amounts and also amount-based direct charges.

Unit-based reservations and charges always use the rating capabilities of the Comverse ONE solution. When a unit reservation is made, the TpChargingParameterSet argument is evaluated to determine the tariff plan to use. Once the tariff plan is determined, the TpVolumeType is evaluated to ensure that the unit type is permitted for the specified tariff. After verification of the tariff parameters, a reservation is made against the subscriber balance(s).

To properly calculate the rating and provide fraud control, the merchant ID provided in the Parlay/OSA request must be provisioned in the Comverse Charging SCS. This provides the base lookup for tariff information. This also provides the means to define the same or different tariffs for one merchant ID versus another.

For unit-based methods, the argument that identifies the unit type and amount is known as the TpVolumeSet. This parameter is defined as a set of unit type/amount pairs, or multiple units in the same reservation. Unit types are identified by the Parlay/OSA parameter TpUnitID, which defines a basic set of unit types (for example, octets, seconds). The Comverse Charging SCS enables extension of the supported unit types through the Product Catalog provisioning interface. This means that the operator defines virtually any unit that is desired and develops a tariff plan for support of these units. The Comverse SCS only provides for a single entry in the TpVolumeSet per reservation, however multiple amounts of the unit are supported (that is, a reservation of 100 seconds is supported while a reservation for 100 seconds and 1MB of octets is not supported).

In Basic charging mode, only the P_CHS_PARAM_ITEM and P_CHS_PARAM_SUBTYPE charging parameter IDs are evaluated to determine the tariff plan. The operator defines the value of the data passed within these parameters and these are used to differentiate between various rating plans. If neither of these parameters is present, then it is also possible to define and revert to a default tariff plan.

As an example of a Basic mode unit-based charge, an operator wants to charge for the number of octets of music downloaded. The reservation is made in the unit type octets and the P_CHS_PARAM_ITEM indicates that this is a download operation and the P_CHS_PARAM_SUBTYPE indicates that the activity is music. A specific tariff is defined for this activity and the rate of $0.10 per 60K octets is applied when calculating charges. Perhaps the operator has defined a balance of download octets and the tariff plan specifies that this balance be consumed before charges are made to any currency balance.

In contrast to unit-based charges, amount-based charging provides no rating. The client application is always expected to provide the rating and all reservations and debits are made in currency. The currency type is specified in the reservation request and must conform to the currency permitted for the identified subscriber’s Primary Offer. Charges can be applied against any currency balance.

Amount-based charges are supported in both reservation-based and direct debit/credit-based schemes. While no rating is available for amount-based operations, the activity type is determined from the TpChargingParameterSet argument and provided in CDRs and activity histories for use in post-processing accounting.

As an example of a direct amount-based charge, the operator supports a point of sale terminal at a retail location. The subscriber makes some type of purchase and the amount is directly debited from the subscriber’s core balance. The directDebitReq() method does not supply any more information than that this is a retail charge.

As an example of an amount-based reservation, the operator provides some type of telecommunication service that provides rating information. The operator wishes to bill for this service in real time. The external system makes an amount-based reservation at the start of the session and while the session is in progress, the external system makes debits against this reservation. After the session is closed, the subscriber’s core balance is updated with the remainder of the reserved amount.

Extended Charging Mode

The Extended charging mode provides all of the same capabilities as the Basic charging mode with additional functionality for more elaborate rating based on extension parameters defined by the user.

The Extended charging mode defines extensions to the TpChargingParameterID data type that provide an additional means to identify the value to be rated. These parameters are used to provide additional information used in rating. Similar to the P_CHS_PARAM_ITEM and P_CHS_PARAM_SUBTYPE parameters, the operator has the ability to define the values that are passed in the service parameters to further refine the rating structure. Four generic service parameters and one Quality of Service (QoS) extension parameter are defined.

Operators use the additional parameters to identify a specific item in inventory or to provide further rating realization. The intent of the extra parameters is to provide the most flexible means to identify the items to be rated. The values of the service parameters are also committed to the CDRs.

Mobility Plus Charging Mode

The Mobility Plus charging mode builds on the Extended and Basic charging modes with the ability to provide location information from the client application. The location information is defined as additional parameters to the TpChargingParameterID, comprising two pieces of location information, location A and location B.

Location Relationships (LR) are provisioned in the Comverse ONE solution that correspond to values provided in the A and B location parameters. The location parameters are then used to further define the tariff that is applied for any given request.

Location information is supplied as a series of digits and a location type indicator. Both location A and B are accompanied by the respective location types. The location types for A and B do not need to be the same to define an LR. For instance, one could define an LR from a cell ID to a destination number in the E.164 number plan.

The intent of the Mobility Plus charging mode is to provide a means to apply tariffs to location sensitive charges. As an example, the operator wishes to charge MMS requests from an international roaming subscriber differently than a subscriber in the home location. The reservation of the MMS unit is made with location information parameters. The Comverse SCS evaluates these parameters and selects a tariff based on the current LR of the request. Thus, the subscriber can be charged a different rate for message requests originated in a home location than those originated from a roaming location.

OSA Client Application

The Client application, in the context of the Comverse OSA Charging SCS, is a web services client that follows the SOAP/HTTP binding. This means that the client application must, at a minimum, implement the callback interfaces IpAppChargingManager and IpAppChargingSession.

SOAP/XML Binding to HTTP

The Parlay/OSA WSDL specifies that the SOAP/XML messages are carried via, or bound to, the HTTP protocol. This is the standard means by which to bind a WSDL service description to an underlying transport.

Client applications use the HTTP POST request to initiate contact with the Comverse SCS. For the initial session creation, the POST request contains the URL of the IpChargingManager, which is provided to the client application by the operator (Comverse SCS does not support Framework interaction).

OSA method return values are transferred to the client application in an HTTP status response. The response to the session creation contains a value known as the ChargingSessionReference of type IpChargingSessionRef. The ChargingSessionReference is the URL address of the charging session and must be used to address the Comverse SCS in all subsequent interactions on this session.

When a client implements the SOAP HTTP binding for callback, it means that the client application acts as an HTTP server responding to an HTTP POST request with an HTTP status response. The interface is said to be bi-directional in this case. Callback methods are used for asynchronous interaction with the Comverse SCS. When the client application issues an asynchronous request, the Comverse SCS evaluates the parameters of the request and, if all is found in order, responds to the client application with an HTTP status response. The Comverse SCS then proceeds to process the client application request. Once processing is complete, the Comverse SCS invokes a callback method, which is carried via an HTTP POST request, to the client application. The client application is expected to handle this post request and respond with the appropriate HTTP status response (that is, HTTP/1.1 200 OK).

All the interface references are URLs in the SOAP HTTP binding. When using the reference to communicate with a specific instance, the reference is not only part of the SOAP-Header portion of the message, but is also used as the URL to address the service.

OSA API Parameters

Comverse Charging SCS Use of Standard OSA Parameters

The OSA Charging SCF specifications contain definitions for many relevant data types and parameter definitions, however use of these parameters is not fully defined and is left up to the vendor implementation. This section identifies the parameters used by the Comverse Charging SCS and how they affect processing.

For a full definition of the defined OSA parameters please refer to [3], [4], and [5].

IpAppChargingSessionRef

Defines a reference to type IpAppChargingSession. For Comverse OSA Charging SCS, this is the URL of the IpAppChargingSession associated with the new session being created.

API Primitive Types

Table 26, “API Primitive Types” lists the five primitive types used in OSA API calls.

Table 26 API Primitive Types

Primitive Name	Primitive Type
TpInt32	IntValue
TpFloat	FloatValue
TpString	StringValue
TpBoolean	BooleanValue
TpOctetSet	OctetValue

TpSessionID

Defines a session ID with a value that is at least unique within the context of a specific instance of an SCF. The session ID is identical to a TpInt32 type.

TpAmount

Defines the sequence of data elements that define an amount in integers as Number * 10 ^ Exponent (for example, if Number = 6543 and Exponent = -2 then the amount is 65.43). This representation avoids unwanted rounding off.

Table 27 TpAmount Data Elements

Sequence Element Name	Sequence Element Type
Number	TpInt32
Exponent	TpInt32

TpChargingPrice

Defines the sequence of data elements that identify a price.

Table 28 TpChargingPrice Data Elements

Sequence Element Name	Sequence Element Type
Currency	TpString
Amount	TpAmount

NOTE Currencies as defined by ISO 4217:1995 [7]

OSA Charging actions are directed by the client applications by supplying the Charging Amounts, Charging Volumes, and Charging Parameters to the OSA SCS. Charging Amount is represented by a TpChargingPrice data type, Charging Volume is given as a TpVolumeSet data type, and Charging Parameter by TpChargingParameterSet data type.

TpChargingParameterSet

Defines a numbered set of data elements of type TpChargingParameter.

TpChargingParameter

Defines a sequence of data elements that defines the used service.

Sequence Element Name: Sequence Element Type

ParameterID: TpChargingParameterID

ParameterValue: TpChargingParameterValue

TpChargingParameterID

Please refer to the section "Comverse Charging Mode Parameters," on page 60 for a complete discussion of this parameter.

TpChargingParameterValue

Defines the tagged choice of data elements that identifies a charging parameter.

TpChargingParameterValueType

Defines the type of charging parameter.

Table 29 TPChargingParameterValueType Values

Name	Value	Description
P_CHS_PARAMETER_INT32	0	Parameter represented by a TpInt32
P_CHS_PARAMETER_FLOAT	1	Parameter represented by a TpFloat
P_CHS_PARAMETER_STRING	2	Parameter represented by a TpString
P_CHS_PARAMETER_BOOLEAN	3	Parameter represented by a TpBoolean
P_CHS_PARAMETER_OCTETSET	4	Parameter represented by a TpOctetSet

TpVolumeSet

Defines the numbered set of data elements that describes list TpVolume.

TpVolume

Defines a volume.

Table 30 TP Volume Data Elements

Sequence Element Name	Sequence Element Type
Amount	TpAmount
Unit	TpUnitID

TpUnitID

Defines the unit used in a TpVolume. This type can be extended with operator specific items.

Table 31 TpUnitID Values

Name	Value	Description
P_CHS_UNIT_UNDEFINED	0	Undefined
P_CHS_UNIT_NUMBER	1	number of times/events
P_CHS_UNIT_OCTETS	2	unit is octets
P_CHS_UNIT_SECONDS	3	unit is seconds
P_CHS_UNIT_MINUTES	4	unit is minutes
P_CHS_UNIT_HOURS	5	unit is hours
P_CHS_UNIT_DAYS	6	unit is days

NOTE The TpUnitID can be extended through the Product Catalog provisioning system of the Comverse Charging SCS, however the values defined in Table 32, “TpApplicationDescription Data Elements” must not be modified.

TpApplicationDescription

Defines a sequence of data elements that specifies what is about to be charged.

Table 32 TpApplicationDescription Data Elements

Sequence Element Name	Sequence Element Type
Text	TpString
AppInformation	TpAppInformationSet

TpAppInformationSet

Defines a numbered set of data elements that further describe what is about to be charged. The data elements are of type TpAppInformation.

TpAppInformation

Defines a tagged choice of data elements that comprises an individual application’s information.

TpAppInformationType

Defines the possible information items.

Table 33 TpAppInformation

Sequence	Type	Description
SwitchName	TpAppInformationType	Identifies application information to use (only P_APP_INF_TIMESTAP is defined).

Table 34 TpAppInformationType to Choice Mapping

Tag Element Value	Choice Element Type	Choice Element Name
P_APP_INF_TIMESTAMP	TpDateAndTime	Timestamp

NOTE

Comverse OSA Charging SCS expects the client application to provide time stamp information. Comverse OSA Charging SCS uses this information to determine the tariff to be applied.

TpMerchantAccountID

Defines a sequence of data elements that defines the used service.

Table 35 TpMerchantAccountID Data Elements

Sequence Element Name	Sequence Element Type
MerchantID	TpString
AccountID	TpInt32

NOTE

The TpMerchantAccountID member AccountID values passed by the Client Application must be provisioned in the Comverse ONE solution. If the passed parameter values do not match any of the valid provisioned values, Comverse OSA SCS issues P_INVALID_ACCOUNT exception.

TpCorrelationID

Defines the sequence of data elements that identifies a correlation.

Table 36 TpCorrelationID Data Elements

Sequence Element Name	Sequence Element Type
CorrelationID	TpSessionID
CorrelationType	TpCorrelationType

TpCorrelationType

Defines the type of correlation. This type can be extended with operator specific items.

Table 37 TpCorrelationType Values

Name	Value	Description
P_CHS_CORRELATION_UNDEFINED	0	Unknown correlation type.
P_CHS_CORRELATION_VOICE	1	Voice Call
P_CHS_CORRELATION_DATA	2	Data Session
P_CHS_CORRELATION_MM	3	Multi Media Session

NOTE The TpCorrelationID parameter value does not affect any of the rating/charging logic in Comverse ONE solution. However, these are reflected in CDRs and call histories.

TpAddress

Defines the sequence of data elements that specifies an address.

Table 38 TpAddress Data Elements

Sequence Element Name	Sequence Element Type
Plan	TpAddressPlan
AddrString	TpString
Name	TpString
Presentation	TpAddressPresentation
Screening	TpAddressScreening
SubAddressString	TpString

TpAddressPlan

Defines the address plan (or numbering plan) used. It is also used to indicate whether an address is actually defined in a TpAddress data element.

Table 39 TpAddressPlan Data Elements

Name	Value	Description
P_ADDRESS_PLAN_NOT_PRESENT	0	No Address Present
P_ADDRESS_PLAN_UNDEFINED	1	Undefined
P_ADDRESS_PLAN_E164	5	E.164

For the case where P_ADDRESS_PLAN_NOT_PRESENT and P_ADDRESS_PLAN_ANY are indicated, the rest of the information in the TpAddress is not valid.

The AddrString defines the actual address information and the structure of the string depends on the plan.

Table 40, “AddrString Formats Supported” shows the format of the AddrString for the P_ADDRESS_PLAN_E164 address plan.

Table 40 AddrString Formats Supported

Address Plan	AddrString Format Description	Example
P_ADDRESS_PLAN_E164	An international number without the international access code, including the country code and excluding the leading zero of the area code.	"31161249111"

NOTE Only P_ADDRESS_PLAN_E164 is supported in Comverse OSA Charging SCS.

TpAddressSet

Defines a numbered set of data elements of TpAddress.

TpAddressPresentation

NOTE This parameter is not evaluated by the Comverse Charging SCS.

TpAddressScreening

NOTE This parameter is not evaluated by the Comverse Charging SCS.

TpAddressError

Defines the reasons why an address is invalid.

Table 41 TpAddressError Data Elements

Name	Value	Description
P_ADDRESS_INVALID_UNDEFINED	0	Undefined error
P_ADDRESS_INVALID_MISSING	1	Mandatory address not present
P_ADDRESS_INVALID_MISSING_ELEMENT	2	Mandatory address element not present
P_ADDRESS_INVALID_OUT_OF_RANGE	3	Address is outside of the valid range
P_ADDRESS_INVALID_INCOMPLETE	4	Address is incomplete
P_ADDRESS_INVALID_CANNOT_DECODE	5	Address cannot be decoded

TpURL

This data type is identical to a TpString and contains a URL address. The usage of this type is distinct from TpAddress, which can also hold a URL. TpAddress contains a user address which is specified in many ways: IP, e-mail, URL, and so on. The TpURL type does not hold the address of a user and always represents a URL. This type is used in user interaction and defines the URL of the test or stream sent to an end-user. It is therefore inappropriate to use a general address here.

TpChargingSessionID

Defines the sequence of data elements that unambiguously specifies the Charging Session object.

Table 42 TpChargingSessionID Data Elements

Sequence Element Name	Sequence Element Type	Sequence Element Description
ChargingSessionReference	IpChargingSessionRef	Specifies the interface reference for the charging session object: a URL for Comverse OSA Charging SCS.
ChargingSessionID	TpSessionID	Specifies the session ID for the charging session. This value must be supplied in all method invocations for this session.
RequestNumberFirstRequest	TpInt32	This element specifies the Request Number to use for the next request.

TpChargingError Data Type

Indicates the error that occurred.

Table 43 TpChargingError Data Types

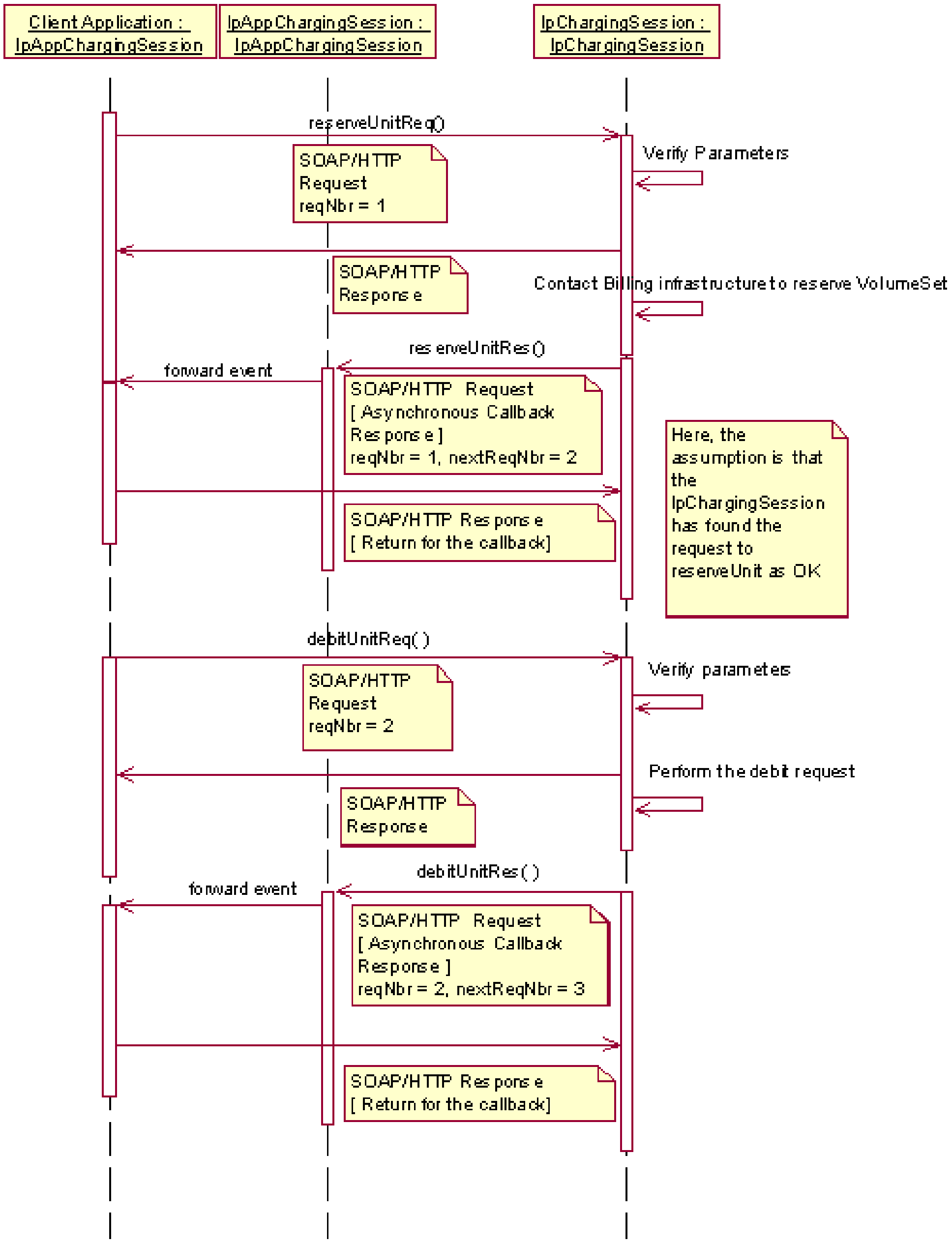
Name	Value	Description
P_CHS_ERR_UNDEFINED	0	Generic error
P_CHS_ERR_ACCOUNT	1	Merchant account unknown
P_CHS_ERR_USER	2	Unknown user
P_CHS_ERR_PARAMETER	3	The set of charging parameters contains an unknown parameter, or a required parameter is missing.
P_CHS_ERR_NO_DEBIT	4	The application is not permitted to get money from this user.
P_CHS_ERR_NO_CREDIT	5	The application is not permitted to pay this user.
P_CHS_ERR_VOLUMES	6	Required volumes are missing.
P_CHS_ERR_CURRENCY	7	This currency is not supported for this transaction.
P_CHS_ERR_NO_EXTEND	8	Request to extend the lifetime of a reservation is rejected.
P_CHS_ERR_RESERVATION_LIMIT	9	This amount or volume violates the bounds of the reservation
P_CHS_ERR_CONFIRMATION_REQUIRED	10	A user confirmation is required, but couldn't be obtained by the SCS. The SCS expects that the client initiates a stored confirmation scenario.

Request Number in Method Callbacks

Normal Method Sequence

Many methods in the OSA deliver the result in the callback method invocation. The Request Number argument of the invoking method serves the purpose of coordinating the original request with the callback response. Figure 22, “Use of Request Numbers, Normal Operation” shows how the Request Number is used in a reservation and debit request under normal circumstances.

Figure 22 Use of Request Numbers, Normal Operation


Client Application : IpAppChargingSession
IpAppChargingSession : IpAppChargingSession
IpChargingSession : IpChargingSession
reserveUnitReq()
SOAP/HTTP Request reqNbr = 1
Verify Parameters
SOAP/HTTP Response
Contact Billing infrastructure to reserve VolumeSet
reserveUnitRes()
forward event
SOAP/HTTP Request [Asynchronous Callback Response] reqNbr = 1, nextReqNbr = 2
Here, the assumption is that the IpChargingSession has found the request to reserveUnit as OK
SOAP/HTTP Response [Return for the callback]
debitUnitReq( )
SOAP/HTTP Request reqNbr = 2
Verify parameters
Perform the debit request
SOAP/HTTP Response
debitUnitRes( )
forward event
SOAP/HTTP Request [Asynchronous Callback Response] reqNbr = 2, nextReqNbr = 3
SOAP/HTTP Response [Return for the callback]

NOTE The initial Request Number is returned as part of the TpSessionID value from the createChargingSession method.

The sequence diagram in Figure 22, “Use of Request Numbers, Normal Operation” illustrates that the requestNumber received in the callback matches the requestNumber of the original request. The nextRequestNumber of the callback contains the value to be used for the requestNumber argument in a subsequent method invocation.

The standard behavior for handling of the Request Number in the Comverse SCS system is summarized as follows:

- After the session is created, only one Request Number is honored by the Comverse SCS: the value returned in the TpSessionID result of the createChargingSession().
- After the first successful request, the Comverse SCS honors only two Request Numbers at any time. The two numbers are the values of the requestNumber and nextRequestNumber arguments in the last callback request.
- When the requestNumber argument does not match an expected value, the Comverse SCS returns the exception **P_INVALID_REQUEST_NUMBER.**
- The Comverse SCS does not attempt retries of callback requests.
- Due to network delay, the client application must always test the requestNumber argument in the callback and discard any duplicates.

Method Sequence for Retransmission

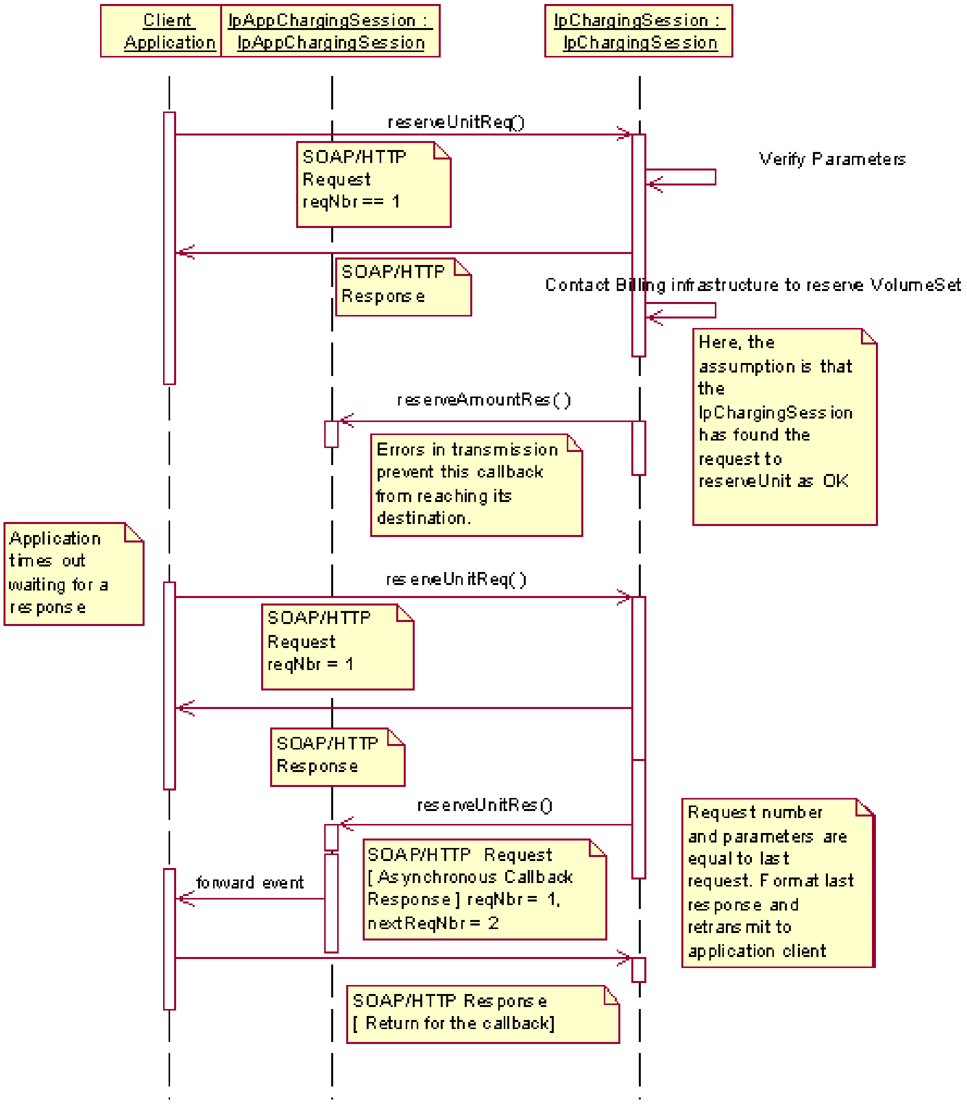
The sequence diagram shown in Figure 23, “Retransmission Scenario” illustrates the behavior of the client application and Comverse SCS when a callback request is lost. In this situation, the client application times out waiting for a response from the Comverse SCS and issues the same request with the previous Request Number. The Comverse SCS reconstructs the last request and sends this to the client application.

The Comverse SCS has built in timers to respond to the application timer in case of failures of internal subsystems. The default value of this timer is 10 seconds. It is important for reliable operation that the client application’s response timer be set for a value larger than the Comverse SCS response timer.

NOTE

Due to network delays it is possible that a callback response to a method is delayed beyond the client application’s response timer. This can lead to the situation of the original callback response being returned to the client application after the retransmission request has been made, and a subsequent callback with the same Request Number. The client application needs to test for a discard duplicate callback responses.

Figure 23 Retransmission Scenario


Client Application
IpAppChargingSession : IpAppChargingSession
IpChargingSession : IpChargingSession
reserveUnitReq()
SOAP/HTTP Request reqNbr == 1
Verify Parameters
SOAP/HTTP Response
Contact Billing infrastructure to reserve VolumeSet
Here, the assumption is that the IpChargingSession has found the request to reserveUnit as OK
reserveAmountRes()
Errors in transmission prevent this callback from reaching its destination.
Application times out waiting for a response
reserveUnitReq()
SOAP/HTTP Request reqNbr = 1
SOAP/HTTP Response
reserveUnitRes()
Request number and parameters are equal to last request. Format last response and retransmit to application client
SOAP/HTTP Request [Asynchronous Callback Response] reqNbr = 1, nextReqNbr = 2
forward event
SOAP/HTTP Response [Return for the callback]

Comverse Charging Mode Parameters

The various charging modes defined by Comverse use extension parameters to the TpChargingParameterSet to provide the necessary additional information. Each charging mode supports a specific set of parameters that are evaluated relative to the specific charging mode. Parameters from a more advanced set that are present in a less capable session are ignored.

Table 44, “Basic Mode Supported TpChargingParameterID Names,” Table 45, “Extended Mode – Supported TpChargingParameterID Names,” and Table 46, “Mobility Plus Mode – Supported TpChargingParameterID Names” describe the additional parameters and how they are evaluated by the SCS for charging, along with a column that indicates if the parameter is mandatory in the identified mode.

Table 44 Basic Mode Supported TpChargingParameterID Names

Name	Value	Description	Affect Rating / Charging?	Mandatory?
P_CHS_PARAM_UNDEFINED	0	Unknown parameter	FALSE	N
P_CHS_PARAM_ITEM	1	Parameter represents kind of service delivered to the end user	TRUE	Y
P_CHS_PARAM_SUBTYPE	2	Parameter represents Sub Type/operation of service delivered to the end user	TRUE	Y
P_CHS_PARAM_CONFIRMATION_ID	3	The ID that references a stored confirmation to authorize the required payment	FALSE	N
P_CHS_PARAM_CONTRACT	4	Parameter represents a signed confirmation of the type P_CHS_ PARAMETER_ OCTETSET	FALSE	N
P_CMVRS_PARAM_TIMEZONE_OFFSET	100	Offset from UTC Time	TRUE	N

Table 45 Extended Mode – Supported TpChargingParameterID Names

Name	Value	Description	Affect Rating / Charging?	Mandatory?
P_CHS_PARAM_UNDEFINED	0	Unknown parameter	FALSE	N
P_CHS_PARAM_ITEM	1	Parameter represents kind of service delivered to the end user	TRUE	Y
P_CHS_PARAM_SUBTYPE	2	Parameter represents Sub Type/operation of service delivered to the end user	TRUE	Y
P_CHS_PARAM_CONFIRMATION_ID	3	The ID that references a stored confirmation to authorize the required payment	FALSE	N
P_CHS_PARAM_CONTRACT	4	Parameter represents a signed confirmation of the type P_CHS_ PARAMETER_ OCTETSET	FALSE	N
P_CMVRS_PARAM_TIMEZONE_OFFSET	100	Offset from UTC Time	TRUE	N
P_CMVRS_PARAM_QoS	101	Parameter represents the Quality of Service Identifier.Type P_CHS_ PARAMETER_STRING	TRUE	N
P_CMVRS_SERVICE_PARAM_1	102	Generic Service Parameter. Must follow the provisioned agreements on type and values.	TRUE	N

Table 45 Extended Mode – Supported TpChargingParameterID Names (Continued)

Name	Value	Description	Affect Rating / Charging?	Mandatory?
P_CMVRS_SERVICE_PARAM_2	103	Generic Service Parameter. Must follow the provisioned agreements on type and values.	TRUE	N
P_CMVRS_SERVICE_PARAM_3	104	Generic Service Parameter. Must follow the provisioned agreements on type and values.	TRUE	N
P_CMVRS_SERVICE_PARAM_4	105	Generic Service Parameter. Must follow the provisioned agreements on type and values.	TRUE	N
P_CMVRS_INFORMATIONAL_PARAM	106	Generic information parameter. Must be of type TpString and is committed to the CDR	FALSE	N

Table 46 Mobility Plus Mode – Supported TpChargingParameterID Names

Name	Value	Description	Affect Rating / Charging?	Mandatory?
P_CHS_PARAM_UNDEFINED	0	Unknown parameter	FALSE	N
P_CHS_PARAM_ITEM	1	Parameter represents kind of service delivered to the end user	TRUE	Y
P_CHS_PARAM_SUBTYPE	2	Parameter represents Sub Type/operation of service delivered to the end user	TRUE	Y
P_CHS_PARAM_CONFIRMATION_ID	3	The ID that references a stored confirmation to authorize the required payment	FALSE	N
P_CHS_PARAM_CONTRACT	4	Parameter represents a signed confirmation of the type P_CHS_PARAMETER_OCTETSET	FALSE	N
P_CMVRS_PARAM_TIMEZONE_OFFSET	100	Offset from UTC Time	TRUE	N
P_CMVRS_PARAM_QoS	101	Parameter represents the Quality of Service Identifier. Type P_CHS_PARAMETER_STRING	TRUE	N
P_CMVRS_SERVICE_PARAM_1	102	Generic Service Parameter. Must follow the provisioned agreements on type and values.	TRUE	N

Table 46 Mobility Plus Mode – Supported TpChargingParameterID Names

Name	Value	Description	Affect Rating / Charging?	Mandatory?
P_CMVRS_SERVICE_PARAM_2	103	Generic Service Parameter. Must follow the provisioned agreements on type and values.	TRUE	N
P_CMVRS_SERVICE_PARAM_3	104	Generic Service Parameter. Must follow the provisioned agreements on type and values.	TRUE	N
P_CMVRS_SERVICE_PARAM_4	105	Generic Service Parameter. Must follow the provisioned agreements on type and values.	TRUE	N
P_CMVRS_INFORMATIONAL_PARAM	106	Generic information parameter. Must be of type TpString and is committed to the CDR	FALSE	N
P_CMVRS_PARAM_SUBSCRIBER_LOCATION	107	Location Parameter.Type P_CHS_PARAMETER_STRING	TRUE	N
P_CMVRS_PARAM_SUBSCRIBER_LOCATION_TYPE	108	Location Parameter.Type TPINT32	TRUE	N
P_CMVRS_PARAM_LOCATION_B	109	Location Parameter.Type P_CHS_PARAMETER_STRING	TRUE	N
P_CMVRS_PARAM_LOCATION_B_TYPE	110	Location Parameter.Type TPINT32	TRUE	N
P_CMVRS_PARAM_IMSI_MIN	111	IMSI or MINType P_CHS_PARAMETER_STRING	FALSE	N

Location Determination in Mobility Plus Mode

Four TpChargingParameters support Location Based billing. These are:

- P_CMVRS_PARAM_SUBSCRIBER_LOCATION
- P_CMVRS_PARAM_SUBSCRIBER_LOCATION_TYPE
- P_CMVRS_PARAM_LOCATION_B
- P_CMVRS_PARAM_LOCATION_B_TYPE

The parameters P_CMVRS_PARAM_SUBSCRIBER_LOCATION and P_CMVRS_PARAM_LOCATION_B contain a string of digits to identify the location of the subscriber. The interpretation of these digits is based on the corresponding LOCATION_TYPE parameters. Each digit string and type identifies a location of either the subscriber or the called (B) party. The locations are compared to form a location relationship, and the location relationship further refines the tariff calculation.

Table 47, “Location Type Values” identifies the valid values used for location type.

Table 47 Location Type Values

Name	Value	Description
HANDSET_MAP	1	Handset
TELEPHONE	2	Dialed number
MSRN	3	MSRN ID
CELL_ID	4	CELL ID
MSC_ID	5	MSC ID
SGSN_ID	6	SGSN ID
IP_ADDRESS	7	IP Address (Used with DCMP)

Exceptions and Errors

Exceptions

All OSA methods are capable of reporting exceptions that result from:

- Errors in the arguments provided with the operation
- The state or content of the provisioning
- Internal conditions that prevent the operation from succeeding.
- Exceptions are reported to the client application in the SOAP-FAULT envelope. Two types of exceptions exist: TpCommonExceptions and standard exceptions. While both the TpCommonException and standard exception are delivered in the SOAP-FAULT envelope, the format of the encoded data is slightly different.

Figure 24, “Standard Exception encoding” and Figure 25, “TpCommonException encoding” demonstrate the difference between the encoding of the two exception types. Notice that each example is a SOAP-FAULT with the Server fault type.

Figure 24 Standard Exception encoding

```
<?xml version="1.0" encoding="UTF-8"?>
 <SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Body>
   <SOAP-ENV:Fault>
    <faultcode>
      SOAP-ENV:Server
             </faultcode>
    <faultstring>
      Server Error
    </faultstring>
    <detail>
     <cs:P_INVALID_ACCOUNT xmlns:cs="http://www.csapi.org/cs/wsdl">
      <ExtraInformation>
        Merchant Account not valid !
      </ExtraInformation>
     </cs:P_INVALID_ACCOUNT>
    </detail>
   </SOAP-ENV:Fault>
  </SOAP-ENV:Body>
 </SOAP-ENV:Envelope>
```

Figure 25 TpCommonException encoding

```
<?xml version="1.0" encoding="UTF-8"?>
 <SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
  <SOAP-ENV:Body>
   <SOAP-ENV:Fault>
    <faultcode>
      SOAP-ENV:Server
    </faultcode>
    <faultstring>
      Server Error
    </faultstring>
    <detail>
     <osa:TpCommonExceptions xmlns:osa="http://www.csapi.org/osa/wsdl">
      <ExceptionType type="osaxsd:TpInt32">
        744
      </ExceptionType>
      <ExtraInformation>
        P_INVALID_STATE-State does not allow Amount Debits
      </ExtraInformation>
     </osa:TpCommonExceptions>
    </detail>
   </SOAP-ENV:Fault>
  </SOAP-ENV:Body>
 </SOAP-ENV:Envelope>
```

HTTP binding in the SOAP recommendation dictates that the SOAP-FAULT result is delivered to the client application with the HTTP response code of 500 Internal Server Error, while normal responses are delivered to the client application with the HTTP response code of 200 "OK.”

The OSA standard defines a limited number of exception codes. Detailed information about the cause of an exception can be found in the ExtraInformation element in the exception definition. The following sections describe the Exception type, the text in the ExtraInformation portion, and a short description on the cause of the fault.

TpCommonExceptions

Table 48, “TpCommonExceptions” identifies the set of TpCommonExceptions provided by the Comverse SCS and information about the cause of the exception. The ExtraInformation element, the TpCommonException name, followed by a hyphen, is prefixed to the string listed in the ExtraInformation column.

Table 48 TpCommonExceptions

TpCommonException	ExtraInformation	Description
P_RESOURCE_UNAVAILABLE	Cannot create context	Internal error indicating the context creation failed.
P_RESOURCE_UNAVAILABLE	Database failure	An internal error has occurred that has made the database inaccessible.
P_RESOURCE_UNAVAILABLE	Cannot create new session[, …]	An internal error prevents the session from being created. Additional information is present after a comma. Typically caused by reaching limits of system capacity.
P_RESOURCE_UNAVAILABLE	Failed to update callback URL	The update of the IpAppChargingManager callback address failed.
P_RESOURCE_UNAVAILABLE	Cannot allocate memory for CDR	Memory constraints within the system prevent the direct method from building a CDR.
P_RESOURCE_UNAVAILABLE	Cannot load account associated data	Internal error prevents the session from being created successfully.
P_TASK_REFUSED	OSA decode error	Error in decoding one or more of the components of the received message.
P_TASK_REFUSED	Invalid TpChargingParameter Id [%d] Type [%d]	A received message contained a parameter with the TpChargingParameterValueType set to an unsupported type.
P_TASK_REFUSED	OSA feature not enabled	The OSA feature is not configured in the platform.
P_TASK_REFUSED	Max. concurrent sessions reached	The platform has reached the maximum number of concurrent sessions that it is configured to handle.
P_TASK_REFUSED	No Manager object for specified URL	The URL of the createChargingSession does not match any of the provisioned URLs.
P_TASK_REFUSED	Invalid appChargingSession	The createChargingSession argument appChargingSession is not present in the request.
P_TASK_REFUSED	Out of Session IDs	Internal error that prevents the session from being created.
P_TASK_REFUSED	Subscriber not in Active or PostActive State	The subscriber is not in the correct state to accept OSA charging requests.
P_TASK_REFUSED	Activity not allowed in current subscriber state	The subscriber is not permitted to perform OSA operations due to being inactive or restrictions in the PO.
P_TASK_REFUSED	Subsequent Reserve Params must match first reservation	When extending a reservation, the provided TpChargingParameterSet does not match the values of the previous reservation.
P_TASK_REFUSED	Maximum number of reservations exceeded	The current reserve request exceeds the maximum number of reservations for this session .
P_TASK_REFUSED	Tariffing Failed	Failed to obtain/resolve the charging parameters and unit type/currency to a tariff.

Table 48 TpCommonExceptions (Continued)

TpCommonException	ExtraInformation	Description
P_TASK_REFUSED	Tariffing Failed – Location information service not invoked	The resolved tariff requires location information and the Mobility Plus charging mode was not invoked.
P_TASK_REFUSED	ERROR: OSA is NOT based on access!	The option Access Method in the sub type translation is not set to NONE.
P_TASK_REFUSED	Failed to obtain tariff	The tariff did not fully resolve. This is typically due to a mismatch in the location information provided and provisioned.
P_TASK_CANCELED	Operation timed out	The method timed out based on internal timers while waiting for a subscriber lookup.
P_TASK_CANCELED	Failed to get update from URE	Internal error preventing the operation from completing.
P_TASK_CANCELED	Failed to send update message to URE	Internal error preventing the operation from completing.
P_TASK_CANCELED	Failed to query account	Internal error preventing the operation from completing.
P_METHOD_NOT_SUPPORTED	Method not supported	The invoked method is one of the methods not supported by the Comverse SCS.
P_INVALID_STATE	Parameter Sequence not provisioned	TpChargingParameters provided as an argument of the method argument do not match provisioned values.
P_INVALID_STATE	State cannot receive volume activities	The session is in an opposite state for the current request (for example, received a reserveUnitReq in state AmountReserved).
P_INVALID_STATE	State cannot receive amount activities	The session is in an opposite state for the current request (for example: received a reserveAmountReq in state UnitReserved).
P_INVALID_STATE	No reservation active	A request to operate on a reservation was received while no reservation is active.
P_INVALID_STATE	State does not allow Volume Debits	A debitUnitRequest() was received while not in the unitReserved state.
P_INVALID_STATE	State does not allow Amount Debits	A debitAmountRequest() was received while not in the amountReserved state.
P_INVALID_STATE	State does not allow Volume Credits	A creditUnitRequest() was received while not in the unitReserved state.
P_INVALID_STATE	State does not allow Amount Credits	A creditAmountRequest() was received while not in the amountReserved state.
P_INVALID_STATE	Amount not Reserved	A getAmountLeft() method was invoked while not in the amountReserved state.
P_INVALID_STATE	Units not reserved	A getUnitLeft() method was invoked while not in the unitReserved state.

Standard Exceptions

Table 49, “Standard Exceptions” identifies the set of standard exceptions provided by the Comverse SCS and information about the cause of the exception.

Table 49 Standard Exceptions

Exception	Extra Information	Description
P_INVALID_ACCOUNT	Merchant Account not valid	The account number of the Merchant ID parameter does not match any of the provisioned values.
P_INVALID_CURRENCY	Invalid Currency specified	The ISO 4217 currency specified is not provisioned into the OSA Charging service.
P_INVALID_AMOUNT	Invalid preferredAmount or minAmount. Must be > 0	The preferred or minimum amount in a reserveAmountReq is less than or equal to zero
P_INVALID_AMOUNT	Invalid Amount Request. Must be > 0	The amount argument specifies a value less than or equal to zero (0).
P_INVALID_VOLUME	Invalid Volume Request. Must be > 0	In the reserveUnitReq the requested volume was less than or equal to zero (0).
P_INVALID_VOLUME	Invalid requested unit	The requested unit type of a debit or credit does not match the reserved unit type.
P_INVALID_VOLUME	Invalid requested volume	The requested volume of a credit is less than zero (0) or is greater than the previously debited amount.
P_INVALID_SESSION_ID	Session ID is not valid	The session ID used in a method from the IpChargingSession object does not match any of the known active session IDs.
P_INVALID_REQUEST_NUMBER	Invalid Request Number	The requestNumber argument to one of the IpChargingSession methods is not the expected value.
P_INVALID_USER	Invalid user number	The user (or subscriber) account is not provisioned in the Comverse ONE solution.

Chapter 4

OSA Method Descriptions

Overview

This chapter contains descriptions of the OSA methods implemented by the Comverse Charging SCS. It includes sequence diagrams that illustrate various interactions of SCS with the client application. The notation used is derived from the Unified Modeling Language (UML), which defines the Parlay/OSA specifications. For a further discussion of UML, please refer to [6].

Charging Session Creation Method: createChargingSession()

This method creates an instance of the IpChargingSession interface to handle the charging events related to the specified user and to the application invoking this method. An IpAppChargingManager must already have been passed to the IpChargingManager, otherwise the charging manager is not able to report a sessionAborted() to the application (the application can invoke setCallback() if it wishes to ensure this).

Returns:

chargingSession: Defines the session.

Signature:

createChargingSession (appChargingSession : in IpAppChargingSessionRef,

sessionDescription : in TpString, merchantAccount : in TpMerchantAccountID,

user : in TpAddress, correlationID : in TpCorrelationID) : TpChargingSessionID

Parameters:

- appChargingSession : in IpAppChargingSessionRef

  Callback interface for the session in the application. URL of the application object callback interface.
- sessionDescription : in TpString

  Descriptive text for informational purposes. This value is committed to the CDR.
- merchantAccount : in TpMerchantAccountID

  Identifies the account of the party providing the application to be used. In the Comverse system, the Merchant account must be provisioned in the Comverse ONE solution.
- user : in TpAddress

  Specifies the subscriber who is using the application. In the Comverse OSA system, this is the user that is charged. When this method is invoked, the charging service determines if charging is permitted for this application for this subscriber. If this type of charging is not permitted, an error is reported.
- correlationID : in TpCorrelationID

  This value is used to correlate the charging to network activity.

Returns:

- TpChargingSessionID

  TpChargingSessionID is a complex object that contains several members. These are:

  - ChargingSessionReference

    Reference to the charging session on the server, IpChargingSessionRef . The base type of this member is a string value. This value is the URL used by the client application to address the session when invoking the IpChargingSession methods.
  - ChargingSessionID

    Charging session ID as a 32-bit number passed as TpSessionID. This value is the first argument to all IpChargingSession methods.
  - RequestNumberFirstRequest
  - Request number, of type TpInt32, to be used in the first subsequent IpChargingSession method invocation. The Request Number is provided by the SCS and used by the client application to ensure the operations remain synchronized. Only one request can be outstanding at any given time.

Raises:

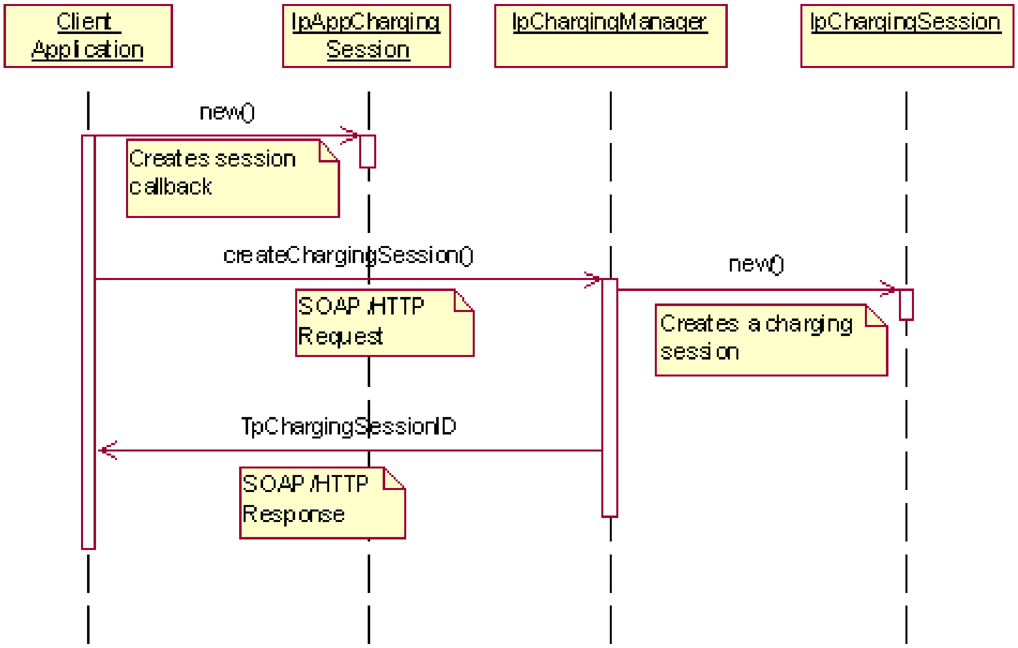
- **TpCommonExceptions**: Common exceptions that typically represent platform errors. Refer to the section "TpCommonExceptions," on page 65.
- **P_INVALID_USER**: The user argument does not specify a valid subscriber ID or the primary Offer does not permit OSA interaction for the specified subscriber.
- **P_INVALID_ACCOUNT**: The merchantAccount parameter does not specify a valid merchant ID.

createChargingSession() Successful Operation

The client application uses the IpChargingManager URL to create a charging session.

This is a synchronous call, shown in Figure 26, "Create Charging Session."

Figure 26 Create Charging Session



Session Release Method: release()

The release() method releases the session and all of its resources. No operations can be conducted with this session after this method is invoked and the sessionID becomes unrecognized. Unused parts of any outstanding reservation are returned to the subscriber's available balance and a CDR is generated for the session. This is a synchronous call that completes its operation before returning to the caller.

Signature:

release (sessionID : in TpSessionID, requestNumber : in TpInt32) : void

Parameters:

- sessionID: in TpSessionID

  The ID of the session.
- requestNumber : in TpInt32

  Specifies the number given in the result of the previous operation on this session, or when creating the session.

Returns:

void: no value

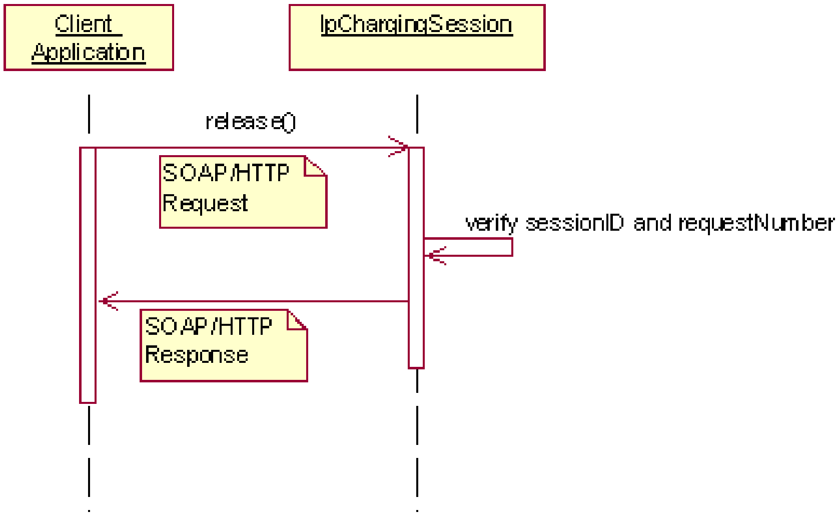
Raises:

- **TpCommonExceptions**: Common exceptions that typically represent platform errors. Please refer to the description in the section "TpCommonExceptions," on page 65.
- **P_INVALID_SESSION_ID**: The sessionID parameter contains a value that is not for a currently known session.
- **P_INVALID_REQUEST_NUMBER**: The requestNumber argument does not contain the expected value, the value that was returned from the previous callback response or from the createChargingSession() method.

release() Successful Operation

Figure 27, "Successful Release" shows the succesful operation of the release() method.

Figure 27 Successful Release



Reservation Lifetime Extension Method: extendLifeTimeReq()

This is an asynchronous call. The response is issued as a callback.

With this method an application can request that the lifetime of the reservation be extended. If no reservation has been made on the charging session, this method raises a TpCommonException (P_TASK_REFUSED).

Signature:

extendLifeTimeReq (sessionID : in TpSessionID) : void

Parameters:

- sessionID : in TpSessionID

  The ID of the current session to which the reservation applies.

Returns:

void: no value

Raises:

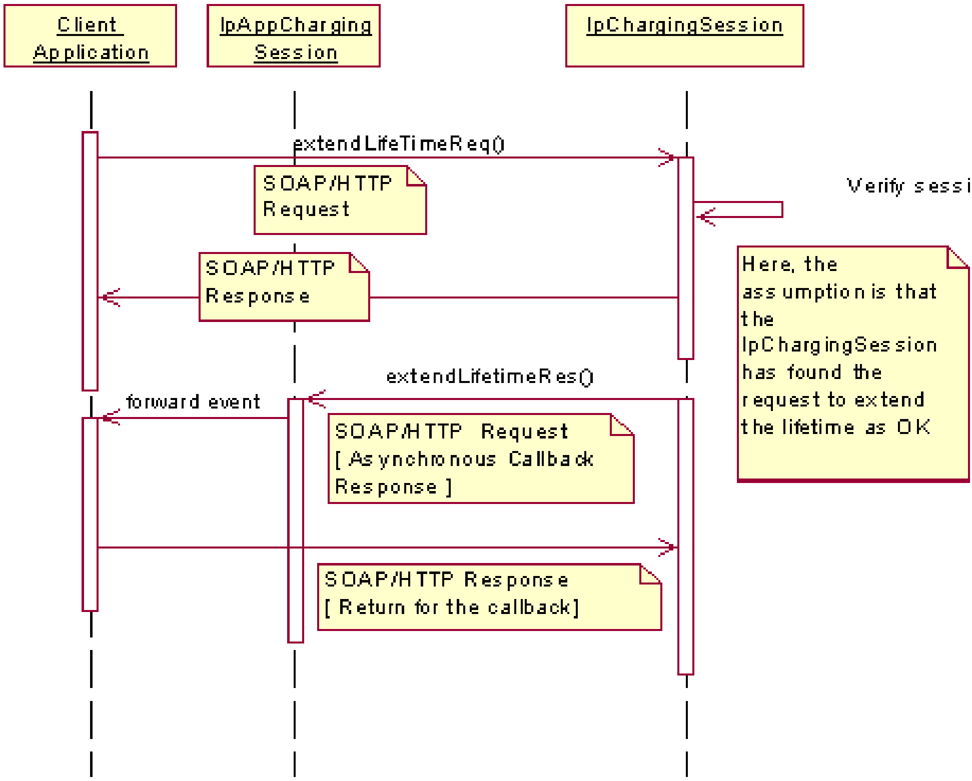
- **TpCommonExceptions**: Common exceptions that typically represent platform errors. Please refer to the description in the section "TpCommonExceptions," on page 65.

- **P_INVALID_SESSION_ID**: The sessionID parameter contains a value that is not for a currently known session.

extendLifeTimeReq() Successful Operation

When making any reservation in the Charging SCS there is always a reservation lifetime associated with the session. During the period when the reservation is active, it is possible for the client application to extend the lifetime of the reservation so that it expires further in the future. When extending the lifetime of the reservation the reserved amount does not change. Figure 28, “extendLifeTimeReq – Successful Operation” shows the succesful operation of the extendLifeTimeReq() method.

Figure 28 extendLifeTimeReq – Successful Operation


Client Application
IpAppCharging Session
IpChargingSession
extendLifeTimeReq()
SOAP/HTTP Request
Verify sessionID
SOAP/HTTP Response
Here, the assumption is that the IpChargingSession has found the request to extend the lifetime as OK
extendLifetimeRes()
forward event
SOAP/HTTP Request [Asynchronous Callback Response]
SOAP/HTTP Response [Return for the callback]

Callback Method: extendLifeTimeRes()

This is a callback to the asynchronous call extendLifeTimeReq(). This method is called to indicate that the corresponding request is successful. It also indicates the amount of seconds in the revised reservation lifetime.

Signature:

extendLifeTimeRes (sessionID : in TpSessionID, sessionTimeLeft : in TpInt32) : void

Parameters:

- sessionID : in TpSessionID

  The ID of the current session to which the reservation applies.
- sessionTimeLeft : in TpInt32

  The number of seconds that the reservation lifetime has been extended.

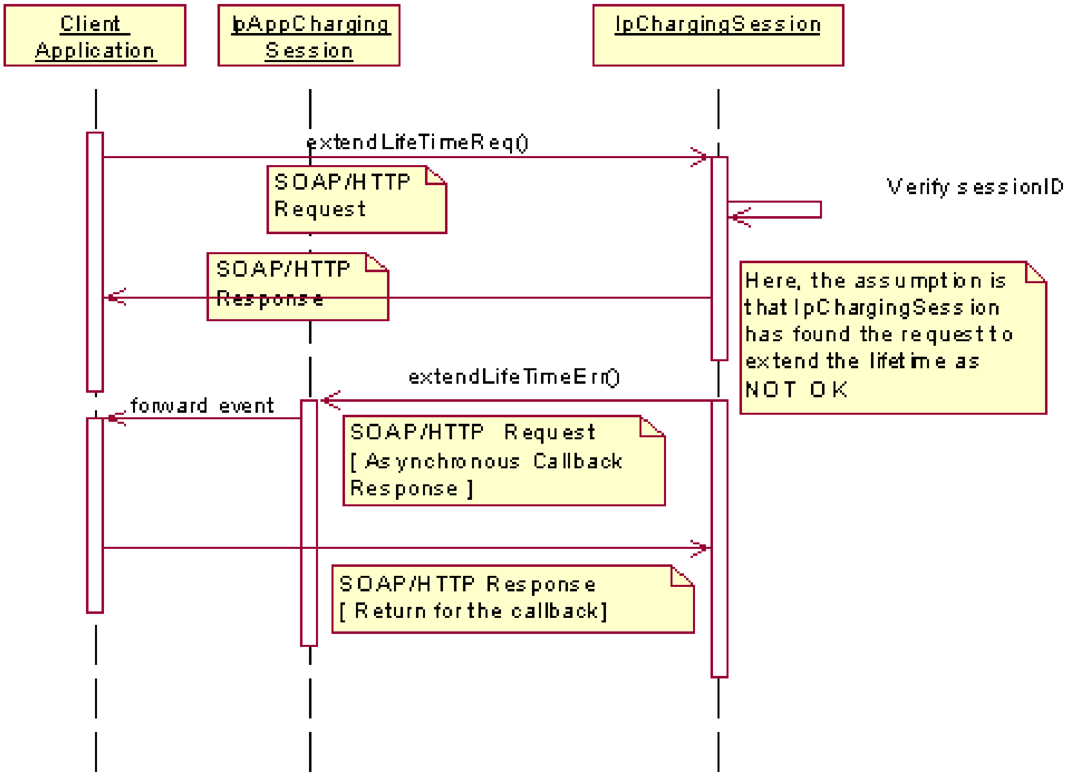
Returns:

void: no value

extendLifeTimeReq() Unsuccessful Operation

If the reservation cannot be extended, the request is denied by the IpChargingSession by issuing an extendLifeTimeErr() callback to the client application, as shown in Figure 29, "extendLifeTimeReq – Unsuccessful Operation."

Figure 29 extendLifeTimeReq – Unsuccessful Operation


Client Application
IpAppCharging Session
IpChargingSession
extendLifeTimeReq()
SOAP/HTTP Request
Verify sessionID
SOAP/HTTP Response
Here, the assumption is that IpChargingSession has found the request to extend the lifetime as NOT OK
extendLifeTimeErr()
forward event
SOAP/HTTP Request [Asynchronous Callback Response]
SOAP/HTTP Response [Return for the callback]

Error Callback Method: extendLifeTimeErr()

This is an error callback to the asynchronous call extendLifeTimeReq(). This method is called to indicate that the corresponding request was unsuccessful.

Signature:

extendLifeTimeErr (sessionID : in TpSessionID, error : in TpChargingError) : void

Parameters:

- sessionID : in TpSessionID

  The ID of the current session that the reservation applies to.
- error : in TpChargingError

  Indicates the reason for the error. Possible errors are: P_CHS_ERR_NO_EXTEND

Returns:

void: no value

Remaining Session Lifetime Query Method: getLifeTimeLeft()

Returns the lifetime left for a valid reservation. This method works for both the Amount Reserved and Volume Reserved states.

Signature:

getLifeTimeLeft (sessionID : in TpSessionID) : TpInt32

Parameters:

- sessionID : in TpSessionID

  The ID of the session.

Returns:

reservationTimeLeft: Indicates the number of seconds that the session remains valid. TpInt32

Raises:

- **TpCommonExceptions**: Common exceptions that typically represent platform errors. Please refer to the description in the TpCommonExceptions .
- **P_INVALID_SESSION_ID**: The sessionID parameter contains a value that is not for a currently known session.

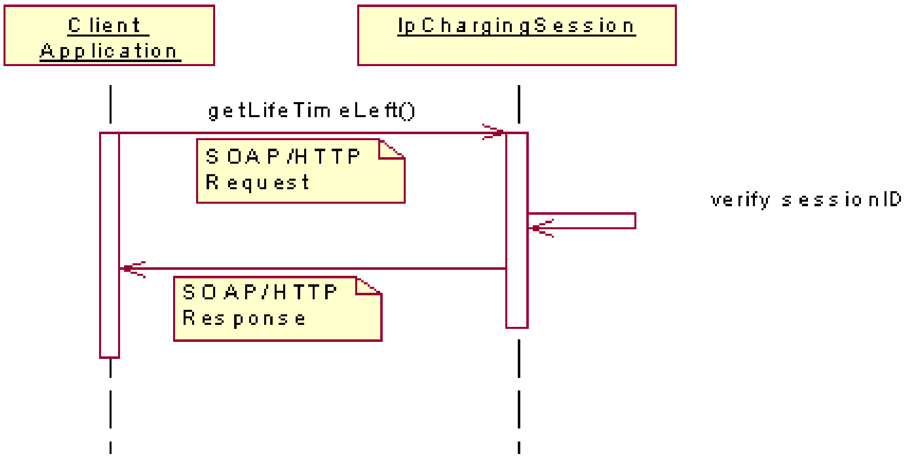
getLifeTimeLeft() Successful Operation

This is accomplished by a synchronous method call: getLifeTimeLeft()

If there are any active reservations on this session, the lifetime of each of those reservations is also the same as the session lifetime.

With this method an application can request the remaining lifetime of the reservation. If no reservation has been made on the charging session, this method raises a TpCommonException (P_TASK_REFUSED). Figure 30, “Query the Remaining Lifetime of a Reservation” shows the successful operation of the getLifeTimeLeft() method.

Figure 30 Query the Remaining Lifetime of a Reservation



Amount-based Reservation Creation Method: reserveAmountReq()

This method is used when an application wants to reserve an amount of money for services to be delivered to a user. It is also possible to enlarge the existing amount reservation by invoking this method. If a reservation is extended, the lifetime of the reservation is re-initialized.

Signature:

reserveAmountReq (sessionID : in TpSessionID,
applicationDescription : in TpApplicationDescription,
chargingParameters : in TpChargingParameterSet,
preferredAmount : in TpChargingPrice, minimumAmount : in TpChargingPrice,
requestNumber : in TpInt32) : void

Parameters:

- sessionID : in TpSessionID

  The ID of the current session to which the reservation applies.
- applicationDescription : in TpApplicationDescription

  Descriptive text for informational purposes (for example, text presented on the bill and used in communication with the user). This value is committed to the CDR.
- chargingParameters : in TpChargingParameterSet

  These parameters and their values specify to the SCS what service was provided to the end user so that the charging service can determine the type of charge. The Comverse OSA SCS expects client applications need to populate this construct based on the Charging Mode specified when creating the session.
- preferredAmount : in TpChargingPrice

  The amount of specified currency that the application wants reserved.

NOTE For Comverse OSA SCS, this must be a non-negative amount.

- minimumAmount : in TpChargingPrice

  The minimum amount that can be used by the application if the preferred amount cannot be granted.

NOTE For Comverse OSA SCS, this must be a non-negative amount. Also, this amount must not be greater than the amount specified in the preferredAmount parameter.

- requestNumber : in TpInt32

  Specifies the number given in the result of the previous operation on this session or when creating the session. When no answer is received, the same operation with the same parameters must be retried with the same requestNumber.

Raises:

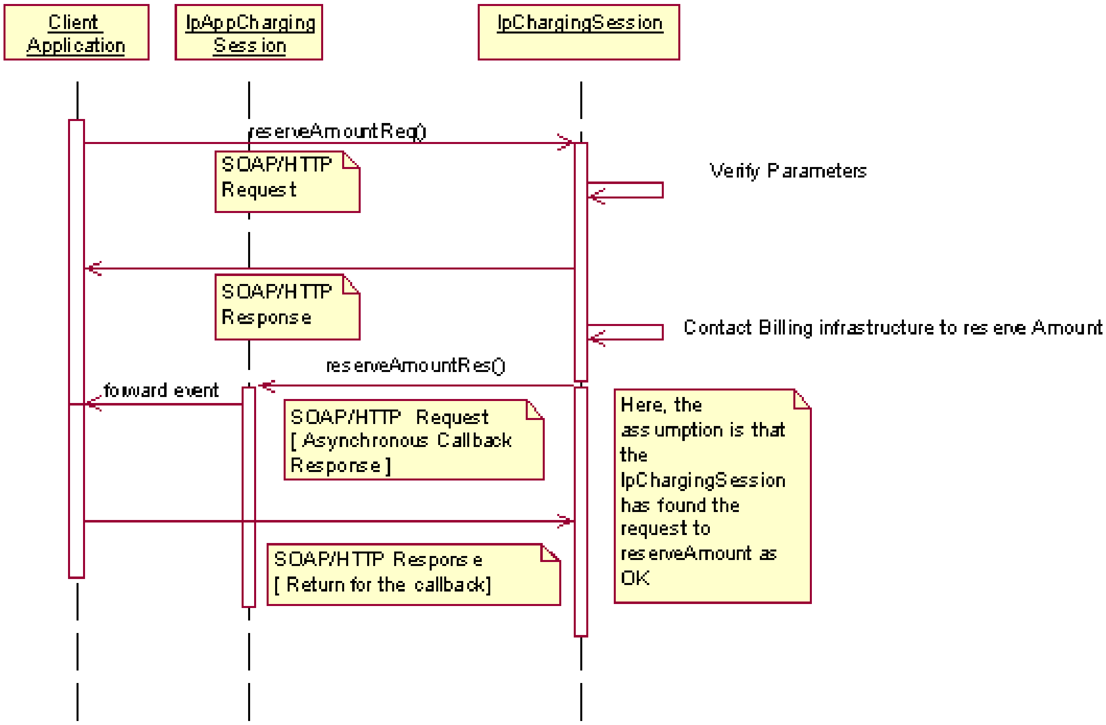
- **TpCommonExceptions**: Common exceptions that typically represent platform errors. Please refer to the description in the section “TpCommonExceptions,” on page 65.
- **P_INVALID_SESSION_ID**: The sessionID parameter contains a value that is not for a currently known session.
- **P_INVALID_AMOUNT:** The preferred or minimum amount in the reserveAmountReq is less than or equal to zero.
- **P_INVALID_CURRENCY:** The ISO 4217 currency specified is not provisioned into the OSA Charging service.

- **P_INVALID_REQUEST_NUMBER**: The requestNumber argument does not contain the expected value, the value that was returned from the previous callback response or from the createChargingSession() method.

reserveAmountReq() Successful Operation

Figure 31, “Create an Amount-based Reservation – Successful Operation” shows the successful operation of the reserveAmountReq() method.

Figure 31 Create an Amount-based Reservation – Successful Operation


Client Application
IpAppCharging Session
IpChargingSession
reserveAmountReq()
Verify Parameters
SOAP/HTTP Request
SOAP/HTTP Response
Contact Billing infrastructure to reserve Amount
reserveAmountRes()
forward event
SOAP/HTTP Request [Asynchronous Callback Response]
Here, the assumption is that the IpChargingSession has found the request to reserveAmount as OK
SOAP/HTTP Response [Return for the callback]

Callback Method: reserveAmountRes()

This callback method indicates that the corresponding request was successful.

Signature:

reserveAmountRes (sessionID : in TpSessionID, requestNumber : in TpInt32,
reservedAmount : in TpChargingPrice, sessionTimeLeft : in TpInt32,
requestNumberNextRequest : in TpInt32) : void

Parameters:

- sessionID : in TpSessionID

  Same as the session ID returned in the request.
- requestNumber : in TpInt32

  Request Number for this request.
- reservedAmount : in TpChargingPrice

  The amount reserved. If there was already a pending reservation, the sum of that and the new reservation is given.
- sessionTimeLeft : in TpInt32

Indicates the number of seconds that the session and the reservation in the session remain valid.

- requestNumberNextRequest : in TpInt32

  This Request Number must be used in the next request (requiring a Request Number) for this session.

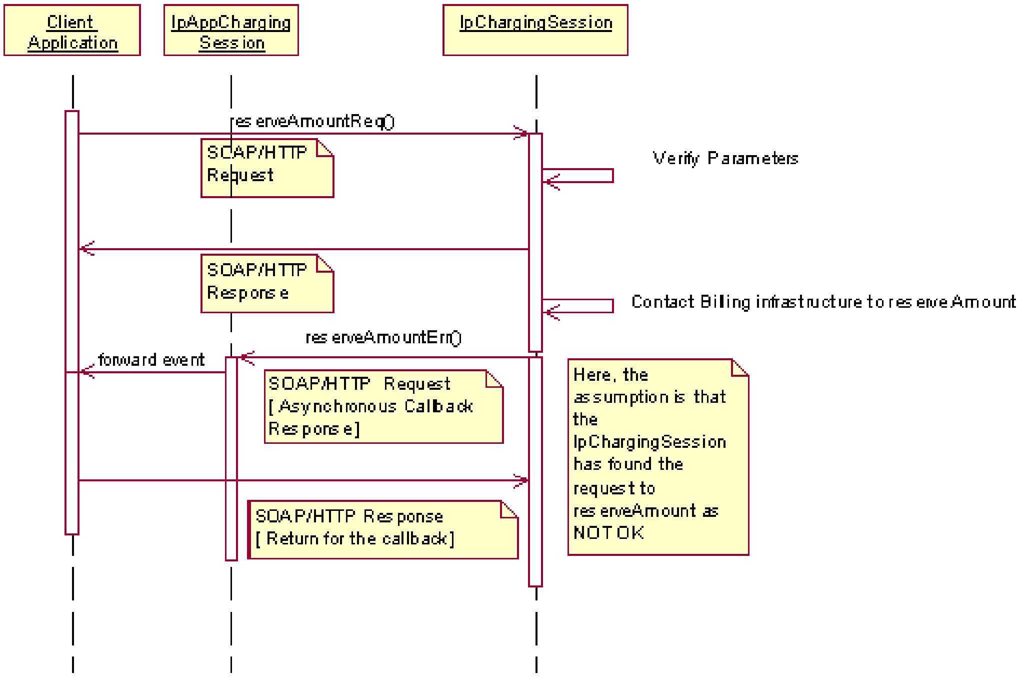
- TpChargingPrice data type :

  Defines the sequence of data elements that identifies a price.

reserveAmountReq() Unsuccessful Operation

Figure 32, “Create an Amount-based Reservation – Unsuccessful Operation” shows the unsuccessful operation of the reserveAmountReq() method.

Figure 32 Create an Amount-based Reservation – Unsuccessful Operation


Client Application
IpAppCharging Session
IpChargingSession
reserveAmountReq()
SOAP/HTTP Request
Verify Parameters
SOAP/HTTP Response
Contact Billing infrastructure to reserveAmount
reserveAmountErr()
forward event
SOAP/HTTP Request [Asynchronous Callback Response]
Here, the assumption is that the IpChargingSession has found the request to reserveAmount as NOT OK
SOAP/HTTP Response [Return for the callback]

Error Callback Method: reserveAmountErr()

This error callback method indicates that the corresponding request failed. The reservation cannot be used.

Signature:

reserveAmountErr (sessionID : in TpSessionID, requestNumber : in TpInt32,

error : in TpChargingError, requestNumberNextRequest : in TpInt32) : void

Parameters:

- sessionID : in TpSessionID

  Same as the session ID returned in the request.

- requestNumber: in TpInt32

  Request Number for this request.

- error: in TpChargingError

  Indicates the reason for failure. Possible errors are:

  - P_CHS_ERR_PARAMETER

  - P_CHS_ERR_RESERVATION_LIMIT
  - P_CHS_ERR_CURRENCY
  - P_CHS_ERR_NO_EXTEND
  - P_CHS_ERR_CONFIRMATION_REQUIRED
- requestNumberNextRequest : in TpInt32

  This Request Number must be used in the next request (requiring a Request Number) for this session.

NOTE If there was an unused or partially used valid reservation that was existing on this session, it remains valid until the end of its original lifetime, in spite of this error.

Remaining Amount Query Method: getAmountLeft()

With this method, an application requests the remaining amount of the reservation.

Returns amountLeft: Gives the amount left in the reservation.

Signature:

getAmountLeft (sessionID : in TpSessionID) : TpInt32

Parameters:

- sessionID : in TpSessionID

  The ID of the session.

Returns:

TpChargingPrice

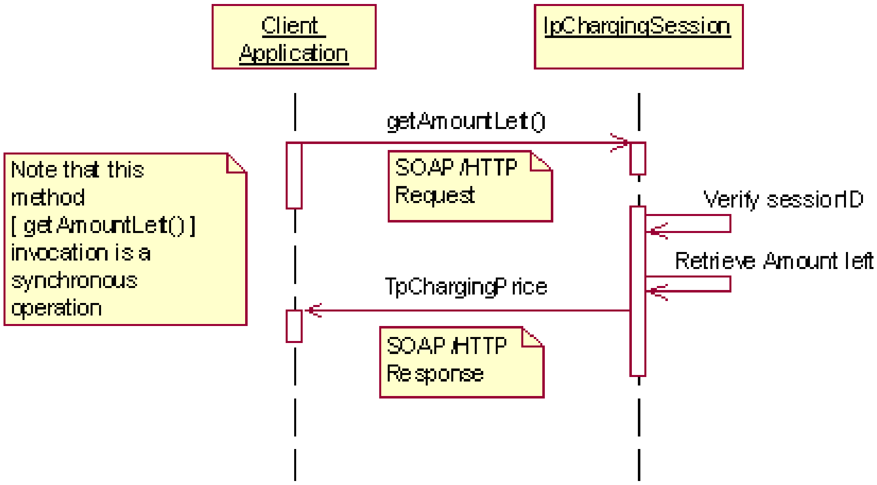
Raises:

- **TpCommonExceptions**: Common exceptions that typically represent platform errors. Please refer to the description in the section “TpCommonExceptions,” on page 65.
- **P_INVALID_SESSION_ID**: The sessionID parameter contains a value that is not for a currently known session.

getAmountLeft() Successful Operation

Figure 33, “Query For Remaining Amount – Successful Operation” shows the successful operation of the getAmountLeft() method.

Figure 33 Query For Remaining Amount – Successful Operation



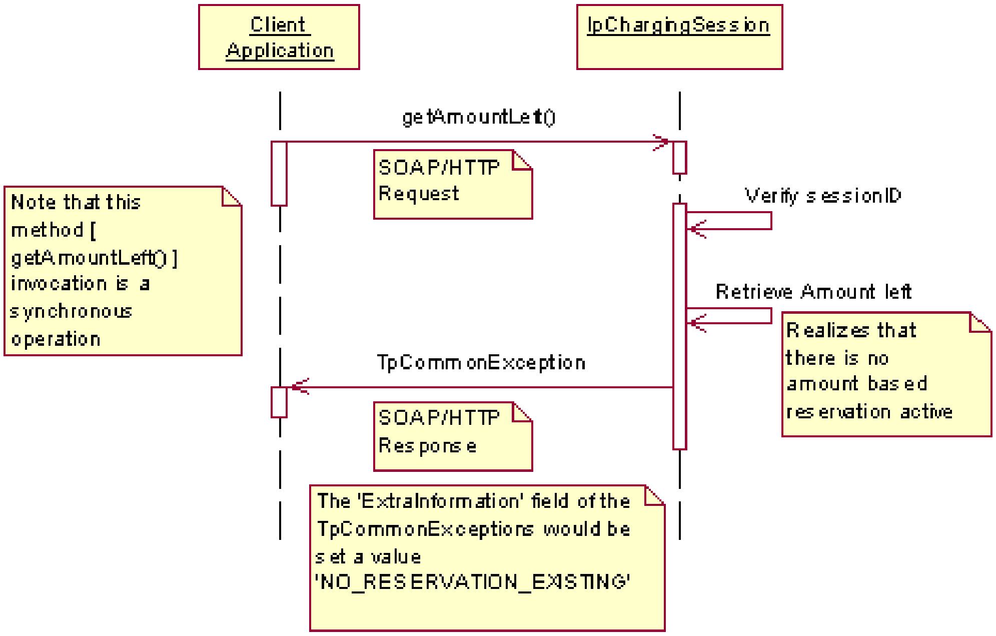
getAmountLeft() Unsuccessful Operation

The method getAmountLeft() can result in exceptions under different conditions.

Exception **P_INVALID_SESSION_ID** is returned if the sessionID passed as an argument to the request is deemed invalid. Another possible scenario is when there is no amount-based reservation existing, even though there is a valid session with the sessionID that is passed.

Figure 34, “Query Remaining Amount – Exception Condition, No Existing Reservation” shows the unsuccessful operation of the getAmountLeft() method.

Figure 34 Query Remaining Amount – Exception Condition, No Existing Reservation


Client Application
IpChargingSession
getAmountLeft()
SOAP/HTTP Request
Verify sessionID
Retrieve Amount left
Realizes that there is no amount based reservation active
TpCommonException
SOAP/HTTP Response
Note that this method [getAmountLeft()] invocation is a synchronous operation
The 'ExtraInformation' field of the TpCommonExceptions would be set a value 'NO_RESERVATION_EXISTING'

Reserved Amount Debit Method: debitAmountReq()

This method debits an amount from the reservation.

The amount left in the reservation is decreased by this amount.

Each request to debit or credit an amount toward a reservation is handled separately. For example, two requests for a payment of EUR 1 gives a total payment of EUR 2.

A credit of EUR 1 and a debit of EUR 1 gives a total payment of EUR 0.

When a debit operation exceeds the limit of the reservation, the debit operation fails.

Signature:

debitAmountReq (sessionID : in TpSessionID,
applicationDescription : in TpApplicationDescription, amount : in TpChargingPrice,
closeReservation : in TpBoolean, requestNumber : in TpInt32)

Parameters:

- sessionID : in TpSessionID

  The ID of the session
- applicationDescription : in TpApplicationDescription

  Descriptive text for informational purposes (for example, text presented on the bill and used for communication with the user).
- amount : in TpChargingPrice

  The amount of specified currency to be debited from the user.

NOTE For Comverse OSA SCS, this must be a non-negative amount.

- closeReservation : in TpBoolean

  If set to True, this parameter indicates that the reservation can be freed. The session is not released. This must be done explicitly by calling the release() method.
- requestNumber : in TpInt32

  Specifies the number given in the result of the previous operation on this session or when creating the session. When no answer is received, the same operation with the same parameters must be retried with the same requestNumber.

Raises:

- **TpCommonExceptions**: Common exceptions that typically represent platform errors. Please refer to the description in the section “TpCommonExceptions,” on page 65.
- **P_INVALID_SESSION_ID**: The sessionID parameter contains a value that is not for a currently known session.
- **P_INVALID_AMOUNT:** The argument amount is less than or equal to zero.
- **P_INVALID_CURRENCY:** The ISO 4217 currency specified is not provisioned into the OSA Charging service.
- **P_INVALID_REQUEST_NUMBER**: The requestNumber argument does not contain the expected value, the value that was returned from the previous callback response or from the createChargingSession() method.

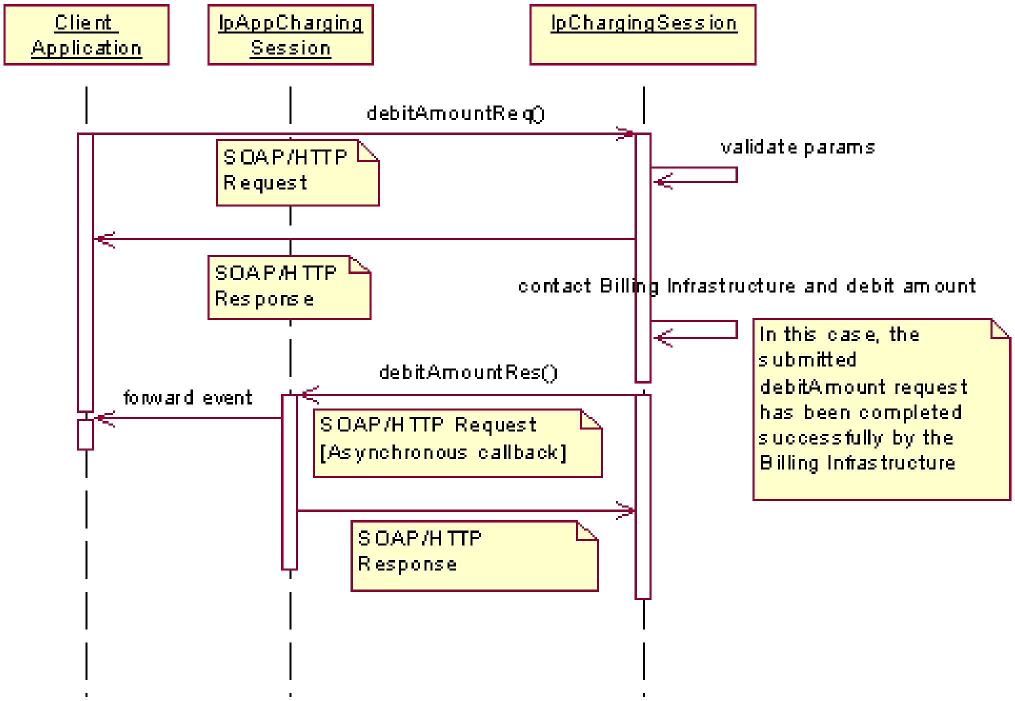
debitAmountReq() Successful Operation

The client application can choose to debit a valid amount from a reserved amount.

The request to debit is submitted to the IpChargingSession by invoking debitAmountReq() method.

An exception could result if the parameter validation fails. However, at this stage, this is only a preliminary verification. The method returns without waiting for debit confirmation. A debit confirmation or debit error is notified asynchronously to the IpAppChargingSession by the SCS using debitAmountRes() or debitAmountErr(). Figure 35, “Debit Amount – Successful Operation” shows the successful operation of the debitAmountReq() method.

Figure 35 Debit Amount – Successful Operation


Client Application
IpAppCharging Session
IpChargingSession
debitAmountReq()
validate params
SOAP/HTTP Request
SOAP/HTTP Response
contact Billing Infrastructure and debit amount
In this case, the submitted debitAmount request has been completed successfully by the Billing Infrastructure
debitAmountRes()
forward event
SOAP/HTTP Request [Asynchronous callback]
SOAP/HTTP Response

Callback Method: debitAmountRes()

This callback method indicates that the corresponding request was successful.

Signature:

debitAmountRes (sessionID : in TpSessionID, requestNumber : in TpInt32,

debitedAmount : in TpChargingPrice, reservedAmountLeft : in TpChargingPrice,

requestNumberNextRequest : in TpInt32) : void

Parameters:

- sessionID : in TpSessionID
  ID of the session for which the operation was called.
- requestNumber : in TpInt32
  Request Number for this request.
- debitedAmount : in TpChargingPrice
  Indicates the debited amount.

NOTE For Comverse OSA SCS, this must be a non-negative amount.

- reservedAmountLeft : in TpChargingPrice

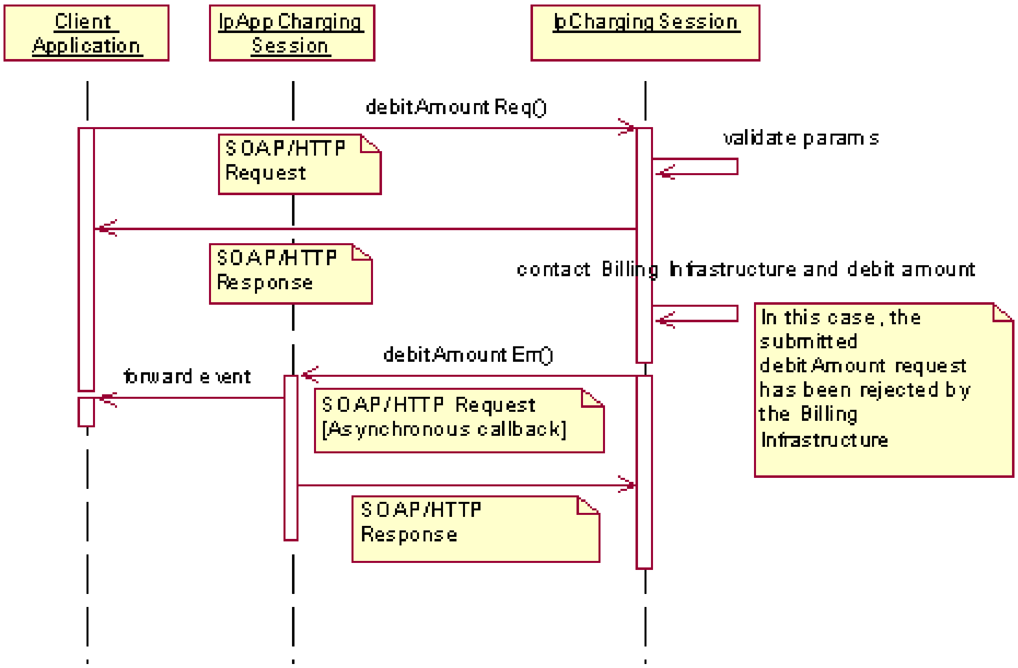
  The amount left of the reservation.
- requestNumberNextRequest : in TpInt32

  This Request Number must be used in the next request (requiring a Request Number) for this session.

debitAmountReq() Unsuccessful Operation

Failure is communicated by SCS to IpAppChargingSession by an asynchronous callback debitAmountErr(), as shown in Figure 36, "Debit Amount – Unsuccessful Operation."

Figure 36 Debit Amount – Unsuccessful Operation


Client Application
IpApp Charging Session
IpCharging Session
debitAmountReq()
validate params
SOAP/HTTP Request
SOAP/HTTP Response
contact Billing Infrastructure and debit amount
In this case, the submitted debitAmount request has been rejected by the Billing Infrastructure
debitAmountErr()
forward event
SOAP/HTTP Request [Asynchronous callback]
SOAP/HTTP Response

Error Callback Method: debitAmountErr()

This error callback method indicates that the corresponding request failed completely and no money has been debited.

Signature:

debitAmountErr (sessionID : in TpSessionID, requestNumber : in TpInt32,

error : in TpChargingError, requestNumberNextRequest : TpInt32) : void

Parameters:

- sessionID : in TpSessionID

  ID of the session for which the operation was called.

- requestNumber : in TpInt32

  Request Number for this request.
- error : in TpChargingError

  Indicates the reason for failure. Possible errors are: P_CHS_ERR_CURRENCY and P_CHS_ERR_RESERVATION_LIMIT.
- requestNumberNextRequest : in TpInt32

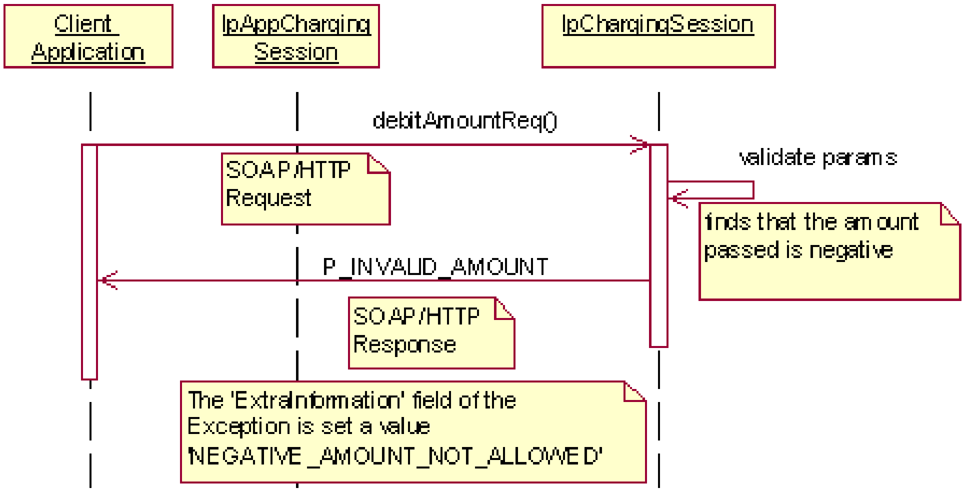
  This Request Number must be used in the next request (requiring a Request Number) for this session.

debitAmountReq() Unsuccessful Operation – Negative Amount Value

An attempt to debit a negative amount is rejected by Comverse OSA SCS.

This is communicated by an exception P_INVALID_AMOUNT, as shown in Figure 37, “Debit Amount – Exception, Negative Amount.”

Figure 37 Debit Amount – Exception, Negative Amount



Reservation Credit Amount Method: creditAmountReq()

This method credits an amount toward the reservation associated with the session.

The amount left in the reservation is increased by this amount.

Each request to debit or credit an amount toward a reservation is handled separately. For example, two requests for a payment of EUR 1, gives a total payment of EUR 2.

A credit of EUR 1 and a debit of EUR 1 gives a total payment of EUR 0.

Signature:

creditAmountReq (sessionID : in TpSessionID, applicationDescription : in TpApplicationDescription, amount : in TpChargingPrice, closeReservation : in TpBoolean, requestNumber : in TpInt32)

Parameters:

- sessionID : in TpSessionID

  The ID of the session.
- applicationDescription : in TpApplicationDescription

Descriptive text for informational purposes (for example, text presented on the bill and used for communication with the user).

- amount : in TpChargingPrice

  The amount of specified currency to be credited to the user.

Comverse OSA SCS mandates this to be a non-negative amount.

- closeReservation : in TpBoolean

  If set to True, this parameter indicates that the remaining part of the reservation can be freed. This can also mean addition of currency to the subscriber's account if more credits than debits have been made. The session is not released. This must be done explicitly by calling the release() method.

- requestNumber : in TpInt32

  Specifies the number given in the result of the previous operation on this session or when creating the session. When no answer is received, the same operation with the same parameters must be retried with the same requestNumber.

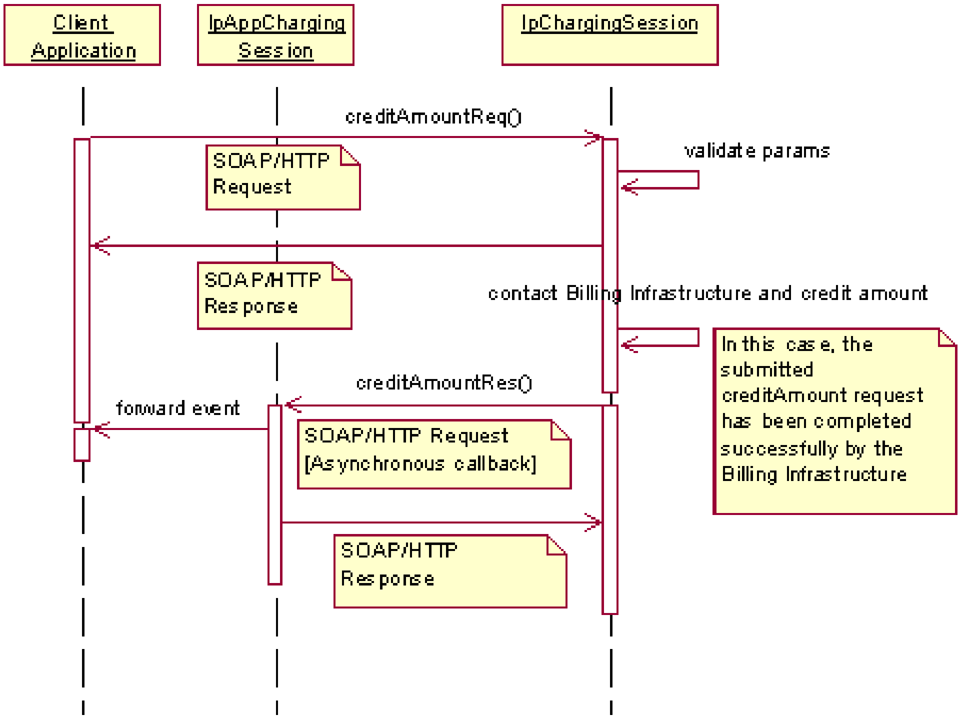
Raises:

- **TpCommonExceptions**: Common exceptions that typically represent platform errors. Please refer to the description in the section “TpCommonExceptions,” on page 65.
- **P_INVALID_SESSION_ID**: The sessionID parameter contains a value that is not for a currently known session.
- **P_INVALID_AMOUNT:** The argument amount is less than or equal to zero.
- **P_INVALID_CURRENCY:** The ISO 4217 currency specified is not provisioned into the OSA Charging service.
- **P_INVALID_REQUEST_NUMBER**: The requestNumber argument does not contain the expected value, the value that was returned from the previous callback response or from the createChargingSession() method.

creditAmountReq() Successful Operation

Figure 38, “Credit Amount – Successful Operation”shows the successful operation of the creditAmountReq() method.

Figure 38 Credit Amount – Successful Operation


Client Application
IpAppCharging Session
IpChargingSession
creditAmountReq()
validate params
SOAP/HTTP Request
SOAP/HTTP Response
contact Billing Infrastructure and credit amount
In this case, the submitted creditAmount request has been completed successfully by the Billing Infrastructure
creditAmountRes()
forward event
SOAP/HTTP Request [Asynchronous callback]
SOAP/HTTP Response

Callback Method: creditAmountRes()

This callback method indicates that the corresponding request was successful.

Signature:

creditAmountRes (sessionID : in TpSessionID, requestNumber : in TpInt32, creditedAmount : in TpChargingPrice, reservedAmountLeft : in TpChargingPrice, requestNumberNextRequest : in TpInt32) : void

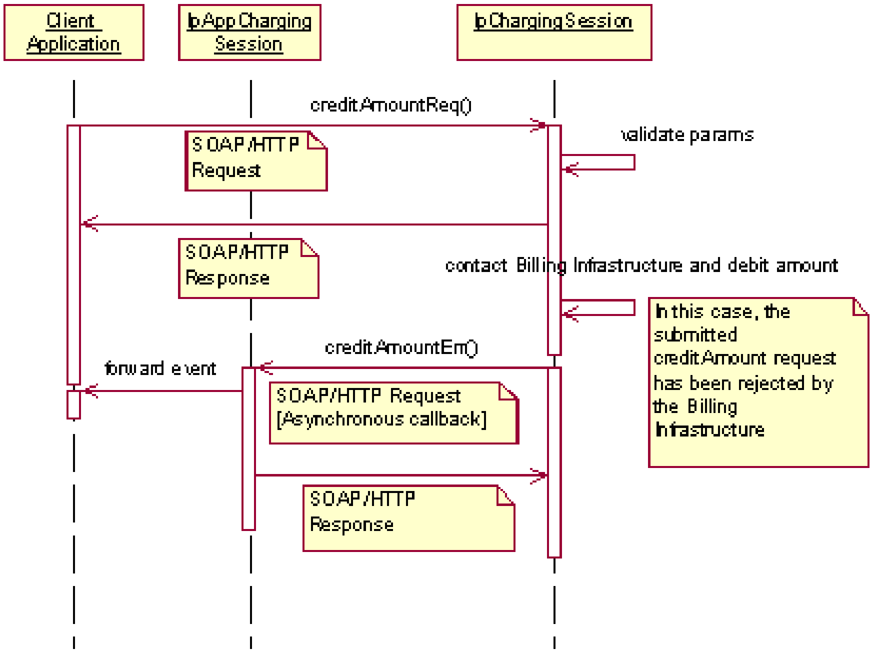
Parameters:

- sessionID : in TpSessionID
  ID of the session for which the operation was called.
- requestNumber : in TpInt32
  Request Number for this request.
- creditedAmount : in TpChargingPrice
  Indicates the credited amount.
- reservedAmountLeft : in TpChargingPrice
  The amount left of the reservation.
- requestNumberNextRequest : in TpInt32
  This Request Number must be used in the next request (requiring a Request Number) for this session.

creditAmountReq() Unsuccessful Operation

Failure is communicated by SCS to IpAppChargingSession by an asynchronous callback creditAmountErr(), as shown in Figure 39, “Credit Amount – Unsuccessful Operation.”

Figure 39 Credit Amount – Unsuccessful Operation


Client Application
IpAppCharging Session
IpChargingSession
creditAmountReq()
validate params
SOAP/HTTP Request
SOAP/HTTP Response
contact Billing Infrastructure and debit amount
In this case, the submitted creditAmount request has been rejected by the Billing Infrastructure
creditAmountErr()
forward event
SOAP/HTTP Request [Asynchronous callback]
SOAP/HTTP Response

Error Callback Method: creditAmountErr()

This error callback method indicates that the corresponding request failed completely and that no money has been credited.

Signature:

creditAmountErr (sessionID : in TpSessionID, requestNumber : in TpInt32,

error : in TpChargingError, requestNumberNextRequest : in TpInt32) : void

Parameters:

- sessionID : in TpSessionID

  ID of the session for which the operation was called.
- requestNumber : in TpInt32

  Request Number for this request.
- error : in TpChargingError

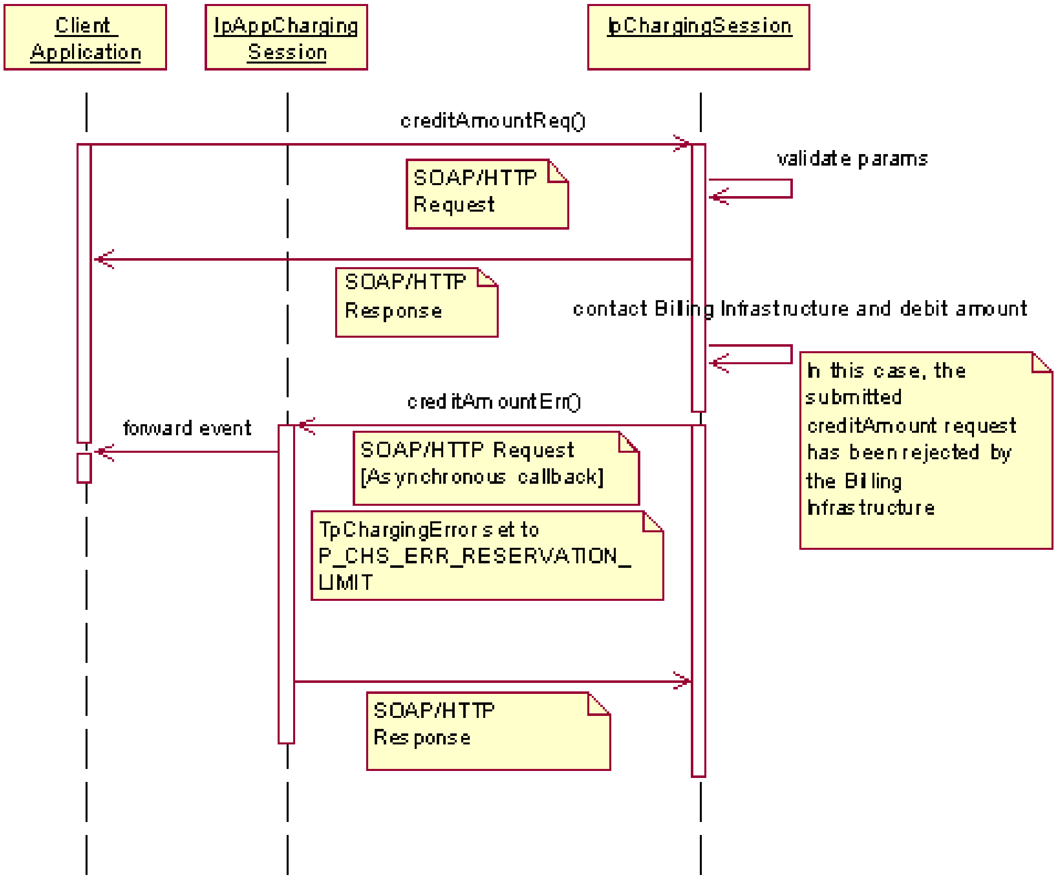
  Indicates the reason for failure. Possible errors are: P_CHS_ERR_CURRENCY and P_CHS_ERR_NO_CREDIT.
- requestNumberNextRequest : in TpInt32

  This Request Number must be used in the next request (requiring a Request Number) for this session.

creditAmountReq() Unsuccessful Operation – Requested Amount Violates Reservation Limits

Comverse OSA SCS ensures that the credit and debit activities honor the reservation limits, when they are conducted within the context of a reservation. This implies that any credit request that does not satisfy the condition [Reserved amount >= (Remaining Balance from Reservation + Requested Credit Amount)] returns an error response from OSA SCS. Figure 40, “Credit Amount – Error, Requested Amount Violates Reservation Limits” shows this scenario.

Figure 40 Credit Amount – Error, Requested Amount Violates Reservation Limits


Client Application
IpAppCharging Session
IpChargingSession
creditAmountReq()
validate params
SOAP/HTTP Request
SOAP/HTTP Response
contact Billing Infrastructure and debit amount
In this case, the submitted creditAmount request has been rejected by the Billing Infrastructure
creditAmountErr()
forward event
SOAP/HTTP Request [Asynchronous callback]
TpChargingError set to P_CHS_ERR_RESERVATION_LIMIT
SOAP/HTTP Response

An OSA client application now wishes to recharge/topup this account using OSA Charging SCS. This is outside of the capabilities of a reservation-based creditAmountReq() operation. With a 0 amount as the current balance, any attempt to create a non-zero amount reservation fails. Recharge/topup activities must invoke a directCreditAmountReq() operation.

Unit-Based Reservation Method: reserveUnitReq()

This method is used when an application wants to reserve volumes of application usage to deliver to a user in the session. When using units, it is assumed that the price setting for the units is handled by the network side services. It is also possible to enlarge the existing unit reservation by invoking this method.

Signature:

reserveUnitReq (sessionID : in TpSessionID, applicationDescription : in TpApplicationDescription, chargingParameters : in TpChargingParameterSet, volumes : in TpVolumeSet, requestNumber : in TpInt32)

Parameters:

- sessionID : in TpSessionID

  The ID of the session.

- applicationDescription : in TpApplicationDescription

  Descriptive text for informational purposes (for example, text presented on the bill and used for communication with the user).

- chargingParameters : in TpChargingParameterSet

  These parameters and their values specify to the charging service what was provided to the end user so that the charging service can determine the applicable tariff.

NOTE For Comverse OSA SCS, Client applications need to populate this construct based on the Charging Mode.

- volumes : in TpVolumeSet

  Specifies the reserved volumes as a sequence of data elements each containing the amount and applied unit. It is possible to make a reservation for 10,000 octets and 5 charging units.
- requestNumber : in TpInt32

  Specifies the number given in the result of the previous operation on this session or when creating the session. When no answer is received, the same operation with the same parameters must be retried with the same requestNumber.

Raises:

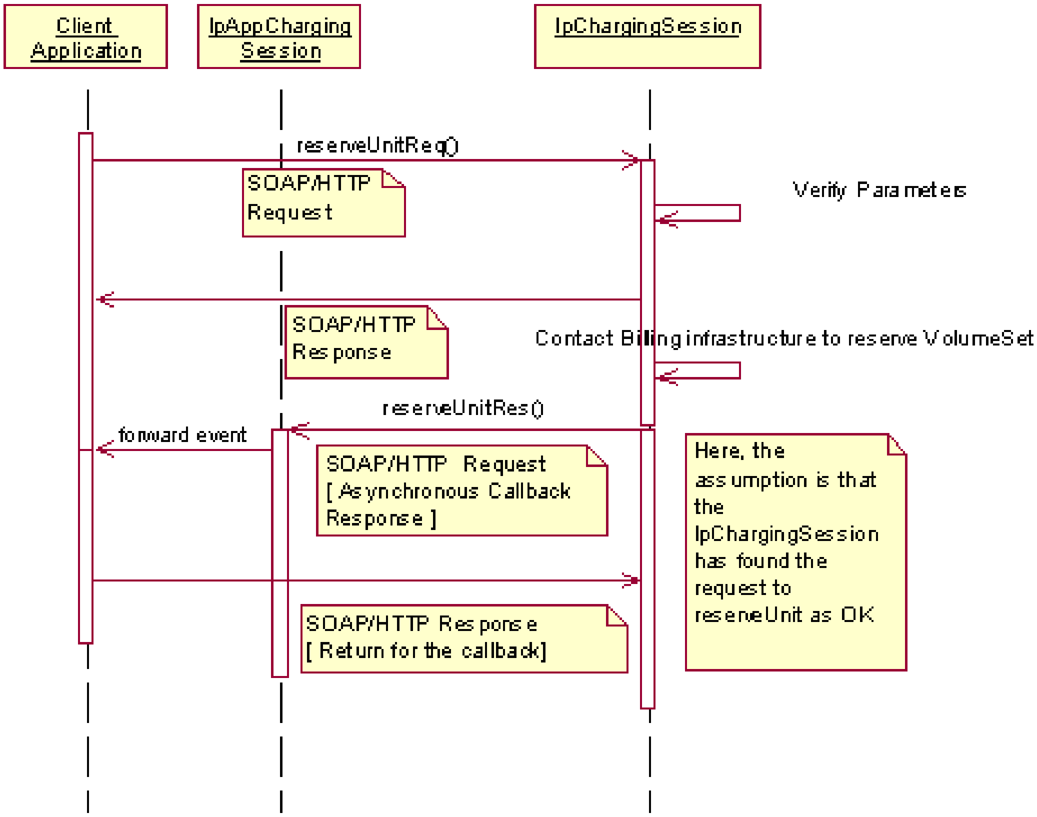
- **TpCommonExceptions**: Common exceptions that typically represent platform errors. Please refer to the description in the section “TpCommonExceptions,” on page 65.
- **P_INVALID_SESSION_ID**: The sessionID parameter contains a value that is not for a currently known session.
- **P_INVALID_VOLUME:** The requested volume was less than or equal to zero.
- **P_INVALID_REQUEST_NUMBER**: The requestNumber argument does not contain the expected value, the value that was returned from the previous callback response or from the createChargingSession() method.

reserveUnitReq() Successful Operation

Unit-based reservations are created by invoking the reserveUnitReq() method on IpChargingSession. This can cause an exception if the preliminary parameter validations result in errors. Beyond that, the response (success or error) is communicated asynchronously by SCS to IpAppChargingSession by invoking the callback methods [reserveUnitRes()/reserveUnitErr()].

Unlike the amount-based reservation signature, the unit-based reservation request in OSA charging SCF specifications does not have a distinct way of communicating a Preferred VolumeSet and a Minimum VolumeSet. This introduces certain important issues, most notably, what must be done when certain volume types do not have sufficient balances as requested. One option is to reject the reservation request altogether. A second, more lenient approach is to grant the reservation with a lower volume. Comverse OSA Charging SCS follows the latter approach. To facilitate this, a default minimum for each unit type must be provisioned in Comverse ONE solution. In general, the volume of units granted by the SCS could be different from the requested volume. Developers are urged to insure that client applications examine the reservedUnits field in the response to verify the exact volume information that is associated with a successful reservation. Figure 41, “Reserve Unit – Successful Operation” shows the successful operation of the reserveUnitReq() method.

Figure 41 Reserve Unit – Successful Operation


Client Application
IpAppCharging Session
IpChargingSession
reserveUnitReq()
SOAP/HTTP Request
Verify Parameters
SOAP/HTTP Response
Contact Billing infrastructure to reserve VolumeSet
reserveUnitRes()
forward event
SOAP/HTTP Request [Asynchronous Callback Response]
Here, the assumption is that the IpChargingSession has found the request to reserveUnit as OK
SOAP/HTTP Response [Return for the callback]

Callback Method: reserveUnitRes()

This method indicates that the corresponding request was successful.

Signature:

ReserveUnitReq (sessionID : in TpSessionID, requestNumber : in TpInt32,

reservedUnits : in TpVolumeSet, sessionTimeLeft : in TpInt32,

requestNumberNextRequest : in TpInt32) : void

Parameters:

- sessionID : in TpSessionID

  Same as the session ID returned in the request.
- requestNumber : in TpInt32

  Request Number for this request.
- reservedUnits : in TpVolumeSet

  The volume of application usage reserved. If there was already a pending reservation, the sum of that and the new reservation is returned. A pending reservation of 25 charging units and a new reservation of 1,000 octets and 10 charging units results in two TpVolume elements for this parameter: 1,000 octets and 35 charging units.
- sessionTimeLeft : in TpInt32

  Indicates the number of seconds that the session and the reservation in the session remain valid.
- requestNumberNextRequest : in TpInt32

  This Request Number must be used in the next request (requiring a Request Number) for this session.

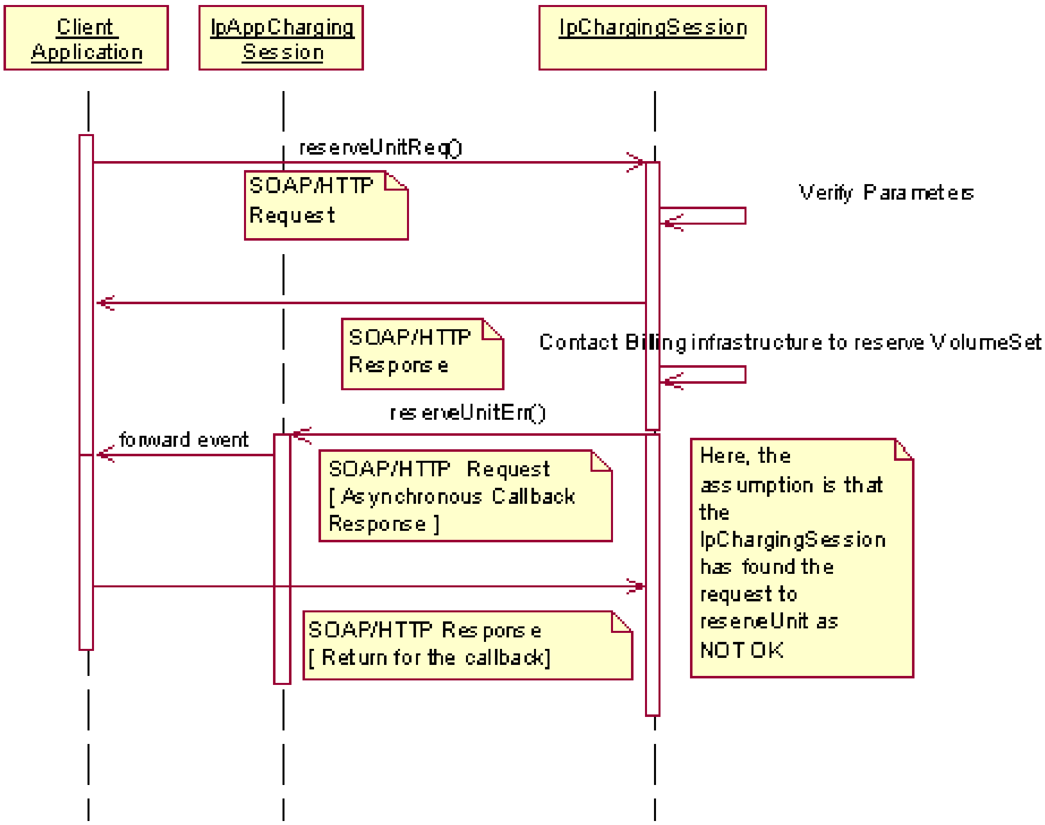
reserveUnitReq() Unsuccessful Operation

A client request to reserve a volume set (a set of units) can return an error response if OSA SCS is unable to honor the request. For Comverse OSA Charging SCS, some of the typical error scenarios are as follows:

- TpChargingParameterSet argument does not map to any charging capability associations provisioned in Comverse ONE solution.
- TpVolumeSet elements are not understood by Comverse OSA SCS.
- The customer account does not have enough balance to grant the reservation request.
- The charging session has lived long enough and no more lifetime extensions are possible. Creating a reservation automatically extends the lifetime of the session as well.
- A unit-based reservation exists on the session already and the TpVolumeSet passed in the new reservation request indicates a unit type that is different from the unit type in effect for the already existing reservation. Such a combination is not supported in Phase 1 of Comverse OSA SCS.
- A unit-based reservation exists on the session already and the TpChargingParameterSet passed in the new reservation request is different from the TpChargingParameterSet that is in effect for the already existing reservation. Such a combination is not supported in Phase 1 of Comverse OSA SCS.

Figure 42, “Reserve Unit – Error Response” shows the unsuccessful operation of the reserveUnitReq() method.

Figure 42 Reserve Unit – Error Response


Client Application
IpAppCharging Session
IpChargingSession
reserveUnitReq()
SOAP/HTTP Request
Verify Parameters
SOAP/HTTP Response
Contact Billing infrastructure to reserve VolumeSet
reserveUnitErr()
forward event
SOAP/HTTP Request [Asynchronous Callback Response]
SOAP/HTTP Response [Return for the callback]
Here, the assumption is that the IpChargingSession has found the request to reserveUnit as NOT OK

Error Callback Method: reserveUnitErr()

This error callback method indicates that the corresponding request failed. The reservation cannot be used.

Signature:

reserveUnitErr (sessionID : in TpSessionID, requestNumber : in TpInt32,
error : in TpChargingError, requestNumberNextRequest : in TpInt32) : void

Parameters:

- sessionID : in TpSessionID
  Same as the session ID returned in the request.
- requestNumber : in TpInt32
  Request Number for this request.
- error : in TpChargingError
  Indicates the reason for failure. Possible errors are:
  - P_CHS_ERR_PARAMETER
  - P_CHS_ERR_VOLUMES
  - P_CHS_ERR_RESERVATION_LIMIT
  - P_CHS_ERR_NO_EXTEND
  - P_CHS_ERR_CONFIRMATION_REQUIRED
- requestNumberNextRequest : in TpInt32
  This Request Number must be used in the next request (requiring a Request Number) for this session.

Query the Remaining Volumes in a Reservation Method: getUnitLeft()

With this method, an application requests the remaining amount of the reservation.

Returns volumesLeft: Specifies the remaining volumes as a sequence of data elements, each containing the amount and applied unit.

Signature:

getUnitLeft (sessionID : in TpSessionID) : TpVolumeSet

Parameters:

- sessionID : in TpSessionID
  The ID of the session.

Returns:

TpVolumeSet

Raises:

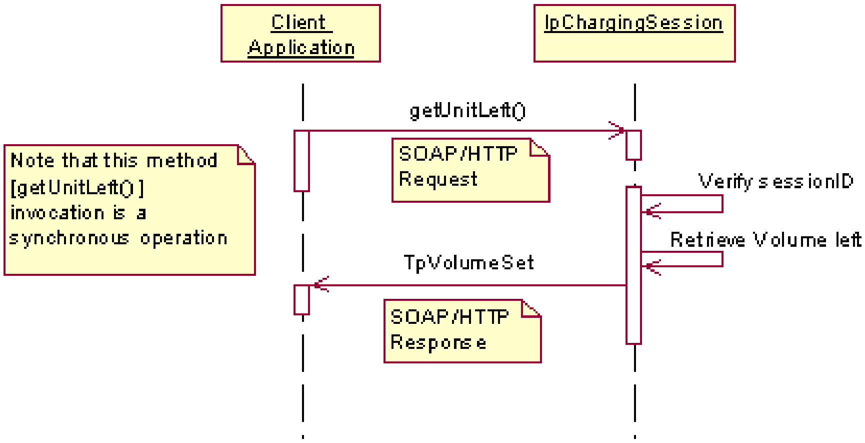
- **TpCommonExceptions**: Common exceptions that typically represent platform errors. Please refer to the description in the section “TpCommonExceptions,” on page 65.
- **P_INVALID_SESSION_ID**: The sessionID parameter contains a value that is not for a currently known session.

getUnitLeft() Successful Operation

The client application can query the SCS to obtain the remaining volumes associated with a session, by invoking getUnitLeft() method on the IpChargingSession. Figure 43, “Query to Get the Remaining Volume” shows this scenario.

This method is relevant only when there is an existing unit-based reservation.

Figure 43 Query to Get the Remaining Volume



Unit-based Reservation Debit Method: debitUnitReq()

This method debits a volume of application usage from the reservation.

The volumes left in the reservation are decreased by this amount.

Each request to debit or credit a volume toward a reservation is handled separately. For example, two requests for a payment for 10 kilobytes gives a total payment of 20 kilobytes.

Signature:

debitUnitReq (sessionID : in TpSessionID, applicationDescription : in TpApplicationDescription, volumes : in TpVolumeSet, closeReservation : in TpBoolean, requestNumber : in TpInt32) : void

Parameters:

- sessionID : in TpSessionID

  The ID of the session.
- applicationDescription : in TpApplicationDescription

  Descriptive text for informational purposes (for example, text presented on the bill and used for communication with the user)
- volumes : in TpVolumeSet

  Specifies the charged volumes as a sequence of data elements, each containing the amount and applied unit.

NOTE Each VolumeSet element must have a matching element in the VolumeSet specified at the reservation request.

- closeReservation : in TpBoolean

  If set to True, this parameter indicates that the reservation can be freed. The session is not released. This must be done explicitly by calling the release() method.

- requestNumber : in TpInt32

  Specifies the number given in the result of the previous operation on this session or when creating the session. When no answer is received, the same operation with the same parameters must be retried with the same requestNumber.

Raises:

- **TpCommonExceptions**: Common exceptions that typically represent platform errors. Please refer to the description in the section "TpCommonExceptions," on page 65.
- **P_INVALID_SESSION_ID**: The sessionID parameter contains a value that is not for a currently known session.
- **P_INVALID_VOLUME:** The requested volume was less than or equal to zero.
- **P_INVALID_REQUEST_NUMBER**: The requestNumber argument does not contain the expected value, the value that was returned from the previous callback response or from the createChargingSession() method.

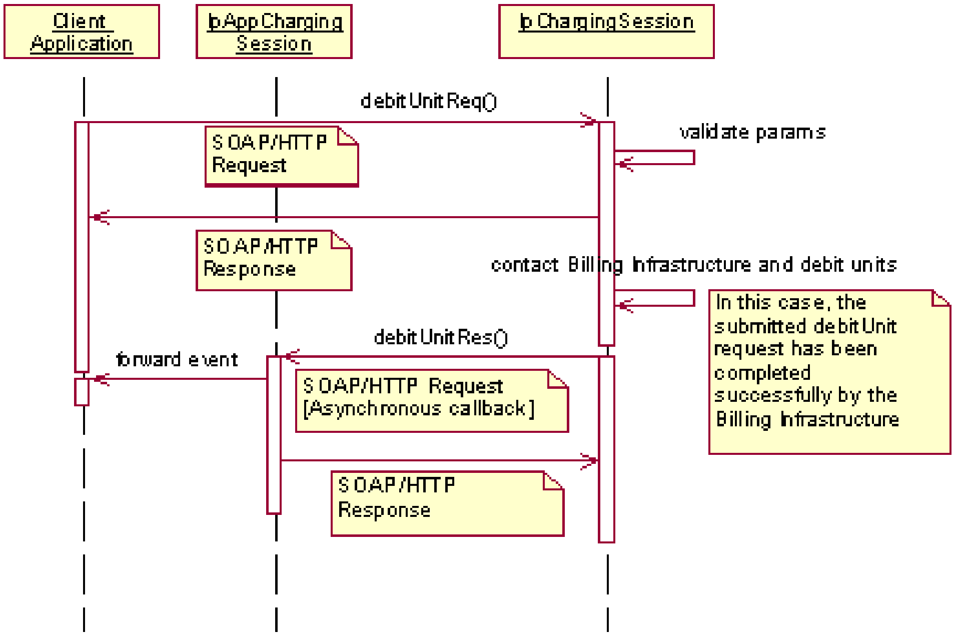
debitUnitReq() Successful Operation

The client application can choose to debit a valid VolumeSet from a reserved volume.

The request to debit is submitted to the IpChargingSession by invoking debitUnitReq() method. Figure 44, "Debit Units [Volume]" shows this scenario.

An exception could result if the parameter validation fails. However, at this stage, this is only preliminary verification. The method returns without waiting for debit confirmation. A debit confirmation or debit error is notified asynchronously to the IpAppChargingSession by the SCS using debitUnitRes() or debitUnitErr().

Figure 44 Debit Units [Volume]


Client Application
IpAppCharging Session
IpChargingSession
debitUnitReq()
validate params
SOAP/HTTP Request
SOAP/HTTP Response
contact Billing Infrastructure and debit units
In this case, the submitted debitUnit request has been completed successfully by the Billing Infrastructure
debitUnitRes()
forward event
SOAP/HTTP Request [Asynchronous callback]
SOAP/HTTP Response

Callback Method: debitUnitRes()

This callback method indicates that the corresponding request was successful.

Signature:

debitUnitRes (sessionID : in TpSessionID, requestNumber : in TpInt32, debitedVolumes : in TpVolumeSet, reservedUnitsLeft : in TpVolumeSet,

requestNumberNextRequest : in TpInt32) : void

Parameters:

- sessionID : in TpSessionID

  ID of the session for which the operation was called.
- requestNumber : in TpInt32

  Request Number for this request.
- debitedVolumes : in TpVolumeSet

  Indicates the debited volumes of application usage.
- reservedUnitsLeft : in TpVolumeSet

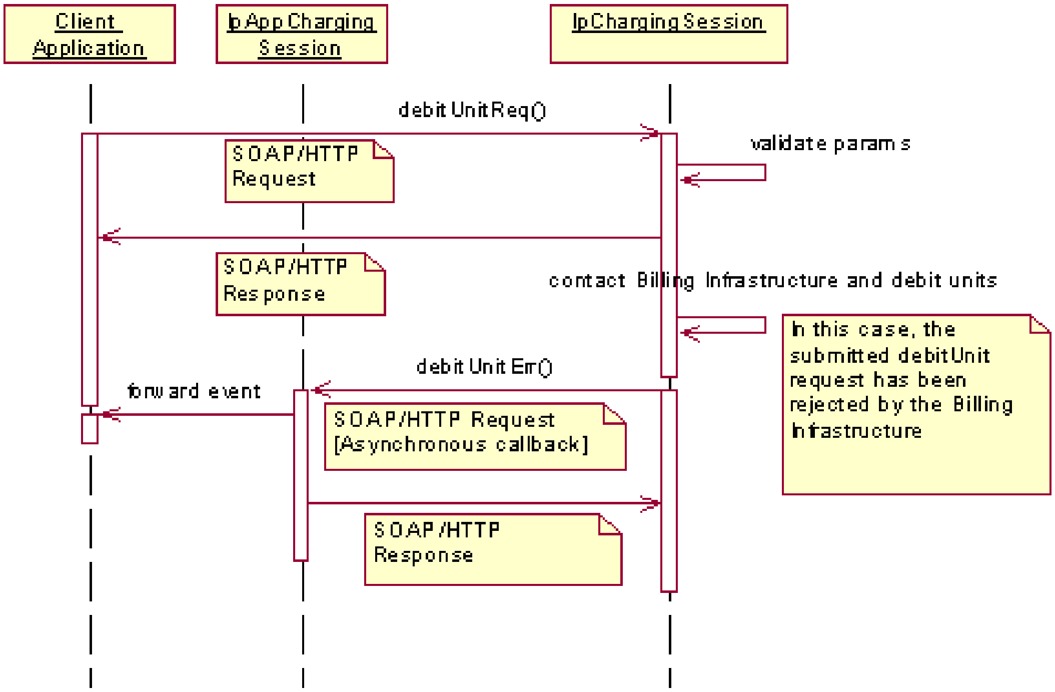
  The volume of application usage left in the reservation.
- requestNumberNextRequest : in TpInt32

  This Request Number must be used in the next request (requiring a Request Number) for this session.

debitUnitReq() Unsuccessful Operation

Failure is communicated by SCS to IpAppChargingSession by asynchronous callback debitUnitErr(), as shown in Figure 45, "Debit Units [Volume] – Error Scenario."

Figure 45 Debit Units [Volume] – Error Scenario


Client Application
IpAppCharging Session
IpChargingSession
debitUnitReq()
SOAP/HTTP Request
validate params
SOAP/HTTP Response
contact Billing Infrastructure and debit units
In this case, the submitted debitUnit request has been rejected by the Billing Infrastructure
debitUnitErr()
forward event
SOAP/HTTP Request [Asynchronous callback]
SOAP/HTTP Response

Error Callback Method: debitUnitErr()

This error callback method indicates that the corresponding request failed completely and that no units have been debited.

Signature:

debitUnitErr (sessionID : in TpSessionID, requestNumber : in TpInt32,

error : in TpChargingError, requestNumberNextRequest : in TpInt32) : void

Parameters:

- sessionID : in TpSessionID

  ID of the session for which the operation was called.
- requestNumber : in TpInt32

  Request Number for this request.
- error : in TpChargingError

  Indicates the reason for failure. Possible errors are: P_CHS_ERR_VOLUMES and P_CHS_ERR_RESERVATION_LIMIT.
- requestNumberNextRequest : in TpInt32

  This Request Number must be used in the next request (requiring a Request Number) for this session.

Unit-based Reservation Credit Method: creditUnitReq()

This method credits a volume of application usage toward the reservation.

The volume left in the reservation is increased by this amount.

Each request to debit or credit a volume toward a reservation is handled separately. For example, two requests for a payment of 10 kilobytes gives a total payment of 20 kilobytes.

Signature:

creditUnitReq (sessionID : in TpSessionID,

applicationDescription : in TpApplicationDescription, volumes : in TpVolumeSet, closeReservation : in TpBoolean, requestNumber : in TpInt32) : void

Parameters:

- sessionID : in TpSessionID

  The ID of the session.
- applicationDescription : in TpApplicationDescription

  Descriptive text for informational purposes (for example, text presented on the bill and used for communication with the user).
- volumes : in TpVolumeSet

  Specifies the credited volumes as a sequence of data elements, each containing the amount and applied unit.
- closeReservation : in TpBoolean

  If set to True, this parameter indicates that the reservation can be freed. The session is not released. This must be done explicitly by calling the release() method.
- requestNumber : in TpInt32

  Specifies the number given in the result of the previous operation on this session or when creating the session. When no answer is received, the same operation with the same parameters must be retried with the same requestNumber.

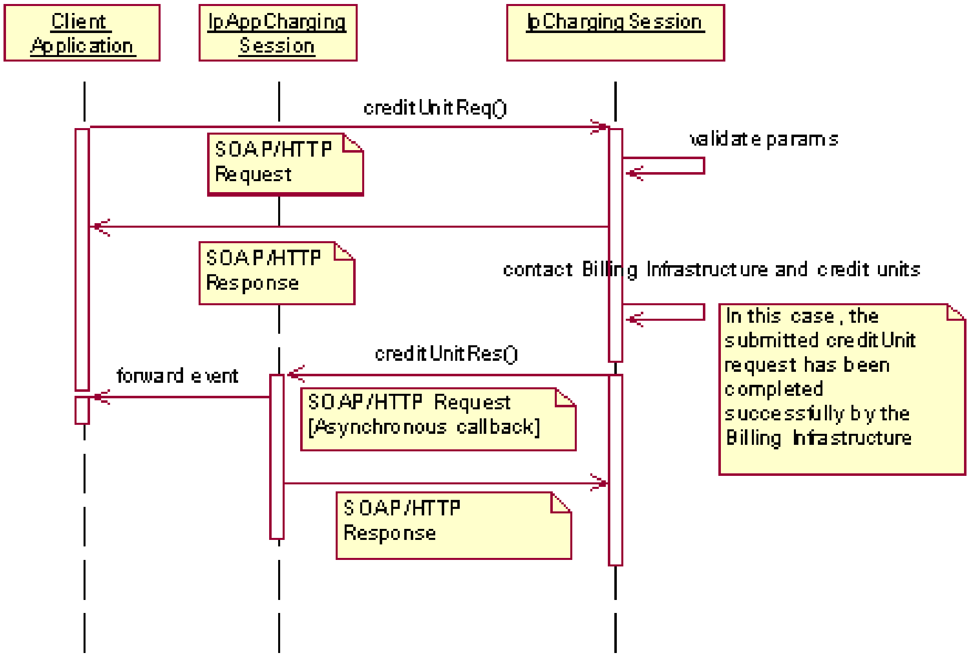
Raises:

- **TpCommonExceptions**: Common exceptions that typically represent platform errors. Please refer to the description in the section "TpCommonExceptions," on page 65.
- **P_INVALID_SESSION_ID**: The sessionID parameter contains a value that is not for a currently known session.
- **P_INVALID_VOLUME:** The requested volume was less than or equal to zero.
- **P_INVALID_REQUEST_NUMBER**: The requestNumber argument does not contain the expected value, the value that was returned from the previous callback response or from the createChargingSession() method.

creditUnitReq() Successful Operation

Figure 46, “Credit Unit” shows the successful operation of the creditUnitReq() method.

Figure 46 Credit Unit


Client Application
IpAppCharging Session
IpCharging Session
creditUnitReq()
validate params
SOAP/HTTP Request
SOAP/HTTP Response
contact Billing Infrastructure and credit units
In this case, the submitted creditUnit request has been completed successfully by the Billing Infrastructure
creditUnitRes()
forward event
SOAP/HTTP Request [Asynchronous callback]
SOAP/HTTP Response

Callback Method: creditUnitRes()

This callback method indicates that the corresponding request was successful.

Signature:

creditUnitRes (sessionID : in TpSessionID, requestNumber : in TpInt32, creditedVolumes : in TpVolumeSet, reservedUnitsLeft : in TpVolumeSet, requestNumberNextRequest : in TpInt32) : void

Parameters:

- sessionID : in TpSessionID

  ID of the session for which the operation was called.
- requestNumber : in TpInt32

  Request Number for this request.
- creditedVolumes : in TpVolumeSet

  Indicates the credited volumes of application usage.
- reservedUnitsLeft : in TpVolumeSet

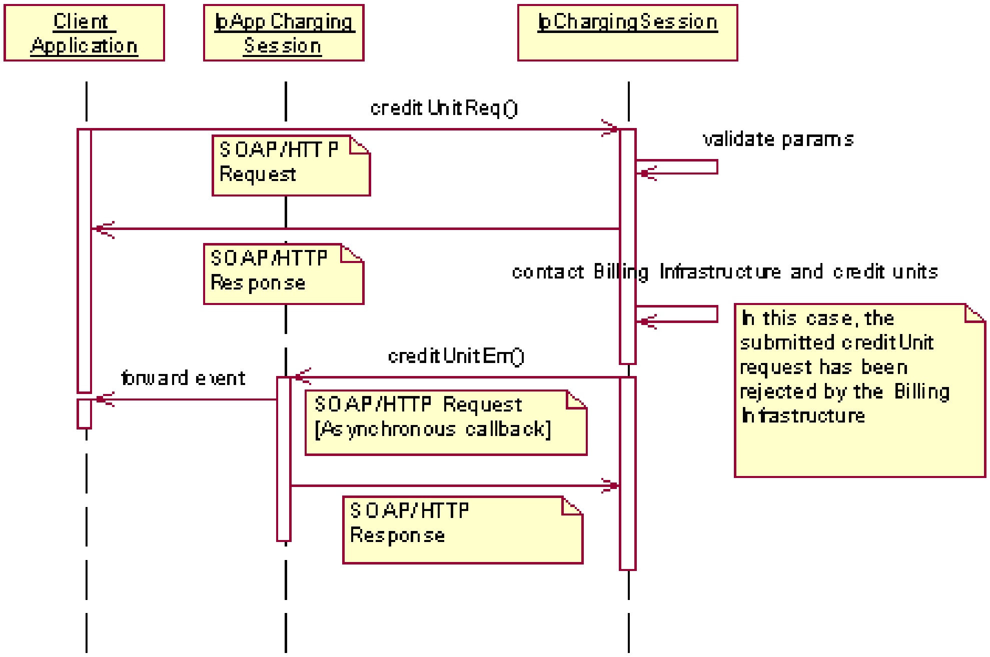
  The volume of application usage left in the reservation.
- requestNumberNextRequest : in TpInt32

  This Request Number must be used in the next request (requiring a Request Number) for this session.

creditUnitReq() Unsuccessful Operation

Figure 47, “Credit Unit – Error Scenario” shows the unsuccessful operation of the creditUnitReq() method.

Figure 47 Credit Unit – Error Scenario


Client Application
IpApp Charging Session
IpChargingSession
creditUnitReq()
validate params
SOAP/HTTP Request
SOAP/HTTP Response
contact Billing Infrastructure and credit units
In this case, the submitted creditUnit request has been rejected by the Billing Infrastructure
creditUnitErr()
forward event
SOAP/HTTP Request [Asynchronous callback]
SOAP/HTTP Response

Error Callback Method: creditUnitErr()

This error calback method indicates that the corresponding request failed completely and that no units have been credited.

Signature:

creditUnitErr (sessionID : in TpSessionID, requestNumber : in TpInt32,

error : in TpChargingError, requestNumberNextRequest : in TpInt32) : void

Parameters:

- sessionID : in TpSessionID

  ID of the session for which the operation was called.

- requestNumber : in TpInt32

  Request Number for this request.

- error : in TpChargingError

  Indicates the reason for failure. Possible errors are:

  - P_CHS_ERR_VOLUMES
  - P_CHS_ERR_NO_CREDIT

- requestNumberNextRequest : in TpInt32

  This Request Number must be used in the next request (requiring a Request Number) for this session.

Immediate Charging Debit Method: directDebitUnitReq()

This method debits a volume of application usage with no prior reservation made.

The volume left in the available balance is decreased by this amount.

Each request to debit a volume for immediate charging is handled separately. For example, two requests for a payment of 10 kilobytes gives a total payment of 20 kilobytes.

Signature:

directDebitUnitReq (sessionID : in TpSessionID, applicationDescription : in TpApplicationDescription, chargingParameters : in TpChargingParametersSet, volumes : in TpVolumeSet, requestNumber : in TpInt32) : void

Parameters:

- sessionID : in TpSessionID

  The ID of the session.

- applicationDescription : in TpApplicationDescription

  Descriptive text for informational purposes (for example, text presented on the bill and used for communication with the user).

- chargingParameters : in TpChargingParametersSet

  These parameters and their values specify to the charging service what was provided to the end user so that the charging service can determine the applicable tariff.

- volumes : in TpVolumeSet

  Specifies the charged volumes as a sequence of data elements, each containing the amount and applied unit.

NOTE Each VolumeSet element must have a matching element in the VolumeSet specified at the reservation request.

- requestNumber : in TpInt32

  Specifies the number given in the result of the previous operation on this session or when creating the session. When no answer is received, the same operation with the same parameters must be retried with the same requestNumber.

Raises:

- **TpCommonExceptions**: Common exceptions that typically represent platform errors. Please refer to the description in the section “TpCommonExceptions,” on page 65.
- **P_INVALID_SESSION_ID**: The sessionID parameter contains a value that is not for a currently known session.
- **P_INVALID_VOLUME:** The requested volume was less than or equal to zero.
- **P_INVALID_REQUEST_NUMBER**: The requestNumber argument does not contain the expected value, the value that was returned from the previous callback response or from the createChargingSession() method.

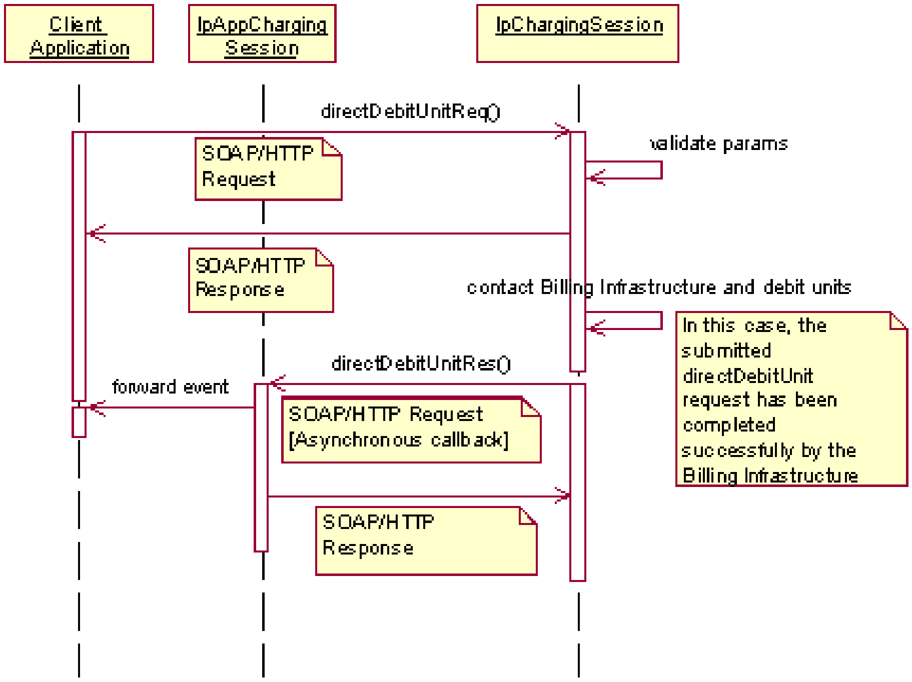
directDebitUnitReq() Successful Operation

The client application can choose to debit a valid VolumeSet with no prior reservations made.

The request to debit is submitted to the IpChargingSession by invoking the directDebitUnitReq() method after a createChargingSession() is invoked. Figure 48, “Direct Debit Units (Volume)” shows this scenario

An exception can result if the parameter validation fails. However, at this stage, this is only preliminary verification. The method returns without waiting for debit confirmation. A debit confirmation or debit error is notified asynchronously to the IpAppChargingSession by the SCS using directDebitUnitRes() or directDebitUnitErr().

Figure 48 Direct Debit Units (Volume)


Client Application
IpAppCharging Session
IpChargingSession
directDebitUnitReq()
validate params
SOAP/HTTP Request
SOAP/HTTP Response
contact Billing Infrastructure and debit units
In this case, the submitted directDebitUnit request has been completed successfully by the Billing Infrastructure
directDebitUnitRes()
forward event
SOAP/HTTP Request [Asynchronous callback]
SOAP/HTTP Response

Callback Method: directDebitUnitRes()

This callback method indicates that the corresponding request was successful.

Signature:

directDebitUnitRes (sessionID : in TpSessionID, requestNumber : in TpInt32, debitedVolumes : in TpVolumeSet, requestNumberNextRequest : in TpInt32) : void

Parameters:

- sessionID : in TpSessionID

  ID of the session for which the operation was called.
- requestNumber : in TpInt32

  Request Number for this request.
- debitedVolumes : in TpVolumeSet

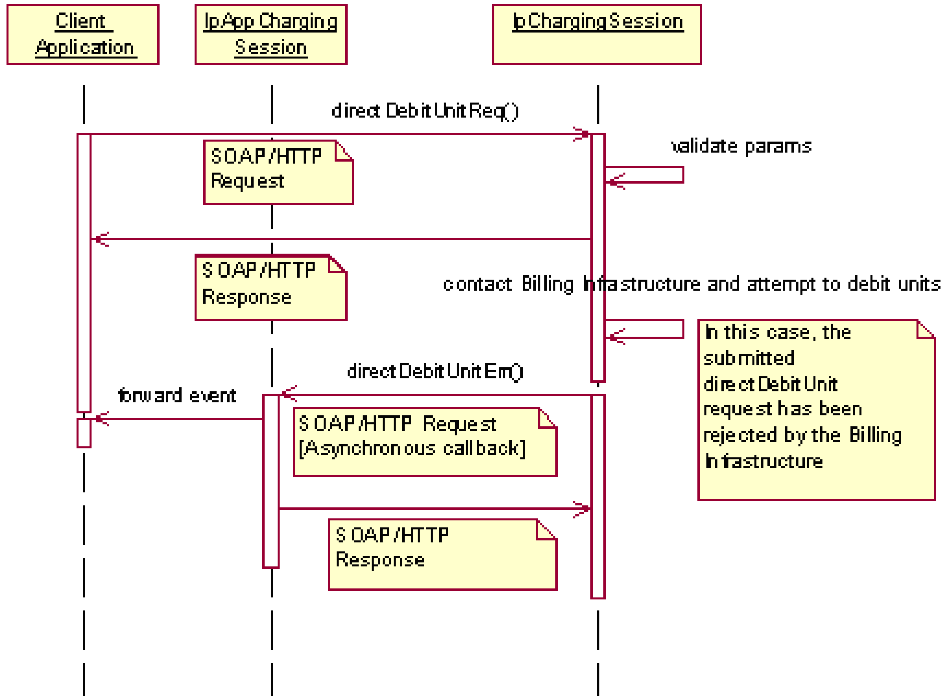
  Indicates the debited volumes of application usage.
- requestNumberNextRequest : in TpInt32

  This Request Number must be used in the next request (requiring a Request Number) for this session.

directDebitUnitReq() Unsuccessful Operation

Failure is communicated by SCS to IpAppChargingSession by asynchronous callback directDebitUnitErr(), as shown in Figure 49, “Direct Debit Units [Volume] – Error Scenario.”

Figure 49 Direct Debit Units [Volume] – Error Scenario


Client Application
IpApp Charging Session
IpChargingSession
directDebitUnitReq()
SOAP/HTTP Request
validate params
SOAP/HTTP Response
contact Billing Infrastructure and attempt to debit units
In this case, the submitted directDebitUnit request has been rejected by the Billing Infrastructure
directDebitUnitErr()
forward event
SOAP/HTTP Request [Asynchronous callback]
SOAP/HTTP Response

Error Callback Method: directDebitUnitErr()

This error callback method indicates that the corresponding request failed completely and that no units have been debited.

Signature:

directDebitUnitErr (sessionID : in TpSessionID, requestNumber : in TpInt32,

error : in TpChargingError, requestNumberNextRequest : in TpInt32) : void

Parameters:

- sessionID : in TpSessionID

  ID of the session for which the operation was called.
- requestNumber : in TpInt32

  Request Number for this request.
- error : in TpChargingError

  Indicates the reason for failure. Possible errors are:
  - P_CHS_ERR_VOLUMES
  - P_CHS_ERR_RESERVATION_LIMIT
- requestNumberNextRequest : in TpInt32

  This Request Number must be used in the next request (requiring a Request Number) for this session.

Immediate Charging Debit Method: directDebitAmountReq()

This method debits an amount of application usage with no prior reservation made from core balance.

The available balance is decreased by this amount.

Each request to debit a volume for immediate charging is handled separately. For example, two requests for a payment for $10 gives a total payment of $20.

When a debit operation exceeds the limit of the available balance, the debit operation fails and the available balance remains unmodified.

Signature:

directDebitAmountReq (sessionID : in TpSessionID, applicationDescription : in TpApplicationDescription, chargingParameters : in TpChargingParametersSet, amount : in TpChargingPrice, requestNumber : in TpInt32) : void

Parameters:

- sessionID : in TpSessionID

  The ID of the session.
- applicationDescription : in TpApplicationDescription

  Descriptive text for informational purposes (for example, text presented on the bill and used for communication with the user).
- chargingParameters : in TpChargingParametersSet

  These parameters and their values specify to the charging service what was provided to the end user so that the charging service can determine the applicable tariff.
- amount : in TpChargingPrice

  Specifies the amount and currency to be charged to the subscriber.
- requestNumber : in TpInt32

  Specifies the number given in the result of the previous operation on this session or when creating the session. When no answer is received, the same operation with the same parameters must be retried with the same requestNumber.

Raises:

- **TpCommonExceptions**: Common exceptions that typically represent platform errors. Please refer to the description in the section “TpCommonExceptions,” on page 65.
- **P_INVALID_SESSION_ID**: The sessionID parameter contains a value that is not for a currently known session.
- **P_INVALID_AMOUNT:** The argument amount is less than or equal to zero.
- **P_INVALID_REQUEST_NUMBER**: The requestNumber argument does not contain the expected value, the value that was returned from the previous callback response or from the createChargingSession() method.

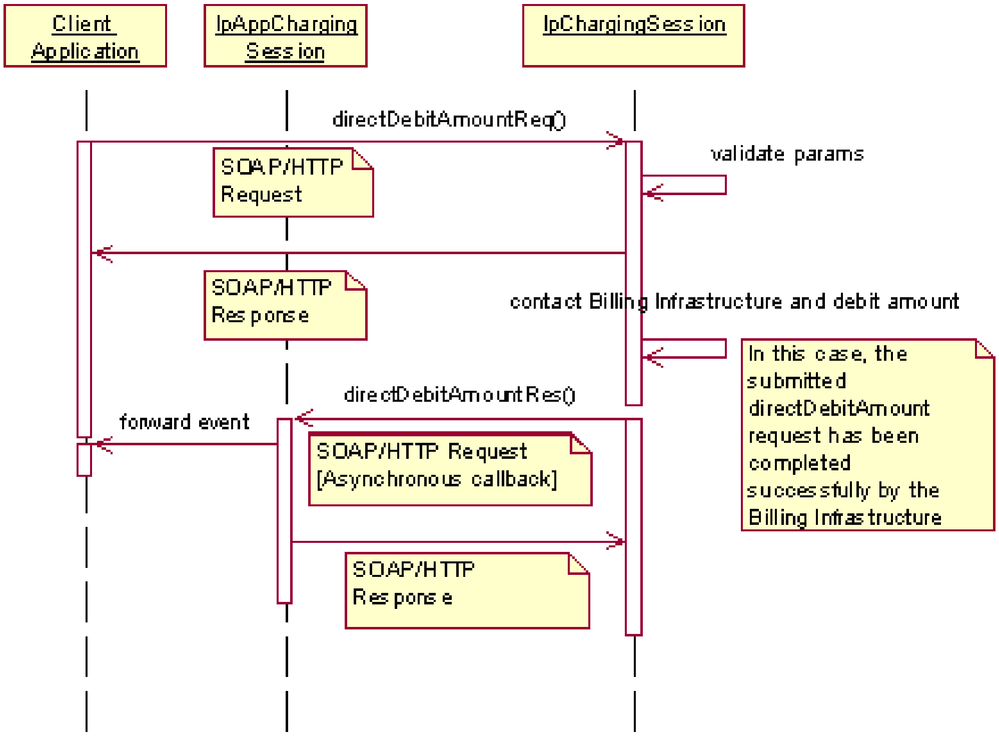
directDebitAmountReq() Successful Operation

The client application can choose to debit a currency amount with no prior reservations made.

The request to debit is submitted to the IpChargingSession by invoking the directDebitAmountReq() method after a createChargingSession() is invoked, as shown in Figure 50, “Direct Debit of Currency.”

An exception could result if the parameter validation fails. However, at this stage, this is only preliminary verification. The method returns without waiting for debit confirmation. The SCS notifies IpAppChargingSession of a debit confirmation or debit error asynchronously using directDebitAmountRes() or directDebitAmountErr().

Figure 50 Direct Debit of Currency


Client Application
IpAppCharging Session
IpChargingSession
directDebitAmountReq()
validate params
SOAP/HTTP Request
SOAP/HTTP Response
contact Billing Infrastructure and debit amount
In this case, the submitted directDebitAmount request has been completed successfully by the Billing Infrastructure
directDebitAmountRes()
forward event
SOAP/HTTP Request [Asynchronous callback]
SOAP/HTTP Response

Callback Method: directDebitAmountRes()

This callback method indicates that the corresponding request was successful.

Signature:

directDebitAmountRes (sessionID : in TpSessionID, requestNumber : in TpInt32, debitedAmount : in TpChargingPrice, requestNumberNextRequest : in TpInt32) : void

Parameters:

- sessionID : in TpSessionID

  ID of the session for which the operation was called.
- requestNumber : in TpInt32

  Request Number for this request.
- debitedAmount : in TpChargingPrice

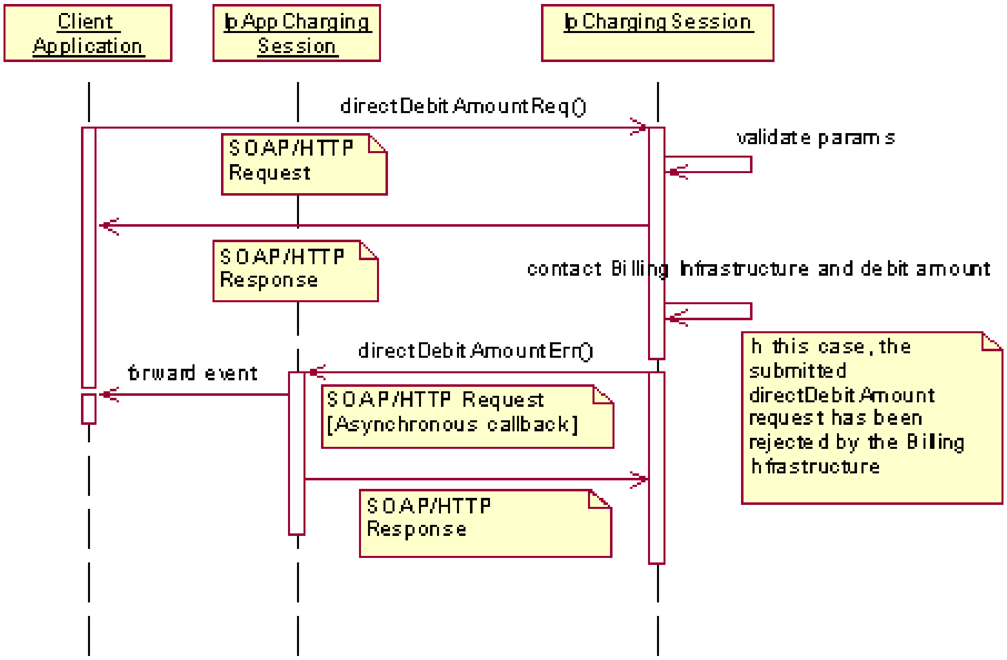
  Indicates the debited amount of application usage.
- requestNumberNextRequest : in TpInt32

  This Request Number must be used in the next request (requiring a Request Number) for this session.

directDebitAmountReq() Unsuccessful Operation

Failure is communicated by SCS to IpAppChargingSession by asynchronous callback directDebitAmountErr(), as shown in Figure 51, "Direct Debit Amount – Error Scenario."

Figure 51 Direct Debit Amount – Error Scenario


Client Application
IpApp Charging Session
Ip Charging Session
directDebitAmountReq()
SOAP/HTTP Request
validate params
SOAP/HTTP Response
contact Billing Infrastructure and debit amount
directDebitAmountErr()
forward event
SOAP/HTTP Request [Asynchronous callback]
In this case, the submitted directDebitAmount request has been rejected by the Billing Infrastructure
SOAP/HTTP Response

Error Callback Method: directDebitAmountErr()

This error callback method indicates that the corresponding request failed completely and that no units have been debited.

Signature:

directDebitAmountErr (sessionID : in TpSessionID, requestNumber : in TpInt32,

error : in TpChargingError, requestNumberNextRequest : in TpInt32) : void

Parameters:

- sessionID : in TpSessionID

  ID of the session for which the operation was called.
- requestNumber : in TpInt32

  Request Number for this request.
- error : in TpChargingError

  Indicates the reason for failure. Possible errors are: P_CHS_ERR_NO_DEBIT and P_CHS_ERR_ RESERVATION_LIMIT.
- requestNumberNextRequest : in TpInt32

  This Request Number must be used in the next request (requiring a Request Number) for this session.

Subscriber Balance Credit Method: directCreditAmountReq()

This method credits an amount of currency with no prior reservation made.

The available balance is increased by this amount.

Each request to credit an amount for immediate update is handled separately. For example, two requests for a credit of $10 results in a credit of $20.

When a credit operation exceeds the limit of the maximum balance, the credit operation fails and the available balance is unaltered.

Signature:

directCreditAmountReq (sessionID : in TpSessionID, applicationDescription : in TpApplicationDescription, chargingParameters : in TpChargingParametersSet, amount : in TpChargingPrice, requestNumber : in TpInt32) : void

Parameters:

- sessionID : in TpSessionID

  The ID of the session.
- applicationDescription : in TpApplicationDescription

  Descriptive text for informational purposes (for example, text presented on the bill and used for communication with the user).
- chargingParameters : in TpChargingParametersSet

  These parameters and their values specify to the charging service what was provided to the end user so that the charging service can determine the applicable tariff.
- amount : in TpChargingPrice

  Specifies the amount and currency to be credited to the subscriber.
- requestNumber : in TpInt32

  Specifies the number given in the result of the previous operation on this session or when creating the session. When no answer is received the same operation with the same parameters must be retried with the same requestNumber.

Raises:

- **TpCommonExceptions**: Common exceptions that typically represent platform errors. Please refer to the description in the section “TpCommonExceptions,” on page 65.
- **P_INVALID_SESSION_ID**: The sessionID parameter contains a value that is not for a currently known session.
- **P_INVALID_AMOUNT:** The argument amount is less than or equal to zero.
- **P_INVALID_REQUEST_NUMBER**: The requestNumber argument does not contain the expected value, the value that was returned from the previous callback response or from the createChargingSession() method.

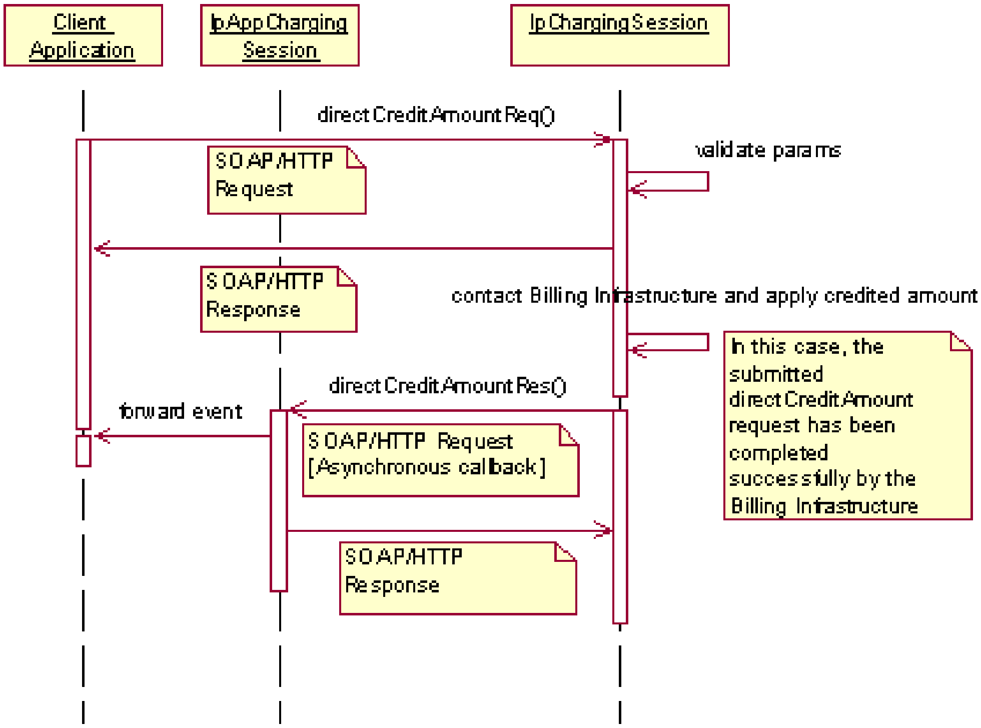
directCreditAmountReq() Successful Operation

The client application can choose to credit an amount to the subscriber’s core balance with no prior reservations made.

The request to credit is submitted to the IpChargingSession by invoking the directCreditAmountReq() method after a createChargingSession() is invoked, as shown in Figure 52, “Direct Credit of Currency.”

An exception can result if the parameter validation fails. However, at this stage, this is only preliminary verification. The method returns without waiting for credit confirmation. The SCS notifies IpAppChargingSession of a credit confirmation or credit error asynchronously using directCreditAmountRes() or directCreditAmountErr().

Figure 52 Direct Credit of Currency


Client Application
IpAppCharging Session
IpChargingSession
directCreditAmountReq()
validate params
SOAP/HTTP Request
SOAP/HTTP Response
contact Billing Infrastructure and apply credited amount
In this case, the submitted directCreditAmount request has been completed successfully by the Billing Infrastructure
directCreditAmountRes()
forward event
SOAP/HTTP Request [Asynchronous callback]
SOAP/HTTP Response

Callback Method: directCreditAmountRes()

This callback method indicates that the corresponding request was successful.

Signature:

directCreditAmountRes (sessionID : in TpSessionID, requestNumber : in TpInt32, creditedAmount : in TpChargingPrice, requestNumberNextRequest : in TpInt32) : void

Parameters:

- sessionID : in TpSessionID

  ID of the session for which the operation was called.
- requestNumber : in TpInt32

  Request Number for this request.
- creditedAmount : in TpChargingPrice

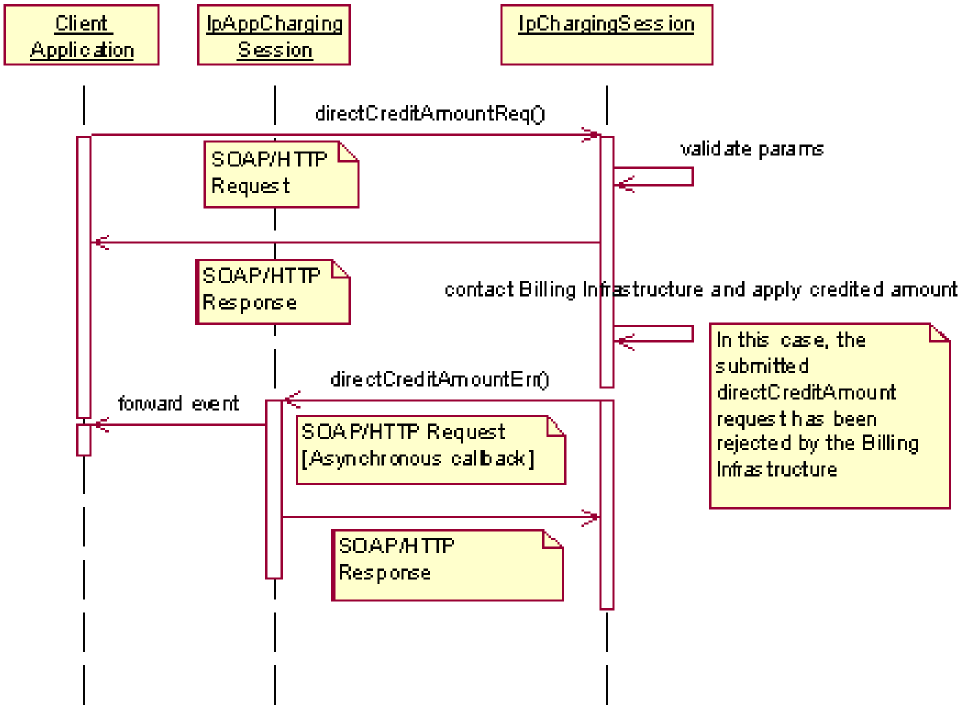
  Indicates the credited amount of currency.
- requestNumberNextRequest : in TpInt32

  This Request Number must be used in the next request (requiring a Request Number) for this session.

directCreditAmountReq() Unsuccessful Operation

Failure is communicated by SCS to IpAppChargingSession by asynchronous callback directCreditAmountErr(), as shown in Figure 53, “Direct Credit Amount – Error Scenario.”

Figure 53 Direct Credit Amount – Error Scenario


Client Application
IpAppCharging Session
IpChargingSession
directCreditAmountReq()
validate params
SOAP/HTTP Request
SOAP/HTTP Response
contact Billing Infrastructure and apply credited amount
In this case, the submitted directCreditAmount request has been rejected by the Billing Infrastructure
directCreditAmountErr()
forward event
SOAP/HTTP Request [Asynchronous callback]
SOAP/HTTP Response

Error Callback Method: directCreditAmountErr()

This error callback method indicates that the corresponding request failed completely and that no units have been debited.

Signature:

directCreditAmountErr (sessionID : in TpSessionID, requestNumber : in TpInt32,

error : in TpChargingError, requestNumberNextRequest : in TpInt32) : void

Parameters:

- sessionID : in TpSessionID

  ID of the session for which the operation was called.
- requestNumber : in TpInt32

  Request Number for this request.
- error : in TpChargingError

  Indicates the reason for failure. Possible error is:
  - P_CHS_ERR_NO_CREDIT.
- requestNumberNextRequest : in TpInt32
- This Request Number must be used in the next request (requiring a Request Number) for this session.

Design Limitation

A negative balance appears when the following scenario occurs:

1. An OSA session is established which starts charging the subscriber’s non-core balance.
2. During this time, a non-core exp job runs and expires the non-core balance and resets it to zero.

3. The original OSA transaction from step 1 is still in progress and the session terminates because of insufficient balance. This is correct behavior because the subscriber’s PO is configured with the PO minimum as zero for this non-core balance.
4. At the end of the session, the non-core balance is updated to a negative value that is equal to the total consumption amount.

Chapter 5

Diameter Infrastructure in Comverse ONE Solution

Overview

Diameter is an Internet Engineering Task Force (IETF) protocol that is extended by the 3rd Generation Partnership Project (3GPP). Diameter provides Authentication, Authorization, and Accounting (AAA) services. The 3GPP charging application is defined as an extension of the Diameter Credit-Control Application (DCCA) defined in RFC 4006. Diameter is a more capable successor to the Remote Access Dial In User Services (RADIUS) AAA protocol, which was widely and successfully deployed.

Diameter is defined in terms of a base protocol and a set of applications. This design extends the protocol to new access technologies. The base protocol provides mechanisms for reliable transport, message delivery, and error handling. The base protocol must be used in conjunction with a Diameter application. Each application relies on the services of the base protocol to support a specific type of network access.

NOTE The term “Diameter application” does not refer to a software application but to a specific extension of the base protocol developed to provide new functionality.

Diameter conveys a collection of Diameter commands and their associated Attribute Value Pairs (AVPs). The extensible nature of the Diameter protocol allows new applications to be built on the framework of the Diameter base protocol by adding new commands, each with a set of new AVPs. DCCA is only one of several applications that the Diameter base protocol supports.

Diameter clients interface to Diameter servers via transport layer connections. Each transport layer connection between client and server is secured using an encryption algorithm such as IPSec or Transport Layer Security (TLS). The clients use Diameter routing to select the appropriate transport layer connection for making requests to the server. The primary design goal is to limit the burden of configuring connections and routing within the client and server.

Diameter Base Protocol

The Diameter base protocol provides an AAA framework for applications such as network access and IP mobility. It is also intended to work in local AAA roaming situations.

The Diameter base protocol performs the following functions:

- Delivers AVPs
- Negotiates different capabilities
- Handles user sessions or accounting
- Sends error notifications
- Provides extensibility through the addition of new commands and AVPs

All of the data delivered by the base protocol is in the form of Attribute Value Pairs. Some of these AVP values are used by the Diameter base protocol itself while others deliver data associated with particular applications that employ Diameter. AVPs specific to Diameter are added and excluded according to the needs of a specific application.

The Diameter protocol works on a request/answer mechanism. The request is a message type with a Request bit set and contains a set of AVPs. The answer contains the same message type with the Request bit cleared and another set of AVPs.

The Diameter base protocol uses AVPs to support the following features:

- Transport user authentication information to enable the Diameter server to authenticate the user.
- Transport service-specific authorization information between client and servers, enabling the peers to decide whether a user’s access request is granted.

- Exchange resource usage information that is used for accounting purposes, capacity planning, and so on.
- Relay, proxy, and redirect the Diameter messages through a server hierarchy.

The Diameter base protocol uses TCP or Stream Control Transport Protocol (SCTP) as transport and IPSec or TLS to ensure the security of the connections. IPSec is mandatory for Diameter clients and servers.

NOTE Comverse ONE solution supports TCP transport and IPSec security only. SCTP and TLS are not supported.

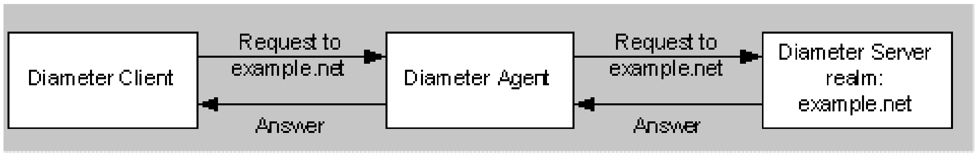
Diameter Nodes

Diameter has three types of network nodes: clients, agents, and servers, as shown in Figure 54, "Example of Diameter Realm-based Routing." A Diameter client is a device in the network that performs access control by originating AAA requests. The Diameter server handles the AAA requests of a particular application type within a particular realm. Diameter clients and servers need to support Diameter base protocol and the Diameter application required to implement the AAA service. Diameter relay agents are intermediaries between Diameter clients and servers. They forward requests and responses to the appropriate node based on the routing-related AVPs specified in the Diameter message.

An agent provides relay, proxy, or redirection service.

- Relay agents route messages while maintaining the transaction state.
- Proxy agents route messages, but also enforce different policies for different application types. Proxy agents maintain transaction state and also maintain session state for applications that are session stateful.
- Redirect agents are used when Diameter routing configuration is centralized. Redirect agents return an answer with the information necessary for Diameter agents to communicate directly. Redirect agents do not maintain transaction state or session state.

Figure 54 Example of Diameter Realm-based Routing



Diameter Routing

The Diameter base protocol specifies a set of message routing requirements which are performed at the Diameter protocol level. This represents a level 4/5 routing mechanism and is independent of standard IP routing.

The Diameter base specification defines two components associated with routing:

- **Realm**: A generic destination within the network. Realms are typically made up of multiple hosts providing the same set of services.
- **Peer Connection**: A communication channel between two Diameter Nodes (that is, an SCTP association or a TCP connection). Peer connections typically come in pairs with a primary and a secondary connection.

Diameter routing requirements specify the mechanism for selecting a transport layer connection for any given request. The selection of the connection depends on the destination realm AVP and, optionally, the destination host AVP of the request. The destination realm AVP is evaluated against a table of known realms to determine a peer connection to the destination.

Diameter clients route through intermediate Diameter agents. Relay agents and proxy agents are two types of agents that transfer requests. Each of these agents has its own characteristics.

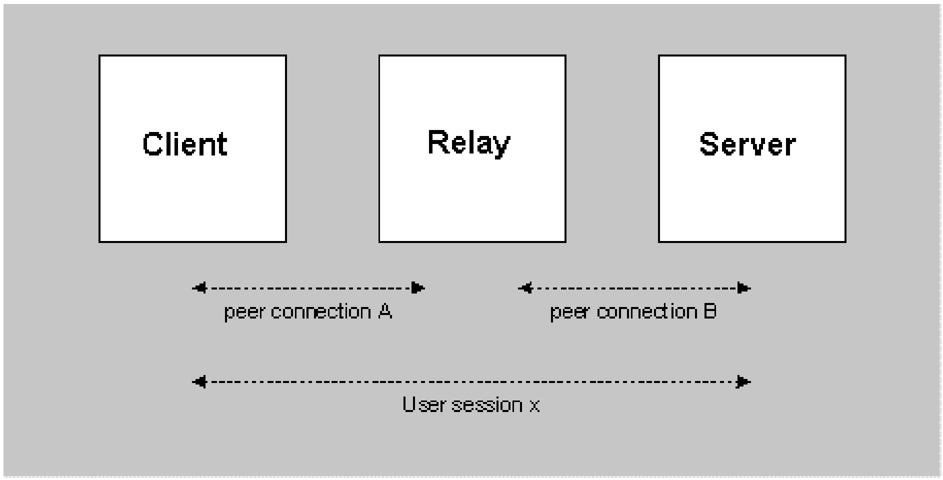
When initiating a message transmission, the Diameter client selects a peer connection to a Diameter agent based on the realm. On the next hop, the agent selects a connection that routes the message to the destination realm. Figure 54, “Example of Diameter Realm-based Routing” shows the Diameter realm-based routing.

The Diameter client uses its realm table to select a transport connection to the Diameter agent. When the Diameter agent receives the request, it uses its realm table to select a connection to the destination realm. Because the Diameter agent maintains transaction state for all requests, the response to the client is routed back along the same path.

Connections versus Sessions

A connection is a transport layer link between two peers that send and receive Diameter messages. A session is a logical concept in the application layer shared between an access device and a server. It is identified by the Session-ID AVP.

Figure 55 Diameter Connections and Sessions



In the example in Figure 55, “Diameter Connections and Sessions,” peer connection A is established between the client and its local relay agent. Peer connection B is established between the relay agent and the server. The user session X spans the client and the server via the Relay. Each user of a service sends an authorization request with a unique session identifier. Once the server accepts the request, both the client and the server are aware of the session.

NOTE There is no relationship between a connection and a session. The Diameter messages for multiple sessions are all multiplexed through a single connection.

Comverse Online Charging Server (OCS)

Comverse Online Charging Server (OCS) is an implementation of the Diameter Credit Control Application (DCCA) as extended by the 3GPP to support its additional requirements.

Comverse OCS is designed to integrate Diameter and 3GPP requirements with Comverse ONE solution functions. The functionality described by the 3GPP OCS is a natural extension of the capabilities of the Comverse ONE solution.

DCCA Description

The Diameter Credit Control Application (DCCA) is a Diameter application that supports real time charging. DCCA defines two command messages along with associated sets of AVPs:

- **Credit-Control-Request (CCR)**
- **Credit-Control-Answer (CCA)**

DCCA supports two types of charging mechanisms:

- **Session Charging**: Enables the Diameter server to allocate usage to a Diameter client. The client monitors the usage and reports on it to the server. This enables payment allocation to take place throughout the duration of the call.
- **Event Charging**: Provides an immediate debit of value from the subscriber's account.

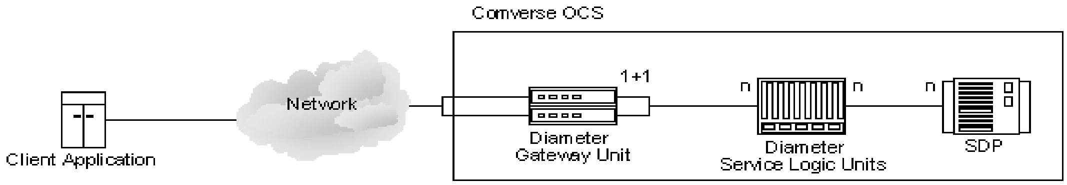
The 3GPP has extended DCCA with its own requirements. The 3GPP extension of DCCA is defined as a functional element called the Online Charging Server. The OCS handles sessions and events from charging clients that are typically various kinds of service elements in the network.

Most of the 3GPP extensions to DCCA take the form of additional AVPs that define the charging characteristics of various Diameter clients. For instance, there are sets of AVPs for Packet-Switched Data, Multi-Media-Service, and Wireless LAN.

OCS Architecture

The Comverse OCS implements the functionality of a Diameter server. The design supports high availability along with a scalable architecture. Figure 56, “Comverse ONE Solution OCS Architecture” shows the distributed OCS system.

Figure 56 Comverse ONE Solution OCS Architecture



The Comverse OCS consists of the following components:

- Diameter Gateway Unit (DGU)
- Diameter Service Logic Unit (DSLU)
- Rating database (SDP)

Service Logic Element (SLE)

The SLE reads and modifies input AVPs and maps the resulting values into the factors used to perform Charging API (CAPI) functions. Multiple SLEs exist within the same DSLU and provide client application-specific handling. For example, when there is a GPRS and SMS client, there is also a GPRS and SMS SLE.

Outbound message handling also requires specific handling based on the interfacing client. Data returned to the client may need customized modification for a specific application. The SDLU provides a means to customize message output from the system on a per-SLE basis.

The data set used to support the SLE does not need to be configurable by the customer operator. It is possible to deliver an SLE as a separate software package, independent of the base release. To avoid compatibility issues, the core application provides an interface version to which the SLE conforms. The core applications and SLE verify the version compatibility upon loading the SLE.

The Comverse ONE solution has implemented Diameter Credit Control (DCC) support so that new services are accommodated without changing the core Diameter software architecture. T support the flexible nature of DCC with different services having different rating AVPs, the SLE is used.

The service-specific logic is implemented by the SLE, which defines how the service-specific rating AVPs are analyzed and mapped into internal charging parameters. The DCC supports multiple SLEs simultaneously.

There are three SLEs supported:

- Comverse defined Diameter Charging application (Comverse SLE)
- 3GPP Diameter Packet Charging application (PS SLE)
- Flex SLE

Comverse SLE and PS SLE are devoloped using a programming language. Flex SLE allows users to meet the needs of a specific DCCA interface without programming language since it’s customized using the FLex SLE mechanism.

NOTE Application and Sub Type mapping is provisioned in the Product Catalog for each SLE. A Service Mapping table is available which provides the input parameters.

OCS State Machine

The OCS state machine provides the core application handling of the basic capabilities of the OCS. The full capabilities are listed below. Not all capabilities are required (or necessarily available) at this time.

- Event-based charges (both immediate and reservation-based)
- Session-based charges
- Charging in units (tariff applied or centralized rating) and currency (no tariff or decentralized rating)
- Assignment of initial unit types and initial consumption amounts by the server (centralized unit determination)
- Support for Advice of Charge (AoC for Information and AoC for Charging
- Credit
    - Reverse transaction
    - Non-voucher recharge
    - Simple balance credit

- Mid-session reauthorization (recalculation of reservation)
- Tariff switching
- Multiple Services within a session using subsessions
- Credit pooling for multiple services using Multiple-Services-Credit-Control
- Redirection indications as part of final unit consumption and graceful session termination.

The default messages supported by the DSLU are ACR/ACA from Base Diameter and CCR/CCA from Diameter Credit Control Application.

Each of these previous services fit into one of the following state machine models:

- Immediate Event (initial only)
- Session (initial, interim, and final)

The OCS state machine handles events from these sources:

- Diameter message events
- SDS events
- Timer events

Diameter Gateway Unit

Within the Comverse ONE solution, the Diameter Gateway Unit (DGU) provides load balancing of traffic from Diameter clients among the replicated resources of the OCS.

The DGU is the only interface between the OCS and the outside network. While it has qualities of a Diameter agent, which handles a few requests and forwards the others, it does not correspond directly to any of the existing Diameter agents. The Diameter protocol does not specifically define a gateway agent, but the DGU provides features that borrow from both the Diameter relay agent and Diameter proxy agent.

The DGU is connected to the back-end Diameter Service Logic Units (DSLUs) that provide the OCS functionality. It provides an interface point for OCS clients within a given domain or realm. The DGU/DSLU combination appears as a Diameter server with respect to the other Diameter clients and agents in the network.

To minimize the impact of the OCS within an enterprise network, a minimum number of exposed interface points are made available from the OCS. This limits the number of available transport connections and simplifies management of these interfaces. For this reason, no more than two DGUs are connected to a single Comverse ONE solution system serving the same domain.

The DGU provides the following functions:

- Distributes system and security administration over a configurable grouping of devices.
- Concentrates requests from several distributed client sets to a set of appropriate servers.
- Provides value-added processing to the requests or responses.
- Handles connection and routing to the back-end DSLUs.
- Selects and routes to the appropriate DSLU based on multiple criteria (application ID, host name, current load, response delay, response pending count, and so on.).
- Manages connections with the OCS clients and DSLUs and takes appropriate action on any failures.

The DGU also supports:

- Load balancing
- TCP connections
- IP Security (IPSec)
- Alarms and events monitoring and logging, as well as measurements (using existing infrastructure)

NOTE Comverse ONE solution supports TCP transport and IPSec security only. SCTP and TLS are not supported.

The DGU maintains a database necessary for the Diameter base protocol. This includes (but is not limited to):

- The peer table and realm table (used in routing of outbound Diameter messages).
- Origin-host(s) and Origin-realm (inserted in Answer messages to clients).
- Various values for AVPs necessary to support Diameter base protocol.

The DGU also maintains the information database of known DSLUs and the current status of these units. The DGU updates this status dynamically on DSLU availability or unavailability. The DSLUs also send information to the DGU indicating their ability to accept traffic.

NOTE The following tables must be configured on the DGU:

- Peer Table
- Service Table
- SLU Table
- SLU Routing Table

For detailed information regarding setup and configuration of the DGU, contact your Comverse support representative.

To support reliability, the DGU is deployed using a 1+1 load sharing configuration. A 1+1 configuration provides alternate paths from the Diameter client to the OCS services provided by the DSLU.

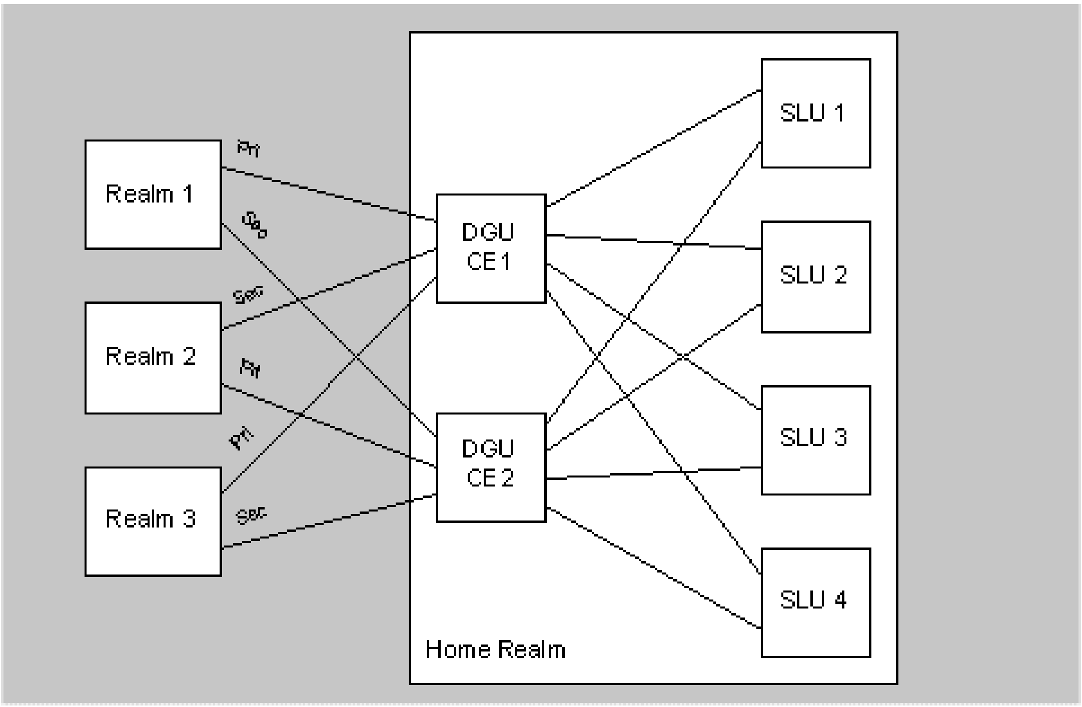
To provide scalability, the DGU interfaces to multiple DSLUs. As the processing requirements increase, DSLUs are added to increase the capacity of the system.

The DSLUs are organized in an N+1 scheme, where there is one more unit available than required to handle the expected traffic. In case of a DSLU failure, the other DSLUs share the load. However, all sessions handled by the failed DSLU are lost.

DGU Interconnection

An established connection is necessary between two peers per realm (called the primary and secondary peers), and implemented as separate computing elements (CEs) of a single DGU. Figure 57, “DGU Interconnection” shows the interconnection of the various components of OCS. Additional connections are also established, if necessary.

Figure 57 DGU Interconnection



Typically, all messages for a realm are sent to the primary peer. However, in the event of a failover procedure, pending requests are sent to the secondary peer. The implementations are free to load balance requests between a set of peers. Also, a peer can act as a primary for one realm while acting as a secondary for another realm.

On the DSLU side, each DSLU sets up a connection to each DGU computing element (CE). Only TCP connections are supported. No security method is required on the DGU to DSLU interface.

Internally, the DGU is realized as a thin proxy or relay layer to the DSLUs that implement the application part of the interface. DGU and DSLU functionality differs as follows:

- DGU implements Diameter base protocol and handles a limited set of non-session messages.
- DSLU implements the Diameter Credit Control Application (DCCA) and service logic.

The DSLU interfaces to the rating database through the Unified Rating Engine (URE) developed for other Comverse ONE solution applications.

The DSLU provides a platform for any of the Comverse implementations of the DCCA:

- Comverse Diameter Charging Interface
- Comverse Packet-Switched Charging Interface

The DGU provides only the basic set of functionality necessary to support load balancing and distribution of traffic to the DSLUs. The DSLUs bear the load of processing requests and interfacing to the subscriber database.

Even within a standard variant of the Credit Control Request (CCR), there can be variations in the AVPs sent to the OCS to support different applications. For example, while both are valid instances of the 3GPP, an SMS Diameter client provides different charging data than a GPRS Diameter client. The Comverse OCS translates this charging data into values consistent with the Comverse ONE solution rating engine.

The data interface between the DGU and DSLU uses the Diameter encoding of message formats.

Peer Connections

When a Diameter entity (a client or a server) attempts to establish a transport connection with a peer unsuccessfully, additional connection attempts are made periodically with a default interval of 30 seconds between connection attempts. A Diameter entity can refuse a connection attempt by a peer by issuing a DO_NOT_WANT_TO_TALK_TO_YOU response. On receipt of this response, the requesting entity ceases further connection attempts.

If Diameter receives data that cannot be parsed or identified as a Diameter error made by the peer, the stream is compromised and cannot be recovered. The transport connection is closed and periodic attempts are made to reestablish the connection.

The DGU listens for TCP connections from the DSLU on a well defined port (10410) selected by Comverse not to conflict with other applications. The DSLU attempts to establish a connection with the DGU on startup. The DSLU repeats the connection attempt at 30 second intervals until it is successful.

High Availability

The DGU is implemented with two computing elements (CEs). Each DGU establishes two peer connections to a Diameter client, one from each CE. For a given Diameter client, one CE is considered the primary connection and the other is the secondary connection. In the event of primary failure, Diameter failover procedures are invoked by the Diameter client and any pending requests are sent to the secondary. Alternately, the two connections are configured as load sharing.

When there are multiple clients, the primary connections are distributed across the two CEs to ensure proper load balancing. When any DGU CE fails, the remaining CE(s) handle the entire load.

Exchange of Capabilities

When two Diameter peers establish a connection, they exchange the Capabilities Exchange messages: Capabilities Exchange Request (CER) and Capabilities Exchange Answer (CEA). These messages provide a peer’s identity and capabilities, including the protocol version number, supported Diameter applications, security mechanisms, and so on.

If a Diameter peer that receives a CER message does not have any applications in common with the sending peer, it returns a CEA with the result-code AVP set to DIAMETER_NO_COMMON_APPLICATION and then disconnects the transport layer connection.

If a Diameter peer that receives a CER message does not have any security mechanisms in common with the sender, it returns a CEA with the result-code AVP set to DIAMETER_NO_COMMON_SECURITY and disconnects the transport layer connection. The DGU raises an alarm when the Capabilities Exchange fails.

The DGU does not support dynamic discovery of peers. Peers must be manually configured in the DGU peer table.

The exchange of CER/CEA messages is also implemented on the DSLU interface. On startup, the DSLU connects to the DGU and sends a CER message. The DGU validates the CER messages against the peer table. On a successful validation, a CEA with result-code of success is returned. On a failure validation, a CEA is returned with the result-code indicating an error, and the connection is closed. The DGU raises an alarm indicating DSLU interface failure.

Load Balancing and Congestion Handling

The DGU balances traffic among the backend DSLUs using a modified round robin scheme in which the handling of an application by DSLUs is prioritized based on the number of outstanding or open transactions to the DSLU. The DSLU with the lowest count has the highest priority for new sessions.

The modified round robin scheme is used only for incoming requests that do not specify a destination host DSLU (that is, new sessions). Any request specifying a destination host DSLU (that is, subsequent requests in an established session) are always routed to that DSLU.

If the DSLU detects congestion, it passes a congestion indication to the DGU. The congestion indication is passed as a Comverse-defined AVP, which is added to all the messages passed from that DSLU to the DGU. After receiving the congested indication, the DGU removes the DSLU from the round robin list and stops sending new sessions to that DSLU. This process is continued till the DSLU sets the congestion indication to available. An alarm is generated when all the DSLUs are congested.

The DGU also tracks the round trip delay for each DSLU. The thresholds are defined and alarms are generated when any threshold is crossed. The minimum, maximum, and average round trip delay measurements are collected on a per-DSLU basis.

Transport Failure Detection

The Device-Watchdog-Request (DWR) and Device-Watchdog-Answer (DWA) messages proactively detect transport failures.

A DWR is sent to a peer when no traffic has been exchanged between two peers. Both peers send this message. The watchdog interval is controlled by a timer with an initial default value of 30 seconds. A DWA is sent as a response to the Device-Watchdog-Request message.

The DWR and DWA messages are also used on the DSLU interface to check for transport failures. When a transport failure to a DSLU is detected the transport connection is closed.

Failover and Failback Procedures

Failover occurs when a transport failure with a peer is detected and all pending request messages are forwarded to an alternate agent, if possible.

The DGU does not retransmit any messages. It depends on the client and DSLU to retransmit the request messages. When a transport failure to a peer is detected, all messages in the pending message queue for that peer are dropped by the DGU.

The DGU returns an answer message with the error bit set and the Result-Code AVP set to DIAMETER_UNABLE_TO_DELIVER.

NOTE

Often multiple identical requests or answers are received as a result of a failover. The End-to-End Identifier field in the Diameter header along with the Origin-Host AVP identifies duplicate messages.

Disconnecting Peer Connections

When a Diameter node has to disconnect one of its transport connections, it sends a Disconnect-Peer-Request (DPR) message with the Disconnect-Cause AVP to inform its peer. Upon the receipt of the message, the Disconnect-Peer-Answer (DPA) is returned. This message can return an error if the the client is breaking down a connection while there are outstanding requests waiting for responses. The receiver of the Disconnect-Peer-Answer initiates the transport disconnect.

The following values are supported in the Disconnect-Cause AVP:

- **REBOOTING**: A scheduled reboot is imminent.
- **BUSY**: The peer’s internal resources are constrained and it has determined that the transport connection needs to be closed.
- **DO_NOT_WANT_TO_TALK_TO_YOU**: The peer has determined that it does not see a need for the transport connection to exist because it does not expect any messages to be exchanged in the near future.

The DEACTIVATE-PEER MML command forces a Diameter client peer disconnect from the DGU side. The DEACTIVATE-SLU MML command forces a DSLU disconnect from the DGU side.

Diameter Service Logic Unit

The DSLU is functionally equivalent to a Diameter server. It implements the Diameter Application functionality.

When a message is forwarded from the DGU to the DSLU, it carries all the data received from the Diameter client. Messages from the DSLU to the DGU have all the necessary AVPs except those directly associated with routing (for example, Origin-Host). The DGU adds any AVPs necessary to support routing or internal indications, then forwards the messages to the Diameter client.

The DSLU validates message received from the DGU against a defined message dictionary. A Diameter message is considered valid if every AVP in the message with the “Must Understand” bit set is defined in the dictionary.

NOTE The term dictionary represents an object that the software can reference to validate the message format of received messages and consult when building messages for transmission.

Message verification by DSLU is not limited to a single format. The Diameter clients from different applications and different sites can have different validity characteristics. The DSLU supports multiple message formats through the use of dictionaries. Each Diameter client application that communicates with the system could require its own dictionary. Examples of differing validity characteristics are:

- Different Diameter messages for charging functions.
- Different mandatory or allowed AVPs for the same message from different Diameter clients.

The message verification is flexible enough to define message content of acceptable messages on a per-site or per-application basis. The DSLU supports the messages Credit-Control-Request and Credit-Control-Answer.

The DSLU integrates several functions to provide services. The following are the primary functions of the DSLU:

- Handling Diameter messages
- Mapping Diameter message types and AVPs into the OCS/Comverse ONE solution state machine
- OCS/Comverse ONE solution functions
- Execution/Operation/Maintenance
- Submitting charge requests to the Unified Rating Engine (URE)

Outage Record Processing

The Comverse Diameter OCS supports Comverse ONE solution Outage Record Processing (ORP), which enables the system to continue its operation when one or more rating databases become unavailable. Outage records are produced internally by the DSLU when the rating database is completely or partially unavailable and the revenue assurance and inaccessible database features are enabled.

When configured for automatic-internal revenue assurance and inaccessible database, the OCS creates and sends outage records to the Billing Manager for calls occurring when the rating database is down. These records are later processed by the ORP to recover the revenue.

The outage records are created when the activity ends. To facilitate the completion of the activity, default rating/billing information (typically configured to limit the activity) is used for any/all information that could not be retrieved from the rating database.

General Description

Diameter OCS implements the logic to allow Diameter Charging Session to continue when the rating database becomes inaccessible and Revenue Assurance feature is enabled.

When the DIAMETER_OCS process receives the SDP_DOWN response from the URE process for any of its requests, it switches to the inaccessible rating database-specific state machine to handle the rest of the Charging Session.

Based on the state of Charging Session when a rating database failure is encountered, a default offer defined in the sytem is used to complete the session.

If a failure is encountered on an Account Validation request, a default Offer is used. Otherwise, the subscriber’s primary offer is used.

OCS generates the Outage Record for such a charging session with all necessary information to enable the Outage Record Processor (ORP) task to perform charging when the rating database becomes available and to generate CDR and activity history for the transaction.

Inaccessible Database Processing

During inaccessible database operation, any Credit Operation from the Diameter client is rejected with a Failure response. In case a rating database failure is encountered for a Charging Session with existing reservations, only single debit operation is allowed. If rating database failure is encountered before any reservation then single reservation requests followed by single debit operations are allowed.

When multiple charges per session associated with Multiple-Services-Credit control are present in the Diameter message, OCS allows multiple charging events present in the request and generate multiple ORP records corresponding to each charging event. Each charging event is treated as an independent charging event.

Outage Record Generation

When configured for automatic-internal revenue recovery and inaccessible database, the code creates and sends outage records to the OR Manager for charging sessions that occur when the rating database is down.

The ORP later processes these records to recover revenue. The outage record reflects the information for a complete charging session with the reserved and consumed amount/units for that session or subsession regardless of the stage at which the rating database failed.

Any partial updates caused by reservations against subscriber balances are rolled back if an inaccessible database condition exists and outage records are being generated at the time of final consumption.

This ensures that outage record processing does the charging only once. In case ORP is enabled and Assume Consumption flag is True, the existing behavior of the job process is modified to not consume the amount based on the reservation record. In this case, the reservation record is deleted and the charge is applied based on the Outage Record.

Diameter Connection Management and Congestion Control

Diameter connection management maintains the status of the Diameter connection and provides some congestion handling capability. This module is responsible for ensuring that the connection is operational and in case of any fault, issues an appropriate alarm.

The following resources are monitored to determine if the system is reaching a congested condition:

- **DSLU consumption of the OCS task**: Monitoring of DSLU consumption for the Diameter task is supported with the times (2) Linux system call. This function reports DSLU time consumed by the application in 10 millisecond increments. The congestion detection logic samples this value at one second intervals and looks for prolonged samples of some value (for example, 400 milliseconds or more). When this condition is detected, the application enters a congestion state.

- **Diameter transaction rate**: Transaction rate monitoring is provided as an adaptive comparison of work performed to DSLU resources consumed. When the DSLU resource is considered congested, the transaction rate is used to guide the application of the congestion control mechanism.

Once congestion is detected, an API call is made to the Diameter Transaction object to initiate congestion reporting. This causes the DWR to be sent with a Comverse-specific AVP to indicate congestion. Another API call is provided to disable congestion when the condition dissipates.

Diameter congestion control maintains the information on the state and health of a task. This information provides congestion indications to the DGU to limit new incoming traffic.

The DSLU performs the following operations when it determines a persistent limitation of resources:

- Enters a congested state
- Generates an alarm
- Invokes congestion control logic

The DSLU remains in the congested state until the resource limitation is reduced. It clears the alarm and resumes normal operation.

This provides adaptive resource monitoring to determine when to enter the congested state.

Every time an application detects a resource shortage and the congestion logic is invoked, the application enters a congestion state. This enables the flow control mechanism and also produces an alarm indicating the type of congestion.

There is a difference between the application congestion state (an alarm) and the congestion control indication presented to the DGU. The congestion control mechanism is designed to constantly set and clear the alarm, and thus alternately enable and disable traffic. This way it limits the amount of work that the application performs, yet keep it as close to the defined limits as possible.

To exit the application congested state, the resource consumption must drop below the congested level and remain below that level, indicating the condition is abated. When the application returns to the normal state and remains in the normal state for some period of time (usually one minute), the active alarms are cleared.

Configuration and Parameterization

The configuration of the OCS system is a major component of the application. A major objective of the design is to replicate the configuration of the OCS system on all DSLUs, so that there are no per unit dependencies in the OCS configuration.

Think of the configuration data as several subcomponents or sections, independent of each other. Each section is associated with, and reflects the needs of, a specific subsystem.

Configuration data is formatted using XML, stored in a file on the DSLU, and read at startup. The configuration data itself is a serialization of data objects within the application. The configuration format is governed by a content validation mechanism, either a DTD or W3C XML schema.

The <ocs> element is the root XML element and contains elements for each subsystem. The current lists of elements are:

- Diameter message handling
- Service logic elements
- Application parameters

OCS and Comverse ONE Solution Capabilities

Rating and Charging

OCS interfaces to all the charging and rating capabilities of Comverse ONE solution.

Currency Charges

Currency charges are prerated charges received from a Diameter client and applied against a currency balance. These charges are mapped to an allowed activity for the subscriber. The activity is defined from the initial application and subsystem parameters in the Comverse ONE solution factors interface. Rate modifiers, such as the location, are not considered for currency charges.

Currency charges are allowed for immediate event, event with reservation, and session-based charges. A currency charge can be made in a different currency from the subscriber's balance using currency conversion, so an appropriate conversion factor must be configured.

Centralized Unit Determination

Centralized Unit Determination is the capability to determine the type of unit to be used for the charge when an event is received. This action is triggered by the lack of a specific unit type in the received request or or by receipt of the AVP CC-Service-Specific-Units.

To support this, the function uses the initial application and the sub type from the Comverse ONE solution factors and performs a lookup in a the Diameter Unit Mapping Table. Once the unit type is found, the activity is retrieved and the normal rating and charging rules are applied.

Determination of Location Relationship

A location relationship in the Comverse ONE solution is defined as the result of comparing location A with location B. The comparison algorithm uses several database tables to achieve its result. OCS uses this capability based on values from specific AVPs identified by the given Diameter charging application.

Locations in the Comverse ONE solution are arbitrary values assigned in a hierarchical scheme. Information from the request is used to determine the location. In OCS, a location is found from the value/type combination. There are two types of combination which are considered, one for the subscriber and the other for the location. The value/type fields determine the location used by Comverse ONE solution.

The location relationship, along with home location data for the subscriber, determines an LR_ID, which is used in the AUT_Translation to find the final Application Unit Type (AUT). The other two Comverse ONE solution factors used to determine LR_ID are:

- Tele-Service-Type
- Tele-Service-Mode

In the Comverse ONE solution Diameter OCS, Tele-Service-Type is always set to NORMAL. Tele-Service-Mode is one of two values, Mode_Originating or Mode_Terminating,as shown in Table 59, "Tele-Service-Modes." For additional details regarding location relationships in Comverse ONE solution Diameter OCS refer to "Location Parameters," on page 147.

NOTE For further details on Location Relationships in the Comverse ONE solution refer to the *Product Catalog User Guide*.

Data Control and Monitoring Point (DCMP) Integration

The DCI supports an authorization request to retrieve the subscriber’s Primary Offer before starting the Diameter session.

- **SLE:** Sets AccountValidationOnly in the Charging Factor interface after parsing CCR when the authorization-only request is received. The SLE also retrieves the Primary Offer ID through SubscriberPrimaryOffer in OcsInterfaceData after the authorization-only request is handled by the OCS State Engine. The SLE builds the response message in CCA through an answer message template.
- **OCS State Engine:** Handles the authorization request by performing an account query to the Unified Rating Engine and also performs a state check for both the subscriber and the owning account in account query response. The call is rejected if the subscriber is invalid or if the subscriber (or owning account) is not in the active state.

If the SDP is down during the query, the default primary offer ID is returned via the SLE. A single measurement counter tallies the number of authorization-only requests received.

Customization Capability

Since DCC does not specify what rating parameters are to be used to rate the specific application or service being accessed by the user, there has to be an agreement between the client and server as to what AVPs are important to charging (or request), how those AVPs map to required charging factors/attributes, and any special manipulation that may be required to normalize the AVP data to match the required charge factor’s format, e.g., special prefix added to location, MSISDN, etc, as is provided by number manipulation script for other interfaces. The Comverse DCI provides a scripting capability to normalize/manipulate/customize data values provided in the CCR AVPs before it’s provided for charging and to manipulate data values for CCA AVPs before being transmitted.

NOTE The customization capability described above is applicable only to the Comverse Diameter Charging Interface.

Midsession Rate Change

There are a number of midsession service events (reauthorization triggers) that affect the rating of the current service usage. Some examples include subscriber’s change of location (roaming from one SGSN to another SGSN), change of Quality of Service during a data session or even change of session type (voice to video). The Comverse ONE solution supports midsession reauthorization capability.

SEE Refer to the Rating Technical Reference for additional information on midsession rate change.

Voice Call Roaming

A roaming voice call is a voice call made from a location that matches with one of the locations defined in Comverse ONE using location hierarchy and does not match with a location provisioned as one of the subscriber’s home locations.

The locations information is part of the CAMEL IDP message that indicates the position of the subscriber at the time of the activity initiation. A subscriber profile can have up to four such locations as “home” locations. If activity is initiated from one of these home locations, the tariff applied for the activity is counted as Home Activity tariff. If the activity is initiated from any other location, the Comverse ONE system identifies the activity as a roaming activity and the Roaming tariff is applied.

The operator may choose to apply a service charge to the subscriber for the first roaming call initiated, in addition to the normal tariff for the roaming call. All subsequent calls made from the same roaming location, or any other roaming location, would not incur any additional service charge.

If the subscriber returns to one of the home locations and initiates activity while at home and then subsequently makes another roaming voice call, this voice call is charged a service charge again.

If the subscriber is not able to pay for the service charge, the roaming call, originating or terminating, is not allowed.

T1 tariff would be the equivalent of the service charge. T2 tariff is the normal roaming tariff applied for the outgoing voice call once the service charge is applied. T2 would also be applied to any subsequent call.

A discount plan may be provisioned for this activity to apply to Tariff T1. A normal accumulator would be provisioned to increment for each usage that qualifies as roaming activity.

Record Management

The Diameter OCS supports three types of records:

- **Diameter History Records**: Used for reverse transaction and immediate event duplicate detection requests.
- **Usage Records or CDRs**: Committed to the local disk for collection by the standard process.
- **Offline Processing Records**: Contain information to reconstruct the charging activity at a later time.

NOTE Comverse Billing Suite produces usage records in Converged mode and CDRs in Prepaid standalone mode.

Usage records, CDRs, and histories contain similar information including the activity time, the usage amount, balances, accumulators, and spending limits. Application-specific information is placed in an extension field and formatted as tag/value pairs.

Events and Alarms

The support for indications from the system is based on OMNI events and alarms. The events are posted for the following conditions:

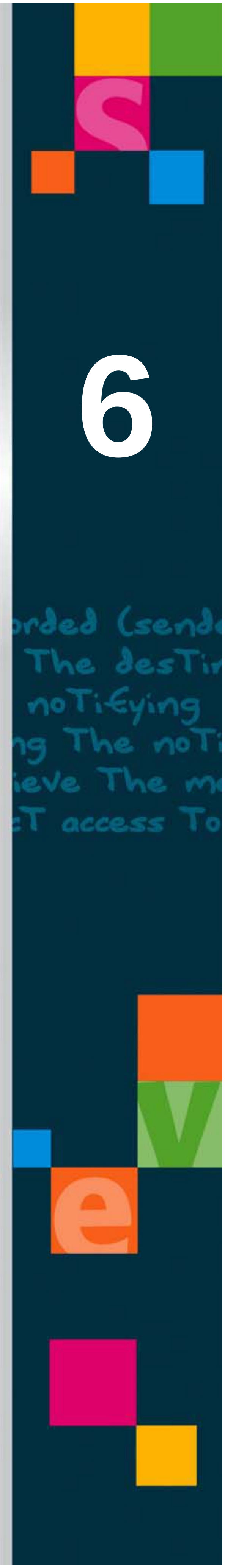
- Successful connection to the DGU
- MML commands that cause the system to refuse additional traffic
- Opening and closing of a trace file for a subscriber

The alarms are generated for the following conditions:

- Loss of the Diameter connection to the DGU
- Persistent congestion (This alarm must be set for a minimum period to avoid alarm floods.)

Chapter 6

Comverse Diameter Charging Interface



Overview

The Comverse Diameter Charging Interface (DCI) enhances the real-time Diameter Credit Control Application (DCCA) defined in RFC 4006, to provide the rating parameters required by Comverse ONE solution for Event- and Session-based rating and charging.

This interface enables Comverse ONE solution to act as a Diameter Credit Control Server (DCCS) to provide mapping between Diameter messages and Comverse ONE solution Charging Factors, and thus provide most of the existing Comverse ONE solution rating and charging capabilities to Diameter Credit Control Client (DCCC) applications.

RFC 4006 defines the basic charging interface between Diameter Credit Control Client and Server applications. The RFC 4006 Credit Control Request (CCR) message does not provide all the parameters which are required to invoke Comverse ONE solution rating and charging capabilities. The Comverse DCI defines those parameters and is used by a Diameter client to send relevant information to Comverse ONE solution within a CCR message.

Most of the Comverse ONE solution rating and charging functionalities currently provided by the Event Charging and OSA interfaces are provided by DCI. This enables charging requests from Value-Added Services (VAS) providers using DCCA, including applications from Comverse (SMS, FunDial, Voicemail) and external VAS providers. Comverse DCI also handles requests for rating and charging voice calls via Diameter.

Comverse DCI Basic Operation

The Comverse Diameter Charging Interface (DCI) allows Diameter Credit Control Client (DCCC) applications to invoke Comverse Billing Suite rating and charging capabilities. The actual DCCC applications that invoke this interface have varied needs, which are met by using the generic parameters that DCI provides.

DCI Message Details

Comverse DCI is an implemetation of the Diameter Credit Control Application (DCCA) defined in RFC 4006. DCCA includes two messages to support real time charging capabilities.

- **Credit Control Request (CCR)**: sent from a Diameter Credit Control Client (DCCC) to request charging or rating from a Diameter Credit Control Server (DCCS).
- **Credit Control Answer (CCA)**: sent as a response from DCCS to DCCC.

NOTE

Comverse DCI supports only the Credit Control Request (CCR) and Credit Control Answer (CCA) messages. It does not support the Accounting Control Request (ACR) and Accounting Control Answer (ACA) messages.

Credit Control Request

Table 50, “Credit-Control-Request AVP Components” shows the Attribute Value Pair (AVP) structure of the CCR message.

Table 50 Credit-Control-Request AVP Components

AVP Name	Rating and Charging AVP?	Type	Multiple Instances OK?
Session-Id	N	Mandatory	N
Origin-Host	N	Mandatory	N
Origin-Realm	N	Mandatory	N
Destination-Realm	N	Mandatory	N
Auth-Application-Id	N	Mandatory	N
Service-Context-Id	Y	Mandatory	N
CC-Request-Type	Y	Mandatory	N
CC-Request-Number	N	Mandatory	N
Destination-Host	N	Optional	N
User-Name	N	Optional	N
CC-Sub-Session-Id	N	Optional	N
Acct-Multi-Session-Id	N	Optional	N
Origin-State-Id	N	Optional	N
Event-Timestamp	Y	Optional	N
Event-Timestamp	Y	Optional	N
Subscription-Id	Y	Optional	Y
Service-Identifier	N	Optional	N
Termination-Cause	N	Optional	N
Requested-Service-Unit	Y	Optional	N
Requested-Action	Y	Optional	N
Used-Service-Unit	N	Optional	Y
Multiple-Services-Indicator	N	Optional	N
Multiple-Services-Credit-Control	N	Optional	Y
Service-Parameter-Info	Y	Optional	Y
CC-Correlation-Id	N	Optional	N
User-Equipment-Info	N	Optional	N
Proxy-Info	N	Optional	Y
Route-Record	N	Optional	Y
AVP	N	Optional	Y

The details of the AVPs in CCR message are defined in Diameter Base RFC 3588 and DCC Application RFC 4006. Some of the relevant AVPs specific to charging are detailed in the remainder of this chapter.

Service-Context-Id AVP

The Service-Context-Id AVP determines the Charging Application. An appropriate value must be configured for this AVP for a particular implementation, such as Comverse DCI.

CC-Request-Type AVP

The CC-Request-Type AVP is of type Enumerated and contains the reason for sending the credit-control request message. It must be present in all Credit-Control-Request messages. The following values are defined for the CC-Request-Type AVP in RFC 4006:

- INITIAL_REQUEST (CC-Request-Type=1).

  Used to initiate a credit-control session and contains credit control information that is relevant to the initiation.

- UPDATE_REQUEST (CC-Request-Type=2)

  Contains credit-control information for an existing credit-control session. Update credit-control requests are sent every time a credit-control reauthorization is needed at the expiration of the allocated quota or validity time.

  Additional service-specific events can trigger further Update requests.

- TERMINATION_REQUEST (CC-Request-Type=3)

  Sent to terminate a credit-control session. Contains credit-control information relevant to the existing session.

- EVENT_REQUEST (CC-Request-Type=4)

  Used when there is no need to maintain any credit-control session state in the credit-control server. Contains all information relevant to the service and is the only request of the service. The reason for the Event request is further detailed in the Requested-Action AVP. The Requested-Action AVP must be included in the Credit-Control-Request message when CC-Request-Type is set to EVENT_REQUEST.

Requested-Action AVP

The Requested-Action AVP is of type Enumerated and contains the requested action sent by the Credit-Control-Request command when the CC-Request-Type is set to EVENT_REQUEST. The following values are defined for the Requested-Action AVP:

- DIRECT_DEBITING (Requested-Action = 0)

  Indicates a request to decrease the end user's account balance according to information specified in the Requested-Service-Unit AVP or the Service-Identifier AVP (additional rating information is included in service-specific AVPs or in the Service-Parameter-Info AVP). The Granted-Service-Unit AVP in the Credit-Control-Answer command contains the debited units.

- REFUND_ACCOUNT (Requested-Action = 1)

  Indicates a request to increase the end user's account balance according to information specified in the Requested-Service-Unit AVP or Service-Identifier AVP (additional rating information is included in service-specific AVPs or in the Service-Parameter-Info AVP). The Granted-Service-Unit AVP in the Credit-Control-Answer command contains the refunded units.

- CHECK_BALANCE (Requested-Action = 2)

  Indicates a balance check request, done without any credit reservation from the account. The Check-Balance-Result AVP in the Credit-Control-Answer command contains the result of the balance check.

- PRICE_ENQUIRY (Requested-Action = 3)

  Indicates a price enquiry request. In this case, neither checking of the account balance nor reservation from the account is done. Only the price of the service is returned in the Cost-Information AVP in the Credit-Control-Answer Command.

NOTE The CHECK_BALANCE (2) and PRICE_ENQUIRY (3) options are not supported by Comverse DCI.

Requested-Service-Unit AVP

The Requested-Service-Unit AVP is of type Grouped and contains the number of requested units specified by the Diameter Credit Control Client. A server is not required to implement all the unit types, and it must treat unknown or unsupported unit types as invalid AVPs.

Table 51 Requested-Service-Unit AVP Components

AVP Name	Rating and Charging AVP?	Type	Multiple Instances OK?
CC-Time	Y	Optional	N
CC-Money	Y	Optional	N
CC-Total-Octets	Y	Optional	N
CC-Input-Octets	Y	Optional	N
CC-Output-Octets	Y	Optional	N
CC-Service-Specific-Units	Y	Optional	N
AVP	Y	Optional	Y

There are three possible scenarios for the content of the Requested-Service-Unit AVP:

- **Centralized Unit Determination / Centralized Rating**: If the DCCC does not send the Requested-Service-Unit AVP, the Comverse DCI determines the units to be granted and calculates the rate.
- **Decentralized Unit Determination / Centralized Rating**: If the DCCC sends a Requested-Service-Unit AVP that does not contain CC-Money, but does contain CC-Service-Specific-Units (used to set Requested Units), the client determines the units and DCI determines the rate.
- **Decentralized Unit Determination / Decentralized Rating**: If the DCCC sends a Requested-Service-Unit AVP that contains CC-Money, the client determines both the units and the rate. The DCI is only responsible for applying the monetary charge.

NOTE Service-specific-units are mapped to an internal unit based on provisioning.

NOTE Comverse DCI supports only single instance of a unit in the Requested-Service-Unit AVP.

Used-Service-Unit AVP

The Used-Service-Unit AVP is of type Grouped and contains the number of used units measured from the point when the service became active or, if interim interrogations are used during the session, from the point when the previous measurement ended.

Table 52 Used-Service-Unit AVP Components

AVP Name	Rating and Charging AVP?	Type	Multiple Instances OK?
Tariff-Change-Usage	Y	Optional	N
CC-Time	Y	Optional	N
CC-Money	Y	Optional	N
CC-Total-Octets	Y	Optional	N
CC-Service-Specific-Units	Y	Optional	N
AVP	Y	Optional	Y

Credit Control Answer

The CCA message consists of following AVPs as defined in RFC 4006:

Table 53 AVP Components

AVP Name	Rating and Charging AVP?	Type	Multiple Instances OK?
Session-Id	N	Mandatory	N
Result-Code	Y	Mandatory	N
Origin-Host	N	Mandatory	N
Origin-Realm	N	Mandatory	N
Auth-Application-Id	N	Mandatory	N
CC-Request-Type	N	Mandatory	N
CC-Request-Number	N	Mandatory	N
User-Name	N	Optional	N
CC-Session-Failover	N	Optional	N
CC-Sub-Session-Id	N	Optional	N
Acct-Multi-Session-Id	N	Optional	N
Origin-State-Id	N	Optional	N
Event-Timestamp	N	Optional	N
Granted-Service-Unit	Y	Optional	N
Multiple-Services-Credit-Control	N	Optional	Y
Cost-Information	Y	Optional	N
Final-Unit-Indication	Y	Optional	N
Check-Balance-Result	N	Optional	N
Credit-Control-Failure-Handling	Y	Optional	N
Direct-Debiting-Failure-Handling	Y	Optional	N
Validity-Time	Y	Optional	N
Redirect-Host]	N	Optional	Y
Redirect-Host-Usage	N	Optional	N
Redirect-Max-Cache-Time	N	Optional	N
Proxy-Info	N	Optional	Y

Table 53 AVP Components (Continued)

AVP Name	Rating and Charging AVP?	Type	Multiple Instances OK?
Route-Record	N	Optional	Y
Failed-AVP	N	Optional	Y
AVP	N	Optional	Y

The Comverse DCI uses the CCA message definition as defined in RFC 4006. No enhancements are required in this message to support Comverse ONE solution charging capabilities.

Most of the AVPs required in the CCA message are automatically populated based on input parameters of the CCR message. Table 54, "AVPs in CCA Automatically Populated Based on Input Parameters in CCR" details these AVPs.

Table 54 AVPs in CCA Automatically Populated Based on Input Parameters in CCR

AVP	Description
Session-ID	The Session ID of the current session, received from the corresponding request.
Result-Code	This value is dependent on the exception generation of the DCI system. When an exception is made in the DCI system the error code is set. Otherwise the Result-Code is set to DIAMETER_SUCCESS
Origin-Host	Filled in with a parameterized value
Origin-Realm	Filled in with a parameterized value
CC-Request-Type	Copied from the Request message
CC-Request-Number	The current request number in a multiple transaction session
Origin-State-Id	Value automatically managed by the DCI system. It represents a new value for every start of the system.
Proxy-Info	Copy every instance of this AVP in the Request message into the Answer

Table 55, "AVPs in CCA Populated Based on Rating Result Parameters" provides a list of AVPs in the CCA message which are populated dynamically based on the rating result parameters.

Table 55 AVPs in CCA Populated Based on Rating Result Parameters

AVP	Description
Granted-Service-Unit	The current amount of reserved volume in a reservation-based session.
Validity-Time	The lifetime of the reservation in seconds.
Cost-Information	The estimated cost of the charge. Used to support AoC services.
Event-Timestamp	The timestamp of the event. The value of this field depends on whether the client provided the timestamp or the DCI node provided the timestamp.

Application Sub Type Determination

Application and Sub Type are internal factors used to perform rating in the Comverse ONE solution. Application and Sub Type are derived from input parameters. One of the functions of the Comverse DCI is to determine the Application and Sub Type. This is performed through the use of a table provisioned using the Product Catalog (PC) interface. Table 56, "Diameter Service Mapping Table" shows this table.

Table 56 Diameter Service Mapping Table

Column	Type	Description
SCID	Mandatory	The Service-Context-ID AVP value of the CCR-Initial or CCR-Event message
Client Account ID	Optional	Identifier of the external clients/merchants sending the charging request via Diameter client.
Service Type	Optional	Service-Parameter-Info: Service Type AVP
Service Sub Type	Optional	Service-Parameter-Info: Service Sub Type AVP
Quality of Service	Optional	Service-Parameter-Info: Quality of service AVP
Custom1	Optional	Service-Parameter-Info:Custom1 AVP
Custom2	Optional	Service-Parameter-Info:Custom2 AVP
Custom3	Optional	Service-Parameter-Info:Custom3 AVP
Custom4	Optional	Service-Parameter-Info:Custom4 AVP
Custom5	Optional	Service-Parameter-Info:Custom5 AVP
Application	Result	Application identifier assigned to a specific type of activity, for example, Voice Call, MPEG, MOSMS, and so on.
Sub Type	Result	Identifier of the Sub Type associated with this Application activity.

Service-Parameter-Info AVP

Comverse Diameter Charging Interface enhances the CCR message by defining additional parameters in the Service-Parameter-Info AVP. Service-Parameter-Info is an optional grouped AVP (zero or more can be present) and is used to add service-specific information to the CCR message. It has the following definition,

- Service-Parameter-Info ::= < AVP Header: 440 >
  { Service-Parameter-Type }
  { Service-Parameter-Value }
- Service-Parameter-Type is of type Unsigned32.
- Service-Parameter-Value is of type OctetString.

The DCCC interacting with Comverse DCI uses Service-Parameter-Info AVPs to provide additional information required by Comverse ONE solution to handle the rating and charging requests of that Diameter client. Depending on the nature of the charging request, a Diameter client sends any subset of service parameters as defined in a CCR request. The actual list of service parameters used is determined based on the needs of the client application on a per-implementation basis.

Encoding of the Service-Parameter-Value AVP always uses UTF8String.

Table 57 Comverse DCI Service-Parameter-Info AVP Usage

Service-Parameter-Type	Description	Service-Parameter-Value (Example Value)
1	**Client Account ID**: This parameter identifies the actual client application which is invoking the transaction via Diameter client.	1112345, rockringers, mycricket.com
2	**Service Type**: This parameter directly maps to the Comverse ONE solution Charging factor Application or is used as an application/service identifier to derive a Comverse ONE solution Application and Sub Type.	SMS, MMS, GPRS, RINGTONE, FUNDIAL, 123, 100
3	**Service Sub Type**: This parameter directly maps to the Charging factor Sub Type or is used to derive the Application and Initial Sub Type.	Local, On-net, Off-net, None
4	**Quality of Service**: This parameter derives Application and Initial Sub Type.	1, 2, 64K, 512K, any
5	**Timezone**: Time zone of the subscriber	Seconds of offset from GMT
6	**Comment**: Comment string used in CDR record	
7	**Origination Location Type**	Enumerated value. Refer to Table 58, "Location Relationship Types."
8	**Origination Location**: Location of Originating party, for example, MSC ID, SGSN ID, Cell ID.	UTF8String
9	**Destination Location Type**	Enumerated value. Refer to Table 58, "Location Relationship Types."
10	**Destination Location**: Location of the Terminating party, for example, MSC ID, SGSN ID, Cell ID.	UTF8String
11	**Reserved**	
12	**Reserved**	
13	**Reserved**	
14	**Reserved**	
15	**Reserved**	
16	**Reserved**	
17	**Special Feature Digits**: Prefix for special feature invocation, for example Calling Circle, Friends and Family.	133, 2569
18	**Service Mode**	Enumerated value. Refer to Table 59, "Tele-Service-Modes."
19	**Reverse Transaction Session ID:** The Session ID of a previous Diameter transaction (Event or session-based) which needs to be reversed. This parameter must be present in a CCR (event) with refund request.	
101	**Custom1**: TBD for a specific customer or application implementation	
102	**Custom2**: TBD for a specific customer or application implementation	

Table 57 Comverse DCI Service-Parameter-Info AVP Usage (Continued)

Service-Parameter-Type	Description	Service-Parameter-Value (Example Value)
103	**Custom3**: TBD for a specific customer or application implementation	
104	**Custom4**: TBD for a specific customer or application implementation	
105	**Custom5**: TBD for a specific customer or application implementation	

Table 58 Location Relationship Types

Enumerated Token	Value	Description
HANDSET_MAP	1	Handset Map
TELEPHONE	2	Telephone
MSRN	3	Mobile Station Roaming Number
CELL_ID	4	Cell ID
MSC_ID	5	Mobile Switching Center ID
SGSN_ID	6	Serving GPRS Support Node ID
IP_ADDRESS	7	Internet Protocol Address

Table 59 Tele-Service-Modes

Enumerated Token	Value	Description
Mode_Originating	0	Represents an originating service (voice call origination)
Mode_Terminating	1	Represents a terminating service (receiving a voice call)

Table 60 Subscriber ID Type (from RFC 4006)

Enumerated Token	Value	Description
END_USER_E164	0	Subscriber ID is of type E.164
END_USER_IMSI	1	Subscriber ID is based on E.212 number (International Mobile Subscriber ID)

Comverse DCI Request Handling

The Comverse ONE solution system contains a parameter called Application Usage Type (AUT). An AUT consists of Application, Sub Type, and Unit Type.

The most important action of Comverse DCI Request handling is the selection of the Application and Sub Type parameters used to initiate the billing process. This selection is based on a comparison of several AVPs against columns in Table 61, "Diameter Service Mapping Table." The result of the table lookup is the Application and Sub Type parameters. Refer to the section "Application Sub Type Determination," on page 142.

Table 61 Diameter Service Mapping Table

Column	Type	Description
Version ID	Mandatory	System-assigned version number of the database.
SCID	Mandatory	The Service-Context-ID AVP value of the CCR-Initial or CCR-Event message.
Client Account ID	Optional	Identifies external clients/merchants which are sending the charging request via Diameter client.
Service Type	Optional	Service-Parameter-Info: Service Type AVP
Service Sub Type	Optional	Service-Parameter-Info: Service Sub Type AVP
Quality of Service	Optional	Service-Parameter-Info: Quality of Service AVP
Custom1	Optional	Service-Parameter-Info:Custom1 AVP
Custom2	Optional	Service-Parameter-Info:Custom2 AVP
Custom3	Optional	Service-Parameter-Info:Custom3 AVP
Custom4	Optional	Service-Parameter-Info:Custom4 AVP
Custom5	Optional	Service-Parameter-Info:Custom5 AVP
Application	Result	Identifies application assigned to a specific type of activity. For example, Voice Call, MPEG, MOSMS, and so on.
Sub Type	Result	Identifies Sub Type associated with this application activity.

Unit Determination

In addition to selection of the Application and Sub Type, if the received message contains the Requested-Service-Unit AVP, Comverse DCI must also inspect this AVP to determine the Unit Type to be used in Comverse ONE solution. If the Requested-Service-Unit AVP is not present, then DCI reverts to Centralized Unit Determination to select the correct unit.

Table 62, “Requested-Service-Unit Member AVP to Comverse ONE Solution Unit Type Mapping” identifies the rules for setting Comverse ONE solution Unit Type if the Requested-Service-Unit AVP is present. The mechanism inspects the member AVP of the Requested-Service-Unit AVP to determine the unit.

Table 62 Requested-Service-Unit Member AVP to Comverse ONE Solution Unit Type Mapping

Requested-Service-Unit member AVP	Comverse ONE Solution Unit Type
CC-Money	CURRENCY
CC-Time	SECONDS
CC-Total-Octets	OCTETS
CC-Service-Specific-Units	For this AVP, the Diameter Unit Mapping Table correlates the Application and Sub Type to the Comverse ONE solution Unit Type.

NOTE

When the member AVP is CC-Money, the AVPs CC-Money.Unit-Value.Exponent and CC-Money.Currency are used to provide more information.

Location Parameters

When tariffs are determined based on the location of the subscriber or destination, the location parameters must be extracted from the received Diameter message and transferred to the Comverse ONE solution. Two pairs of parameters are used for location: one for subscriber location and the other for the location of the destination or some other location.

Location relationships do not always require both the subscriber and the other location to be present. Absence of either causes the DCI to provide the default location for the missing element. For consistency with telecommunication applications, the subscriber location and the other location are dependent on the tele-service-mode parameter.

When tele-service-mode is tsm_Mode_Originating (and also default behavior):

- Subscriber-location is set from Service-Parameter-Info: Origination Location
- Subscriber-location-type is set from Service-Parameter-Info: Origination Location Type
- Other-location is set from Service-Parameter-Info: Destination Location
- Other-location-type is set from Service-Parameter-Info: Destination Location Type

When tele-service-mode is tsm_Mode_Terminating:

- Subscriber-location is set from Service-Parameter-Info: Destination location
- Subscriber-location-type is set from Service-Parameter-Info: Destination Location Type
- Other-location is set from Service-Parameter-Info: Origination Location
- Other-location-type is set from Service-Parameter-Info: Origination Location type

Diameter Answer Handling

Table 63, “Answer Unit Type Mapping” identifies the Comverse ONE solution Unit mapping to the member AVP of the Granted-Service-Unit. DCI fetches the reserved-unit-type from the Result Interface and adds the appropriate AVP.

Table 63 Answer Unit Type Mapping

reserve-unit-type	Granted-Service-Unit AVP
CURRENCY	CC-Money.Unit-Value.Value-Digits
SECONDS	CC-Time
OCTETS	CC-Total-Octets
Default	CC-Service-Specific-Units

Diameter Answer Result Codes

All Diameter requests sent to the DCI are acknowledged with a Diameter Answer, however receipt of a Diameter Answer does not constitute a successful operation. The Diameter Answer contains the Result-Code AVP that must be inspected by the Diameter client to determine the success of the operation. Any Result-Code other than DIAMETER_SUCCESS, such as DIAMETER_LIMITED_SUCCESS, indicates that there was an error. In addition, certain errors indicate that the reservation-based session is now closed.

The Result-Code DIAMETER_SUCCESS indicates that the operation was completely successful. The Result-Code DIAMETER_LIMITED_SUCCESS indicates that some part of the operation did not succeed. For example, if the subscriber has limited funds and the granted amount is only a portion of the requested amount, then DIAMETER_LIMITED_SUCCESS is returned. Regardless, the Granted-Service-Units AVP must be consulted in all cases to determine the allowed quota.

Table 64, “Result Codes” lists possible values for the Result-Code AVP in the Answer message.

Table 64 Result Codes

Enumerated Token	Value	Description
DIAMETER_SUCCESS	2001	The complete operation was successful.
DIAMETER_LIMITED_SUCCESS	2002	A part of the operation did not succeed. For example, if the subscriber has limited funds and the granted amount is only a portion of the requested amount.
DIAMETER_UNABLE_TO_DELIVER	3002	The DCI was unable to process the message. This can occur due to a problem decoding the message or no connection to the peer host.
DIAMETER_REALM_NOT_SERVED	3003	The DCI received a message that did not contain an Origin-Host AVP.
DIAMETER_TOO_BUSY	3004	The DCI cannot process any more transactions because it has reached its maximum number of concurrent transactions.
DIAMETER_INVALID_AVP_BITS	3009	The Origin-Host AVP is malformed.
DIAMETER_UNKNOWN_PEER	3010	The configured peer host does not match the value of Origin-Host AVP in Capabilities Exchange Request or Capabilities Exchange Answer.
DIAMETER_AUTHENTICATION_REJECTED	4001	The DCI determined that the user account is invalid.
DIAMETER_OUT_OF_SPACE	4002	The DCI could not allocate space for a session.
DIAMETER_END_USER_SERVICE_DENIED	4010	DCI determined that the service was denied. The possible reasons include: ■ The activity is not allowed in the current subscriber state ■ Cannot load the account data ■ An exception condition occurred while processing an event or session
DIAMETER_CREDIT_LIMIT_REACHED	4012	The reservation limit for the user was reached.
DIAMETER_UNKNOWN_SESSION_ID	5002	DCI received an unknown session ID. For example, it received a session ID in a CCR-Update or CCR-Terminate which does not match the session ID for any currently open sessions.
DIAMETER_INVALID_AVP_VALUE	5004	The DCI was unable to retrieve data from an AVP.
DIAMETER_MISSING_AVP	5005	A mandatory AVP is missing from the message.
DIAMETER_NO_COMMON_APPLICATION	5010	DCI was unable to process the message. The possible reasons include: ■ Auth-Application-Id AVP does not match values configured in DC ■ Acct-Application-Id AVP does not match values configured in DCI ■ No service logic to handle the message ■ There was a problem decoding the message.

Table 64 Result Codes (Continued)

Enumerated Token	Value	Description
DIAMETER_UNABLE_TO_COMPLY	5012	DCI was unable to process the message. The possible reasons include: ■ AVP(s) are missing. ■ An Application and Sub Type cannot be determined. ■ Capabilities Exchange Request was received with an Application ID not equal to 0.
DIAMETER_USER_UNKNOWN	5030	DCI was unable to access the user's account.
DIAMETER_RATING_FAILED	5031	DCI had an error with the reservation on the user's account, or there was no tariff plan, or the amount or unit was invalid.

Client Requirements for Sessions Using Multiple Transactions

When the charging scenario requires multiple Diameter transactions for the same session ID, there is one additional requirement for the Diameter client application when interfacing to the Comverse DCI. This includes any scenario that starts with a CCR with the CC-Request-Type AVP as INITIAL_REQUEST and is followed by a CCR with the CC-Request-Type AVP set to UPDATE_REQUEST or TERMINATION_REQUEST.

To ensure proper operation, the client application must populate the Destination-Host AVP in each subsequent CCR message for the same Session-ID that it sends to the Comverse DCI . The value for the Destination-Host AVP is reported to the client application in the Origin-Host AVP from the last CCA message received from the Comverse DCI for the corresponding Session-ID. The value of the Origin-Host AVP in the CCA identifies the DCI subsystem that is handling the session. Providing Destination-Host AVP in the CCR for UPDATE_REQUEST and TERMINATION_REQUEST ensures that the correct subsystem is addressed for these subsequent requests.

Credit Control Failure Handling (CCFH) and Session Failover

Two optional AVPs in the CCA are the Credit-Control-Failure-Handling AVP and the CC-Session-Failover AVP. The Comverse DCI can send any value in these AVPs based on configuration within the DCI. The usage of these AVPs is at the Diameter client end. A Comverse DCI system is normally deployed in a mated-pair configuration of the front-end. If a front-end node fails, a request can be routed via the other front-end node to the back-end node handling the session. If a back-end node handling a Diameter session fails, the Comverse DCI cannot recover that session.

Charging Scenarios

The charging scenarios supported by Comverse DCI are:

- Immediate Event Charge
- Event with Reservation
- Session Charge
- Reverse Transaction Credit (Refund)

Each of the first three scenarios can have the following variations:

- **Decentralized Unit determination/Decentralized Rating (Currency charge)**: The Requested-Service-Unit AVP is present and contains a monetary unit (CC-Money).
- **Decentralized Unit determination/Centralized Rating**: The Requested-Service-Unit AVP is present and contains a non-monetary unit.

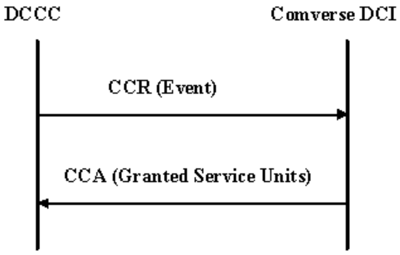
- **Centralized Unit determination/Centralized Rating**: The Requested-Service-Unit AVP is not present.

The expected AVPs in the initial request depend on which of these charging variations exists, but the rules are applied consistently across all scenarios.

Immediate Event Charge

The Diameter client uses the Immediate Event Charge scenario, shown in Figure 58, “Immediate Event Charge,” to request a one-time debit. For the Immediate Event request, a Success response is only sent if the subscriber has enough balance to cover all the requested units from the Requested-Service-Unit AVP or, in the case of Centralized Unit Determination, the reserve amount for the activity. Partial units are not granted for an event request.

Figure 58 Immediate Event Charge



The following Table 65, “Credit Control Request – Immediate Event” describes the mapping between DCI and Diameter.

Table 65 Credit Control Request – Immediate Event

AVP	Type	Description
Session-Id	Mandatory	Session ID
Origin-Host	Mandatory	Origin Host
Origin-Realm	Mandatory	Origin Realm
Destination-Realm	Mandatory	Destination Realm
Auth-Application-Id	Mandatory	4
Service-Context-Id	Mandatory	Identifies the service logic at the DCI
CC-Request-Type	Mandatory	EVENT_REQUEST
CC-Request-Number	Mandatory	Request Number
Requested-Action	Mandatory	DIRECT_DEBITING
Subscription-Id: Subscription-Id-Type	Mandatory	Uses the values MSISDN or IMSI. Refer to Table 60, “Subscriber ID Type (from RFC 4006).”
Subscription-Id: Subscription-Id-Data	Mandatory	UTF8String
Requested-Service-Unit	Optional	Possible member AVPs are listed in Table 62, “Requested-Service-Unit Member AVP to Comverse ONE Solution Unit Type Mapping.”.
Event-Timestamp	Optional	Timestamp of the charge event
Service-Parameter-Info: Client Account ID	Optional	Identifies the Diameter client

Table 65 Credit Control Request – Immediate Event (Continued)

AVP	Type	Description
Service-Parameter-Info: Service type	Optional	Service type
Service-Parameter-Info: Service subtype	Optional	Service Sub Type
Service-Parameter-Info: Quality of Service	Optional	Quality of Service
Service-Parameter-Info: Time zone	Optional	Time zone
Service-Parameter-Info: Comment	Optional	The value in this AVP is transferred to the CDR.
Service-Parameter-Info: Origination Location Type	Optional	Refer to section "Location Parameters," on page 147.
Service-Parameter-Info: Origination Location	Optional	Refer to section "Location Parameters," on page 147.
Service-Parameter-Info: Destination Location Type	Optional	Refer to section "Location Parameters," on page 147.
Service-Parameter-Info: Destination Location	Optional	Refer to section "Location Parameters," on page 147.
Service-Parameter-Info: Special Feature Digits	Optional	Prefix for a special feature invocation. For example, Calling Circle, Friends and Family.
Service-Parameter-Info: Service Mode	Optional	Defines originating or terminating call type. Possible values are listed in Table 59, "Tele-Service-Modes.".
Service-Parameter-Info: Custom1	Optional	TBD for a specific Customer/ Application implementation
Service-Parameter-Info: Custom2	Optional	TBD for a specific Customer/ Application implementation
Service-Parameter-Info: Custom3	Optional	TBD for a specific Customer/ Application implementation
Service-Parameter-Info: Custom4	Optional	TBD for a specific Customer/ Application implementation
Service-Parameter-Info: Custom5	Optional	TBD for a specific Customer/ Application implementation

Table 66 Credit Control Answer – Immediate Event

AVP	Type	Description
Session-Id	Mandatory	Session ID
Result-Code	Mandatory	Result of the operation
Origin-Host	Mandatory	Origin Host
Origin-Realm	Mandatory	Origin Realm
Auth-Application-Id	Mandatory	4
CC-Request-Type	Mandatory	INITIAL_REQUEST
CC-Request-Number	Mandatory	From initial request
Origin-State-Id	Mandatory	Origin State ID
Event-Timestamp	Mandatory	Timestamp of the charge event
Granted-Service-Unit	Mandatory	Number of units that were granted
CC-Time	Optional	Time
CC-Money	Optional	Money
CC-Total-Octets	Optional	Total Octets
CC-Input-Octets	Optional	Input Octets

Table 66 Credit Control Answer – Immediate Event (Continued)

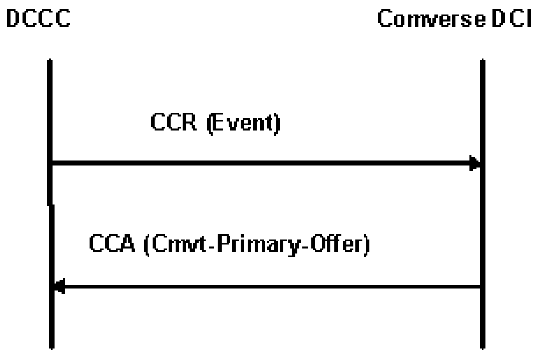
AVP	Type	Description
CC-Output-Octets	Optional	Output Octets
CC-Service-Specific-Units	Optional	Comverse ONE solution unit type other than CURRENCY, SECONDS or OCTETS
Validity-Time	Optional	Time duration in seconds that the reservation is valid.
Final-Unit-Indication	Optional	Populated when a partial reservation was done or the Comverse ONE solution Rating Database was down.

NOTE Only one of the child AVPs of Granted-ServiceUnit is returned in the CCA.

Authorization Immediate Event Message

This message shall be used by DCMP (Diameter client) to request for real subscriber's primary offer ID. For the Authorization CCR- Event request, a success response will only will be sent if subscriber is valid, active and its parent account is also active. In case of 1SA the subscriber needs to be active. The following diagram shall further illustrate the flow.

Figure 59 Authorization CCR-EVENT/CCA-EVENT Sequence



In this scenario, the Diameter client (DCMP) intends to request authorization using the CCR-Event request message. Once the CCR-message is received by Comverse ONE (OCS), the subscriber is validated and the primary offer is included in the CCA response message. The CCA message also indicates if the subscriber lookup was successful. Also note in this scenario that the Diameter session is closed after the CCA is transmitted by the OCS.

The authorization CCR-Event message contains the following details.

Table 67 Authorization CCR-Event

AVP	Type	Description
Session-Id	M	Session ID
Origin-Host	M	Origin host
Origin-Realm	M	Origin realm
Destination-Realm	M	Destination realm
Auth-Application-Id	M	4
Service-Context-Id	M	Used to identify the service logic at the DCCS.
CC-Request-Type	**M**	**EVENT_REQUEST**
CC-Request-Number	M	Request number
Requested-Action	**M**	**Price_Enquiry**
Subscription-Id: Subscription-Id-Type	M	Uses the values MSISDN or IMSI.
Subscription-Id: Subscription-Id-Data	M	UTF8String
Requested-Service-Unit	O	Possible member AVPs are listed in the DCI ICD.
Event-Timestamp	O	Timestamp of the charge event.
Service-Parameter-Info: Client Account ID	O	Identifier of the Diameter client.
Service-Parameter-Info: Service Type	O	Service type
Service-Parameter-Info: Service Subtype	O	Service subtype
Service-Parameter-Info: Quality of Service	O	Quality of Service
Service-Parameter-Info: Timezone	O	Timezone
Service-Parameter-Info: Comment	O	Comment
Service-Parameter-Info: Origination Location Type	O	See description in the DCI ICD.
Service-Parameter-Info: Origination Location	O	See description in the DCI ICD.
Service-Parameter-Info: Destination Location Type	O	See description in the DCI ICD.
Service-Parameter-Info: Destination Location	O	See description in the DCI ICD.
Service-Parameter-Info: Special Feature Digits	O	Prefix for a special feature invocation such as CUG or VPN.
Service-Parameter-Info: Service Mode	O	Defines originating or terminating call type. Possible values are listed in the DCI ICD
Service-Parameter-Info: Authorization	**M**	**Defines the authorization request to OCS** **This tag must be present when DCMP is using Price_Enquiry.**
Service-Parameter-Info: Custom1	O	To be decided for a specific Customer/Application implementation.
Service-Parameter-Info: Custom2	O	To be decided for a specific Customer/Application implementation.

Table 67 Authorization CCR-Event

AVP	Type	Description
Service-Parameter-Info: Custom3	O	To be decided for a specific Customer/Application implementation.
Service-Parameter-Info: Custom4	O	To be decided for a specific Customer/Application implementation.
Service-Parameter-Info: Custom5	O	To be decided for a specific Customer/Application implementation.

The authorization CCA-Event response message contains the following details.

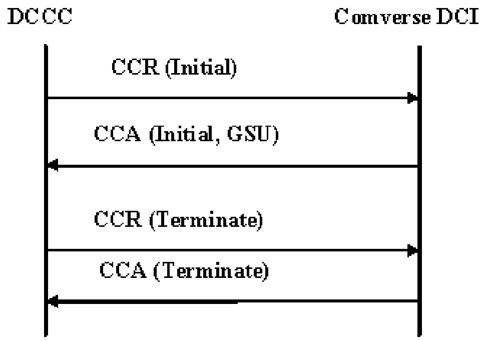
Table 68 Authorization CCA-Event

AVP	Type	Description
Session-Id	M	Session ID
Result-Code	M	Result of the operation
Origin-Host	M	Origin host
Origin-Realm	M	Origin realm
Auth-Application-Id	M	4
CC-Request-Type	M	EVENT_REQUEST
CC-Request-Number	M	From the request
Origin-State-Id	M	Origin state ID
Event-Timestamp	M	Timestamp of the request event
CMVT-Primary-Offer	O	**Returns real subscriber’s Primary Offer. If the SDP is down, it returns the default Primary Offer as defined in the system.**

Event with Reservation

The Event with Reservation scenario, shown in Figure 60, “Event with Reservation,” is characterized by an initial reservation with a CCR (Initial) followed by a debit with the CCR Terminate. Functionally, there is only a slight difference between this scenario and the Session reservation scenario without an update. This scenario is typically used when the consumed amount is known at the beginning of the session and the subscriber must be authenticated before the service is actually delivered (for example, downloading an application or ring tone).

Figure 60 Event with Reservation



Initial Exchange

Table 69 Credit Control Request – Initial Exchange – Event with Reservation

AVP	Type	Description
Session-Id	Mandatory	Session ID
Origin-Host	Mandatory	Origin Host
Origin-Realm	Mandatory	Origin Realm
Destination-Realm	Mandatory	Destination Realm
Auth-Application-Id	Mandatory	4
Service-Context-Id	Mandatory	Identifies the service logic at the DCI
CC-Request-Type	Mandatory	INITIAL_REQUEST
CC-Request-Number	Mandatory	Request Number
Subscription-Id: Subscription-Id-Type	Mandatory	Uses the values MSISDN or IMSI. Refer to Table 60, "Subscriber ID Type (from RFC 4006)"
Subscription-Id: Subscription-Id-Data	Mandatory	UTF8String
Event-Timestamp	Optional	Timestamp of the charge event
Service-Parameter-Info: Client Account ID	Optional	Identifies the Diameter client
Service-Parameter-Info: Service type	Optional	Service type
Service-Parameter-Info: Service subtype	Optional	Service Sub Type
Service-Parameter-Info: Quality of Service	Optional	Quality of Service
Service-Parameter-Info: Time zone	Optional	Time zone
Service-Parameter-Info: Comment	Optional	The value in this AVP is transferred to the CDR.
Service-Parameter-Info: Origination Location Type	Optional	Refer to section "Location Parameters," on page 147.
Service-Parameter-Info: Origination Location	Optional	Refer to section "Location Parameters," on page 147.
Service-Parameter-Info: Destination Location Type	Optional	Refer to section "Location Parameters," on page 147.
Service-Parameter-Info: Destination Location	Optional	Refer to section "Location Parameters," on page 147.
Service-Parameter-Info: Special Feature Digits	Optional	Prefix for a special feature invocation. For example, Calling Circle, Friends and Family.

Table 69 Credit Control Request – Initial Exchange – Event with Reservation (Continued)

AVP	Type	Description
Service-Parameter-Info: Service Mode	Optional	Defines originating or terminating call type. Possible values are listed in Table 59, “Tele-Service-Modes.”.
Service-Parameter-Info: Custom1	Optional	
Service-Parameter-Info: Custom2	Optional	
Service-Parameter-Info: Custom3	Optional	
Service-Parameter-Info: Custom4	Optional	
Service-Parameter-Info: Custom5	Optional	

Table 70 Credit Control Answer – Initial Exchange – Event with Reservation

AVP	M/O	Description
Session-Id	Mandatory	Session ID
Result-Code	Mandatory	Result of the operation
Origin-Host	Mandatory	Origin Host
Origin-Realm	Mandatory	Origin Realm
Auth-Application-Id	Mandatory	4
CC-Request-Type	Mandatory	INITIAL_REQUEST
CC-Request-Number	Mandatory	From initial request passed through DCI
Origin-State-Id	Mandatory	DMH fills this automatically
Event-Timestamp	Mandatory	Timestamp of the charge event
Granted-Service-Unit	Mandatory	Member AVP based on the rules outlined in Table 63, “Answer Unit Type Mapping”
CC-Time	Optional	Time
CC-Money	Optional	Money
CC-Total-Octets	Optional	Total Octets
CC-Service-Specific-Units	Optional	Comverse ONE solution unit type other than CURRENCY, SECONDS or OCTETS
Validity-Time	Optional	Time duration, in seconds, for which the reservation is valid.
Final-Unit-Indication	Optional	Populated when a partial reservation was done or when the rating database was down.

NOTE Only one of the children AVPs of Granted Service Unit is returned in the CCA.

Terminate Exchange

Table 71 Credit Control Request – Terminate Exchange – Event with Reservation

AVP	Type	Description
Session-Id	Mandatory	Session ID
Origin-Host	Mandatory	Origin Host
Origin-Realm	Mandatory	Origin Realm
Destination-Host	Mandatory	Destination Host The value of Destination-Host comes from the Origin-Host AVP in the CCR-Initial.
Destination-Realm	Mandatory	Destination Realm
Auth-Application-Id	Mandatory	4
Service-Context-Id	Mandatory	Identifies the service logic at the DCI
CC-Request-Type	Mandatory	TERMINATION_REQUEST
CC-Request-Number	Mandatory	Request Number
Used-Service-Unit	Mandatory	Number of units that were used
Event-Timestamp	Optional	Timestamp of the charge event

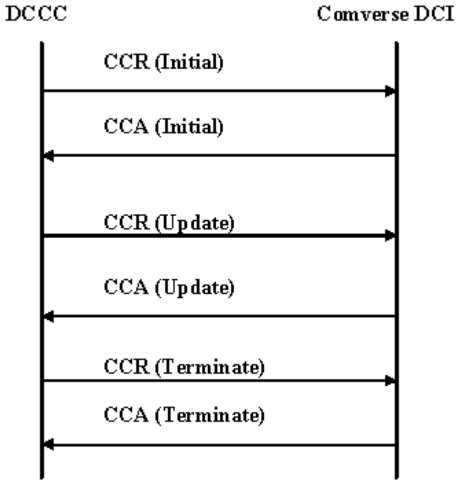
Table 72 Credit Control Answer – Terminate Exchange – Event with Reservation

AVP	Type	Description
Session-Id	Mandatory	Session ID
Result-Code	Mandatory	Result of the operation
Origin-Host	Mandatory	Origin Host
Origin-Realm	Mandatory	Origin Realm
Auth-Application-Id	Mandatory	4
CC-Request-Type	Mandatory	TERMINATION_REQUEST
CC-Request-Number	Mandatory	From the request
Origin-State-Id	Mandatory	Origin State ID
Event-Timestamp	Mandatory	Timestamp of the charge event

Session-Based Charge

This scenario is used when the consumed amount is not predetermined, such as in a voice call. The nature of the exchange is that the subscriber is authenticated and some initial value is reserved. After some period of time, the DCCC notifies the DCI that some consumption has occurred and additional balance is reserved. At the end of the session, the final consumption amount is reported.

Figure 61 Session-Based Charge



NOTE The CCR/CCA exchange for Update can be repeated as often as necessary to continue the session.

Initial Exchange

Table 73 Credit Control Request – Initial Exchange – Session-Based Charge

AVP	Type	Description
Session-Id	Mandatory	Session ID
Origin-Host	Mandatory	Origin Host
Origin-Realm	Mandatory	Origin Realm
Destination-Realm	Mandatory	Destination Realm
Auth-Application-Id	Mandatory	4
Service-Context-Id	Mandatory	Identifies the service logic at the DCI
CC-Request-Type	Mandatory	INITIAL_REQUEST
CC-Request-Number	Mandatory	Request Number
Subscription-Id: Subscription-Id-Type	Mandatory	Uses the values MSISDN or IMSI. Refer to Table 60, “Subscriber ID Type (from RFC 4006).”.
Subscription-Id: Subscription-Id-Data	Mandatory	UTF8String
Requested-Service-Unit	Optional	Requested Service Unit. The possible member AVPs are listed in Table 62, “Requested-Service-Unit Member AVP to Comverse ONE Solution Unit Type Mapping.”
Event-Timestamp	Optional	Timestamp of the charge event
Service-Parameter-Info: Client Account ID	Optional	Identifies the Diameter client
Service-Parameter-Info: Service type	Optional	Service type
Service-Parameter-Info: Service subtype	Optional	Service Sub Type
Service-Parameter-Info: Quality of Service	Optional	Quality of Service

Table 73 Credit Control Request – Initial Exchange – Session-Based Charge (Continued)

AVP	Type	Description
Service-Parameter-Info: Time zone	Optional	Time zone
Service-Parameter-Info: Comment	Optional	The value in this AVP is transferred to the CDR.
Service-Parameter-Info: Origination Location Type	Optional	Refer to section "Location Parameters," on page 147.
Service-Parameter-Info: Origination Location	Optional	Refer to section "Location Parameters," on page 147.
Service-Parameter-Info: Destination Location Type	Optional	Refer to section "Location Parameters," on page 147.
Service-Parameter-Info: Destination Location	Optional	Refer to section "Location Parameters," on page 147.
Service-Parameter-Info: Special Feature Digits	Optional	Prefix for a special feature invocation. For example, Calling Circle, Friends and Family.
Service-Parameter-Info: Service Mode	Optional	Defines originating or terminating call type. Possible values are listed in Table 59, "Tele-Service-Modes."
Service-Parameter-Info: Custom1	Optional	TBD for a specific Customer/ Application implementation
Service-Parameter-Info: Custom2	Optional	TBD for a specific Customer/ Application implementation
Service-Parameter-Info: Custom3	Optional	TBD for a specific Customer/ Application implementation
Service-Parameter-Info: Custom4	Optional	TBD for a specific Customer/ Application implementation
Service-Parameter-Info: Custom5	Optional	TBD for a specific Customer/ Application implementation

Table 74 Credit Control Answer – Initial Exchange – Session-Based Charge

AVP	M/O	Description
Session-Id	Mandatory	Session ID
Result-Code	Mandatory	Result of the operation
Origin-Host	Mandatory	Origin Host
Origin-Realm	Mandatory	Origin Realm
Auth-Application-Id	Mandatory	4
CC-Request-Type	Mandatory	INITIAL_REQUEST
CC-Request-Number	Mandatory	From initial request passed through DCI
Origin-State-Id	Mandatory	Origin State ID
Event-Timestamp	Mandatory	Timestamp of the charge event
Granted-Service-Unit	Mandatory	Number of units that were granted
CC-Time	Optional	Time
CC-Money	Optional	Money
CC-Total-Octets	Optional	Total Octets
CC-Input-Octets	Optional	Input Octets
CC-Output-Octets	Optional	Output Octets

Table 74 Credit Control Answer – Initial Exchange – Session-Based Charge (Continued)

AVP	M/O	Description
CC-Service-Specific-Units	Optional	Comverse ONE solution unit type other than CURRENCY, SECONDS or OCTETS
Validity-Time	Optional	Time duration, in seconds, for which the reservation is valid.
Final-Unit-Indication	Optional	Populated when a partial reservation was done or when the rating database was down.

NOTE Only one of the children AVPs of Granted Service Unit is returned in the CCA.

Update Exchange

Table 75 Credit Control Request – Update Exchange – Session-Based Charge

AVP	Type	Description
Session-Id	Mandatory	Session ID
Origin-Host	Mandatory	Origin Host
Origin-Realm	Mandatory	Origin Realm
Destination-Host	Mandatory	Destination Host The value of Destination-Host comes from the Origin-Host AVP in the last CCA.
Destination-Realm	Mandatory	Destination Realm
Auth-Application-Id	Mandatory	4
Service-Context-Id	Mandatory	Identifies the service logic at the DCI
CC-Request-Type	Mandatory	UPDATE_REQUEST
CC-Request-Number	Mandatory	Request Number
Used-Service-Unit	Mandatory	Number of units that were used
Requested-Service-Unit	Mandatory	Requested Service Unit. The possible member AVPs are listed in Table 62, “Requested-Service-Unit Member AVP to Comverse ONE Solution Unit Type Mapping.”
Event-Timestamp	Optional	Timestamp of the charge event

Table 76 Credit Control Answer – Update Exchange – Session-Based Charge

AVP	Type	Description
Session-Id	Mandatory	Session ID
Result-Code	Mandatory	Result of the operation
Origin-Host	Mandatory	Origin Host
Origin-Realm	Mandatory	Origin Realm
Auth-Application-Id	Mandatory	4
CC-Request-Type	Mandatory	UPDATE_REQUEST
CC-Request-Number	Mandatory	Request Number
Event-Timestamp	Optional	Timestamp of the charge event
Origin-State-Id	Mandatory	Origin State ID
Event-Timestamp	Mandatory	Timestamp of the charge event
Granted-Service-Unit	Mandatory	Number of units that were granted
CC-Time	Optional	Time
CC-Money	Optional	Money
CC-Total-Octets	Optional	Total-Octets
CC-Input-Octets	Optional	Input-Octets
CC-Output-Octets	Optional	Output-Octets
CC-Service-Specific-Units	Optional	A unit type other than time, money, or octets.
Validity-Time	Optional	Time duration, in seconds, for which the reservation is valid.
Final-Unit-Indication	Optional	Populated when a partial reservation was done or when the rating database was down.

NOTE Only one of the child AVPs of Granted Service Unit is returned in the CCA.

Terminate Exchange

Table 77 Credit Control Request – Terminate Exchange – Session-Based Charge

AVP	Type	Description
Session-Id	Mandatory	Session ID
Origin-Host	Mandatory	Origin Host
Origin-Realm	Mandatory	Origin Realm
Destination-Host	Mandatory	Destination Host The value of Destination-Host comes from the Origin-Host AVP in the last CCA.
Destination-Realm	Mandatory	Destination Realm
Auth-Application-Id	Mandatory	4
Service-Context-Id	Mandatory	Identifies the service logic at the DCI
CC-Request-Type	Mandatory	TERMINATION_REQUEST

Table 77 Credit Control Request – Terminate Exchange – Session-Based Charge (Continued)

AVP	Type	Description
CC-Request-Number	Mandatory	Request Number
Used-Service-Unit	Mandatory	Number of units that were used
Event-Timestamp	Optional	Timestamp of the charge event

Table 78 Credit Control Answer – Terminate Exchange – Session-Based Charge

AVP	Type	Description
Session-Id	Mandatory	Session ID
Result-Code	Mandatory	Result of the operation
Origin-Host	Mandatory	Origin Host
Origin-Realm	Mandatory	Origin Realm
Auth-Application-Id	Mandatory	4
Service-Context-Id	Mandatory	Identifies the service logic at the DCI
CC-Request-Type	Mandatory	TERMINATION_REQUEST
CC-Request-Number	Mandatory	From the request
Origin-State-Id	Mandatory	Origin State ID
Event-Timestamp	Optional	Timestamp of the charge event

Reverse Transaction Credit

A Diameter client sends a request to reverse a previously charged transaction by sending the session ID of the past transaction in an Event Request. The past transaction could have been an event- or session-based charge. An Event Request with a Requested-Action AVP of 1 (Refund Account) is sent with the Session-ID of the past transaction in the Service-Parameter-Info AVP. Figure 62, “Reverse Transaction Credit” shows the exchange of messages for this scenario.

NOTE The reverse transaction can only occur for the duration of the history database.

Figure 62 Reverse Transaction Credit

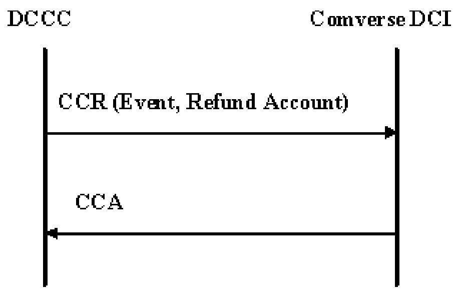


Table 79 Credit Control Request – Reverse Transaction Credit

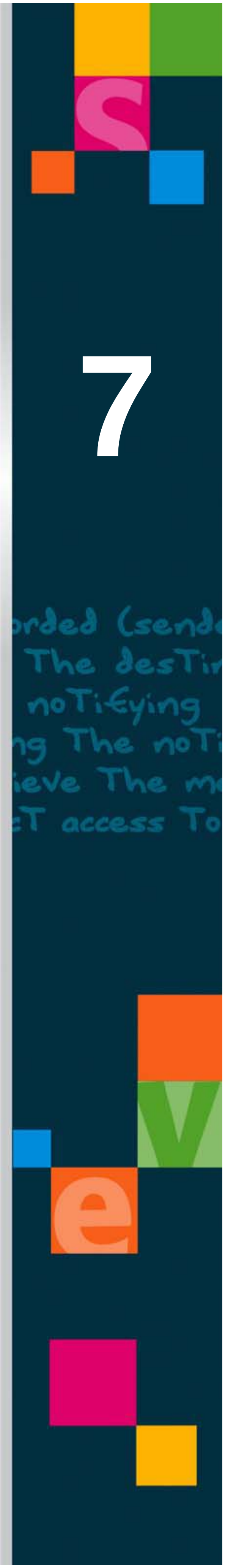
AVP	Type	Description
Session-Id	Mandatory	Session ID
Origin-Host	Mandatory	Origin Host
Origin-Realm	Mandatory	Origin Realm
Destination-Realm	Mandatory	Destination Realm
Auth-Application-Id	Mandatory	4
Service-Context-Id	Mandatory	Identifies the service logic at the DCI
CC-Request-Type	Mandatory	EVENT_REQUEST
CC-Request-Number	Mandatory	Request Number
Requested-Action	Mandatory	REFUND_ACCOUNT
Subscription-Id: Subscription-Id-Type	Mandatory	Uses the values MSISDN or IMSI. Refer to Table 60, "Subscriber ID Type (from RFC 4006)."
Subscription-Id: Subscription-Id-Data	Mandatory	UTF8String
Event-Timestamp	Optional	Timestamp of the charge event
Service-Parameter-Info: Time zone	Optional	Time zone
Service-Parameter-Info: Reverse Transaction Session ID	Mandatory	Session ID of the previous transaction

Table 80 Credit Control Answer – Reverse Transaction Credit

AVP	Type	Description
Session-Id	Mandatory	Session ID
Result-Code	Mandatory	Result of the operation
Origin-Host	Mandatory	Origin Host
Origin-Realm	Mandatory	Origin Realm
Auth-Application-Id	Mandatory	4
CC-Request-Type	Mandatory	EVENT_REQUEST
CC-Request-Number	Mandatory	From the request
Origin-State-Id	Mandatory	Origin State ID
Event-Timestamp	Mandatory	Timestamp of the charge event

Limitations

- Multiple-Services-Credit-Control, as defined in RFC 4006, is not supported by the Comverse DCI.
- Advice of Charge (Price Inquiry Request) and Balance Check, as defined in RFC 4006, are not supported.
- Tariff switching, as defined in RFC 4006, is not supported.



Chapter 7

Comverse Packet-Switched Diameter Charging Interface

Overview

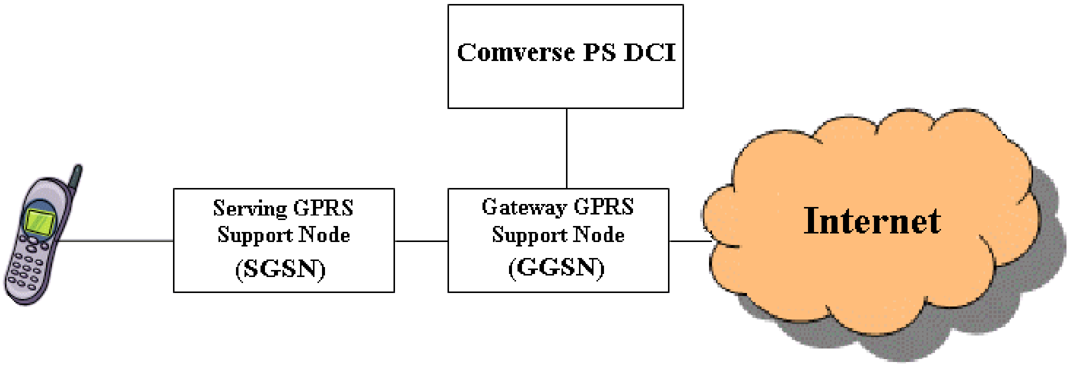
This chapter describes the Comverse Packet-Switched Diameter Charging Interface (PS DCI). The Comverse PS DCI is an implementation of the Packet-Switched Domain Charging described in the 3GPP specification TS 32.251. There are no Comverse extensions.

The PS DCI provides the rating parameters required by Comverse ONE solution for Session-based Rating and Charging. This interface and logic enables Comverse ONE solution to act as a Diameter Credit Control Server and provide most of the existing Comverse ONE solution rating and charging capabilities to applications acting as Diameter Credit Control Clients (DCCCs).

Comverse PS DCI is used for flow-based rating and charging activities within a data session in a mobile network. Comverse PS DCI supports General Packet Radio Service (GPRS) sessions. It is possible to use the Comverse PS DCI in other data networks if the client supports this interface.

Figure 63, "High-Level View of the Network from the User's Perspective" illustrates the Comverse PS DCI in a GPRS network from the user's perspective.

Figure 63 High-Level View of the Network from the User's Perspective


Comverse PS DCI
Serving GPRS Support Node (SGSN)
Gateway GPRS Support Node (GGSN)
Internet

Flow-Based Charging

The Comverse Packet-Switched Diameter Charging Interface (PS DCI) allows Diameter Credit Control Client (DCCC) applications to invoke existing Comverse ONE solution Rating and Charging capabilities to support flow-based charging of a GPRS session.

Flow-based charging is an application in GPRS in which multiple charging activities, or flows, comprise a session. In Flow-Based Charging, the Multiple-Services-Credit-Control (MSCC) AVP is used to support independent credit-control of multiple services in a single credit-control session. Within the MSCC AVP, the combination of the Rating-Group AVP and Service-ID AVP uniquely identify a service, or flow. It is possible that one flow comprises more than one unit type (for example, Input-Octets and Output-Octets).

When the user turns on his handset and connects to the Internet, a PDP Context is established, and the Comverse PS DCI receives a CCR-Initial message.

A Traffic Plane Function (TPF), which can reside in a GPRS Gateway Support Node (GGSN) or a Deep Packet Inspector (DPI), identifies the service flows within a data session and invokes real time charging for each flow. Though these flows are identified by such criteria as source IP address, destination IP address, source port number, destination port number, protocol ID of the protocol above IP, and so on, the TPF client categorizes them into Rating Group and Service Identifier. A Rating-Group is a set of services, identified by a Service-Identifier. Multiple Service-Identifiers can belong to the same Rating-Group. The DCCC identifies individual flows within a GPRS session via the Service-Identifier AVP and Rating-Group AVP. Examples of different flows are video streaming, FTP access, and particular URL access.

It is important to understand the difference between PDP context-level charging versus flow-based charging. A GGSN with a TPF can categorize different service flows occurring within a PDP context. This TPF can initiate credit control requests for all or few of these service flows and manage the allocated quotas independently. However, the TPF can also combine some of these flows and present them as one service to the PS DCI.

This interface can be used for charging even when the client is not using flow-based charging. In that case, Access Point Name (APN) becomes the key rating factor.

NOTE Flow-based charging is defined in 3GPP TS 23.125.

Authorization-only

If authorization-only is enabled, the PS DCI only performs authorization and does not grant any quota upon receiving the CCR-Initial. If authorization-only is disabled, the PS DCI performs authorization and grants quota upon receiving the CCR-Initial.

Starting a Flow

The DCCC starts a new flow by sending a CCR-Initial or CCR-Update message which contains an MSCC AVP containing a Requested-Service-Unit AVP. If TPF is enabled in the client, when the client identifies a specific service flow, it sends a CCR-Update with an MSCC corresponding to the service flow. If no previous credit control relationship exists, this service flow results in a CCR-Initial message.

The granting of the initial quota is the start of the first flow. Thereafter, the DCC can start additional flows. The PS DCI rates and charges each new flow separately. Each MSCC AVP with a unique combination of Rating-Group and Service-ID represents one flow. Thus, a message containing two MSCC AVPs represents two flows.

Stopping a Flow

During a session, a flow is stopped by the DCCC sending a CCR-Update message which contains an MSCC AVP containing no Requested-Service-Unit AVP. A CCR-Terminate message stops all currently open flows.

Message Flow Description

The message flow example shown in Figure 64, “Example of Independent Credit Control of Multiple Services in a Credit Control Session (Part 1)” and Figure 65, “Example of Independent Credit Control of Multiple Services in a Credit Control Session (Part 2)” illustrates a usage scenario in which the Diameter Credit Control Client (DCCC) and Comverse PS DCI support independent credit-control of multiple services.

Figure 64 Example of Independent Credit Control of Multiple Services in a Credit Control Session (Part 1)

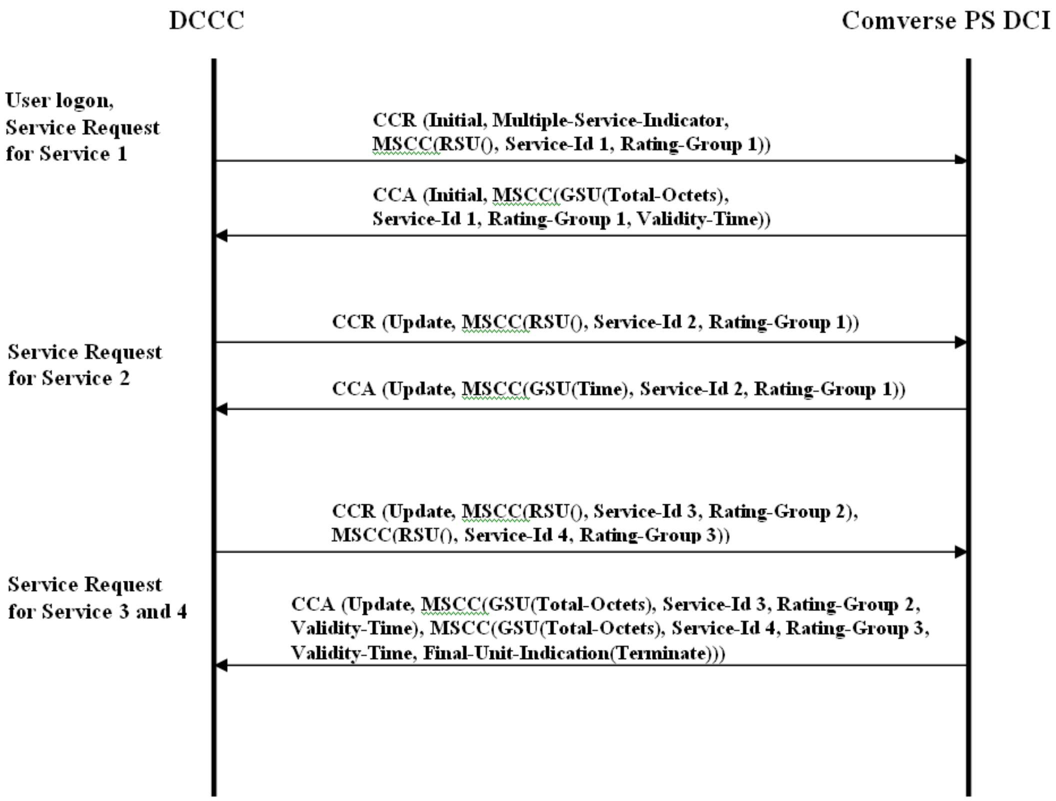
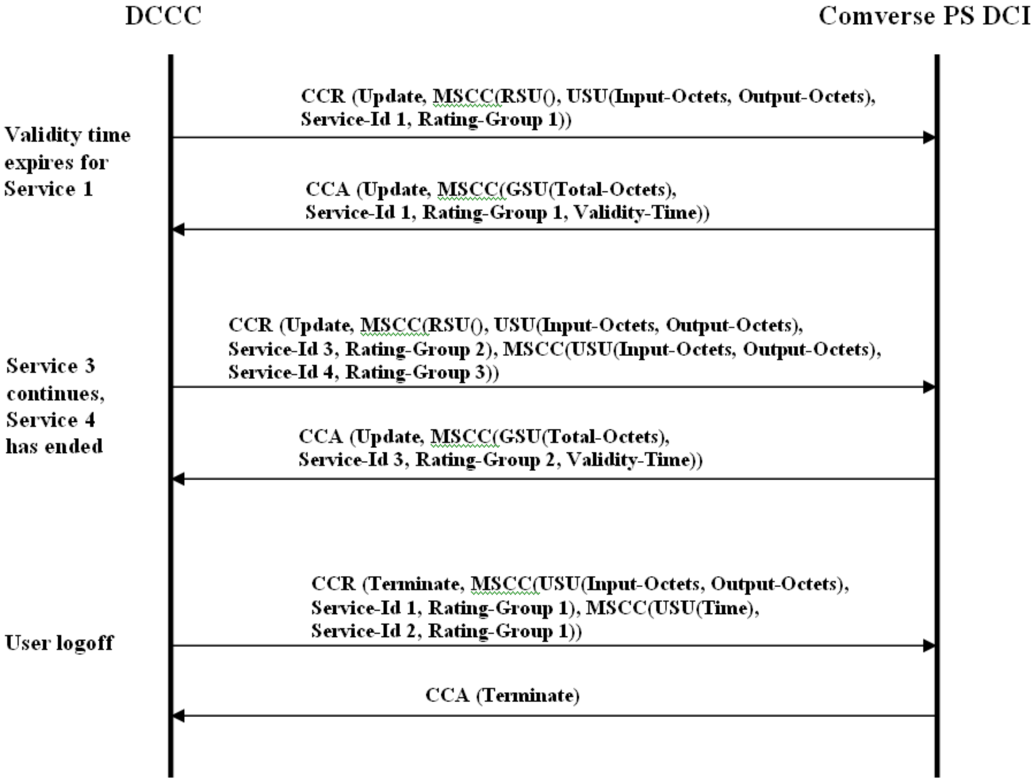

DCCC
Comverse PS DCI
User logon, Service Request for Service 1
CCR (Initial, Multiple-Service-Indicator, MSCC(RSU(), Service-Id 1, Rating-Group 1))
CCA (Initial, MSCC(GSU(Total-Octets), Service-Id 1, Rating-Group 1, Validity-Time))
CCR (Update, MSCC(RSU(), Service-Id 2, Rating-Group 1))
Service Request for Service 2
CCA (Update, MSCC(GSU(Time), Service-Id 2, Rating-Group 1))
CCR (Update, MSCC(RSU(), Service-Id 3, Rating-Group 2), MSCC(RSU(), Service-Id 4, Rating-Group 3))
Service Request for Service 3 and 4
CCA (Update, MSCC(GSU(Total-Octets), Service-Id 3, Rating-Group 2, Validity-Time), MSCC(GSU(Total-Octets), Service-Id 4, Rating-Group 3, Validity-Time, Final-Unit-Indication(Terminate)))

Figure 65 Example of Independent Credit Control of Multiple Services in a Credit Control Session (Part 2)


DCCC
Comverse PS DCI
Validity time expires for Service 1
CCR (Update, MSCC(RSU(), USU(Input-Octets, Output-Octets), Service-Id 1, Rating-Group 1))
CCA (Update, MSCC(GSU(Total-Octets), Service-Id 1, Rating-Group 1, Validity-Time))
Service 3 continues, Service 4 has ended
CCR (Update, MSCC(RSU(), USU(Input-Octets, Output-Octets), Service-Id 3, Rating-Group 2), MSCC(USU(Input-Octets, Output-Octets), Service-Id 4, Rating-Group 3))
CCA (Update, MSCC(GSU(Total-Octets), Service-Id 3, Rating-Group 2, Validity-Time))
User logoff
CCR (Terminate, MSCC(USU(Input-Octets, Output-Octets), Service-Id 1, Rating-Group 1), MSCC(USU(Time), Service-Id 2, Rating-Group 1))
CCA (Terminate)

The following sections describe the message flow.

User Logon (Service Request for Service 1)

The user connects to the Internet with his handset. The Diameter Credit Control Client sends a CCR-Initial to the PS DCI to perform credit authorization for the bearer service (for example, Internet access service) and to establish a credit-control session. In this message, the DCCC includes the Multiple-Services-Indicator AVP to indicate support for independent credit-control of multiple services within the session.

The PS DCI checks the end user’s account balance. The PS DCI uses rating information (Service-ID and Rating-Group) received from the DCCC to rate the request.

Next, the PS DCI reserves credit from the user’s account. Then the PS DCI sends a CCA message to the DCCC. The CCA message includes an MSCC AVP that contains a Granted-Service-Units AVP containing CC-Total-Octets.

Service Request for Service 2

The end user uses Service 2. The DCCC sends a CCR-Update to the PS DCI to perform credit authorization for Service 2. The CCR-Update message includes the MSCC AVP to request service units for Service 2 that belong to Rating Group 1.

The PS DCI rates the request according to the Service-ID/Rating Group and updates the existing reservation by requesting more credit. The PS DCI then sends a CCA message to the DCCC. The CCA includes an MSCC AVP that contains a Granted-Service-Units AVP containing CC-Time.

Service Request for Service 3 and Service 4

The user now requests Services 3 and 4 as well, which are not authorized yet. The DCCC sends a CCR-Update to the PS DCI to perform credit authorization for services 3 and 4. This message includes two instances of the MSCC AVP to request service units for Service 3 that belongs to Rating-Group 2 and for Service 4 that belongs to Rating-Group 3.

The PS DCI checks the end user’s account balance and, according to the Service-IDs/Rating-Groups information, rates the request. The PS DCI then authorizes Services 3 and 4, then reserves credit.

Next, the PS DCI sends a CCA message to the DCCC. The CCA includes two instances of the MSCC AVP. Each instance of the MSCC contains a Granted-Service-Units AVP containing CC-Total-Octets. The PS DCI determines that Service 4 is not allowed to continue after these units are consumed. Therefore the Final-Unit-Indication AVP with action TERMINATE is associated to the Service-ID 4.

Validity Time Expires for Service 1

Next, the Validity-Time for Service 1 expires. The DCCC sends a CCR-Update message to the PS DCI to perform credit reauthorization for Service 1. This message carries one instance of the MSCC AVP that includes the units used by Service 1.

The PS DCI then deducts the appropriate amount from the user’s account and updates the reservation by requesting more credit. The PS DCI then sends a CCA message to the DCCC. The CCA includes an MSCC AVP that contains a Granted-Service-Units AVP containing CC-Total-Octets.

Service Request for Service 3, Service 4 has Ended

Next, the DCCC determines that Service 4 has ended, and Service 3 must continue. The DCCC sends a CCR-Update to PS DCI to perform credit reauthorization for Service 3 and to end the charging for Service 4.

The CCR message contains two instances of the MSCC AVP to report the units used by Services 3 and 4. The MSCC AVP for Service 3 contains an RSU AVP to indicate that the DCCC is requesting more usage. The MSCC AVP for Service 4 contains no RSU AVP, which indicates that the flow has ended.

User Logoff

Next, the end user logs off from the network. The DCCC sends a CCR-Terminate to PS DCI to debit the used units from the end user’s account and to stop the credit-control session. The CCR message contains the units consumed by each of the used services in multiple instances of the MSCC AVPs. The used units are associated with the relevant Service-Identifier and Rating-Group. The PS DCI debits the used units to the user's account and acknowledges the session termination by sending a CCA to the DCCC

PS DCI Basic Operation

PS DCI Message Details

3GPP DCCA has defined the following two messages to support real time charging capabilities,

- Credit Control Request (CCR) message is sent from DCCC to request charging and rating from PS DCI.
- Credit Control Answer (CCA) is sent as a response from PS DCI to the DCCC.

Credit Control Request

Table 81, “Credit-Control-Request AVP Components” shows the Attribute Value Pair (AVP) structure of the CCR message, as defined in 3GPP TS 32.299.

The CCR message header has the form:

```
<CCR> ::= < Diameter Header: 272, REQ, PXY >
```

<Session-Id> must be the first AVP after the Diameter header.

Table 81 Credit-Control-Request AVP Components

AVP Name	Rating and Charging AVP?	Type	Multiple Instances OK?	Supported by 3GPP?
Session-Id	YES	Mandatory	NO	YES
Origin-Host	NO	Mandatory	NO	YES
Origin-Realm	NO	Mandatory	NO	YES
Destination-Realm	NO	Mandatory	NO	YES
Auth-Application-Id	NO	Mandatory	NO	YES
Service-Context-Id	YES	Mandatory	NO	YES
CC-Request-Type	YES	Mandatory	NO	YES
CC-Request-Number	NO	Mandatory	NO	YES
Destination-Host	NO	Optional	NO	YES
User-Name	NO	Optional	NO	YES
CC-Sub-Session-Id	NO	Optional	NO	NO
Acct-Multi-Session-Id	NO	Optional	NO	NO
Origin-State-Id	NO	Optional	NO	YES
Event-Timestamp	YES	Optional	NO	YES
Subscription-Id	YES	Optional	YES	YES
Service-Identifier	NO	Optional	NO	NO
Termination-Cause	NO	Optional	NO	YES
Requested-Service-Unit	NO	Optional	NO	NO
Requested-Action	NO	Optional	NO	YES
Used-Service-Unit	NO	Optional	YES	YES
Multiple-Services-Indicator	NO	Optional	NO	YES
Multiple-Services-Credit-Control	YES	Optional	NO	YES
Service-Parameter-Info	NO	Optional	YES	NO
CC-Correlation-Id	NO	Optional	YES	NO
User-Equipment-Info	NO	Optional	NO	YES
Proxy-Info	NO	Optional	YES	YES
Route-Record	NO	Optional	YES	YES
Service-Information	YES	Optional	NO	YES
AVP	NO	Optional	YES	YES

The CCR message content is based on RFC 4006 definitions and is modified and enhanced by 3GPP as per their application needs.

The details of the AVPs in CCR message are defined in Diameter Base RFC 3588 and 3GPP TS 32.299. Some of the relevant AVPs specific to charging are detailed in the following sections.

Service-Context-Id AVP

The Service-Context-Id AVP determines the Charging Application. An appropriate value for this AVP must be determined for a particular implementation.

CC-Request-Type AVP

The CC-Request-Type AVP is of type Enumerated and contains the reason for sending the credit-control request message. It must be present in all Credit-Control-Request messages. The following values are defined for the CC-Request-Type AVP in 3GPP TS 32.299:

Table 82 CC-Request-Type AVP Fields

Enumerated Token	Value	Description
INITIAL_REQUEST	1	An Initial request initiates a credit-control session and contains credit control information that is relevant to the initiation.
UPDATE_REQUEST	2	An Update request contains credit-control information for an existing credit-control session. Update credit-control requests are sent every time a credit-control reauthorization is needed at the expiry of the allocated quota or validity time.
TERMINATION_REQUEST	3	An Update request contains credit-control information for an existing credit-control session. Update credit-control requests are sent every time a credit-control reauthorization is needed at the expiry of the allocated quota or validity time.Further, additional service-specific events can trigger a spontaneous Update request.
EVENT_REQUEST	4	This value is not supported for PS Charging. If a message is received with CC-Request-Type equal to EVENT_REQUEST, the message is rejected with a Result-Code of DIAMETER_UNABLE_TO_COMPLY.

Subscription-Id AVP

This AVP is of type Grouped and contains the Subscription-Id-Type AVP and the Subscription-Id-Data AVP. The Subscription-Id-Type AVP can contain one of the values in Table 83, “Subscription ID AVP Fields”.

Table 83 Subscription ID AVP Fields

Enumerated Token	Value	Description
END_USER_E164	0	Subscriber ID is of type E.164
END_USER_IMSI	1	Subscriber ID is based on E.212 number (International Mobile Subscriber ID)
END_USER_SIP_URI	2	Subscriber ID is of type SIP URI

Multiple-Services-Credit-Control AVP

The 3GPP Ro interface permits request and grant of service quota only at the Multiple-Services-Credit-Control (MSCC) level, not at the Credit Control Request (CCR) level. The CCR message must always contain at least one MSCC , even when the charging request is for only a single service. Each service is identified by one MSCC AVP.

Table 84, “Multiple-Services-Credit-Control AVP Components” shows the MSCC AVP as defined in 3GPP 32.299.

The MSCC header has the form:

```
<Multiple-Services-Credit-Control> ::=   < AVP Header: 456 >
```

Table 84 Multiple-Services-Credit-Control AVP Components

AVP Name	Supported by 3GPP?
Granted-Service-Unit	YES
Requested-Service-Unit	YES
Used-Service-Unit	YES
Tariff-Change-Usage	NO
Service-Identifier	YES
Rating-Group	YES
G-S-U-Pool-Reference	NO
Validity-Time	YES
Result-Code	YES
Final-Unit-Indication	YES
Time-Quota-Threshold	YES
Volume-Quota-Threshold	YES
Quota-Holding-Time	YES
Quota-Consumption-Time	YES
Reporting-Reason]	YES
Trigger-Type	YES
PS-Furnish-Charging-Information	YES
AVP	NO

The MSCC AVP is used in both the CCR and the CCA messages.

Two key AVPs inside the CCR's MSCC determine the rating and charging of an activity:

- Service-Identifier AVP
- Rating Group AVP

The Service-Identifier AVP identifies a service and the Rating Group AVP identifies the rating group consisting of multiple services. Thus multiple services can belong to the same Rating Group AVP. The PS DCI uses a combination of Rating Group and Service-Identifier to identify the Application and Sub Type. The Rating Group AVP and the Service-Identifier AVP are optional AVPs.

Requested-Service-Unit AVP

The Requested-Service-Unit AVP is of type Grouped and contains the amount of requested units specified by the Diameter credit-control client. The Requested-Service-Unit AVP can contain any combination of the following AVPs:

- CC-Time
- CC-Total-Octets
- CC-Input-Octets
- CC-Output-Octets
- CC-Service-Specific-Units

If the Requested-Service-Unit AVP is empty, the PS DCI determines the unit to be granted and calculates the rate. This scenario is called "Centralized Unit Determination and Centralized Rating."

If the Requested-Service-Unit AVP sent by the DCCC contains units, this scenario is called "Decentralized Unit Determination and Centralized Rating." The units are determined by the client and the rate is determined by the server.

A missing RSU is also a valid scenario. The DCCC denotes the end of a charging activity via a missing RSU.

Used-Service-Unit AVP

The Used-Service-Unit AVP is of type Grouped and contains the amount of used units measured from the point when the service became active or, if interim interrogations are used during the session, from the point when the previous measurement ended.

The Used-Service-Unit header has the form:

```
Used-Service-Unit ::= < AVP Header: 446 >
```

The Used-Service-Unit AVP can contain any combination of the following AVPs:

- Tariff-Change-Usage
- CC-Time
- CC-Total-Octets
- CC-Input-Octets
- CC-Output-Octets
- CC-Service-Specific-Units
- AVP

Service-Information AVP

The Service-Information AVP is of type Grouped. It contains the PS-Information AVP, which contains several AVPs important for PS Charging. Table 85, "PS-Information AVPs Important for PS Charging" describes these AVPs. All of these AVPs are optional.

Table 85 PS-Information AVPs Important for PS Charging

AVP	Description
Service-Information.PS-Information.3GPP-GPRS-Negotiated-QoS-Profile	The QOS profile applied by the GGSN. Refer to section "Diameter Request Handling," on page 179.
Service-Information.PS-Information.SGSN-Address	Holds the SGSN IP address used by the GTP control plane for the handling of control messages. Refer to section "Subscriber Location," on page 194.
Service-Information.PS-Information.Called-StationId	Contains the identifier of the access point name (APN) the user is connected to. Refer to section "Diameter Request Handling," on page 179.
Service-Information.PS-Information.3GPP-SGSN-MCC-MNC	Holds the MCC and MNC extracted from the RAI within the Create PDP Context Request or Update PDP Context Request message. Refer to section "Default Data," on page 194.
Service-Information.PS-Information.3GPP-MS-Timezone	Indicates the offset between universal time and local time in steps of 15 minutes of where the MS currently resides
Service-Information.PS-Information.3GPP-User-Location-Info	Indicates details of where the User Equipment is currently located (for example,. RAI, SAI, or CGI). Refer to section "Subscriber Location," on page 194.
Service-Information.PS-Information.3GPP-RAT-Type	Indicates which Radio Access Technology is currently serving the User Equipment. Refer to section "Diameter Request Handling," on page 179.

Credit Control Answer

Table 86, “Credit Control Answer AVP Components” shows the Attribute Value Pair (AVP) structure of the Credit Control Answer (CCA)message as defined in 3GPP TS 32.299.

The CCA message header has the form:

```
<CCA> ::=  < Diameter Header: 272, PXY >
```

<Session-Id> must be the first AVP after the Diameter header.

Table 86 Credit Control Answer AVP Components

AVP Name	Rating and Charging AVP?	Type	Multiple Instances OK?	Supported by 3GPP?
Session-Id >	NO	Mandatory	NO	YES
Result-Code }	NO	Mandatory	NO	YES
Origin-Host }	NO	Mandatory	NO	YES
Origin-Realm }	NO	Mandatory	NO	YES
Auth-Application-Id }	NO	Mandatory	NO	YES
CC-Request-Type }	NO	Mandatory	NO	YES
CC-Request-Number }	NO	Mandatory	NO	YES
User-Name	NO	Optional	NO	NO
CC-Session-Failover	NO	Optional	NO	YES
CC-Sub-Session-Id	NO	Optional	NO	NO
Acct-Multi-Session-Id	NO	Optional	NO	NO
Origin-State-Id	NO	Optional	NO	NO
Event-Timestamp	NO	Optional	NO	NO
Granted-Service-Unit	NO	Optional	NO	NO
Multiple-Services-Credit-Control	YES	Optional	YES	YES
Cost-Information	NO	Optional	NO	NO
Final-Unit-Indication	NO	Optional	NO	NO
Check-Balance-Result	NO	Optional	NO	NO
Credit-Control-Failure-Handling	NO	Optional	NO	YES
Direct-Debiting-Failure-Handling	NO	Optional	NO	NO
Validity-Time	NO	Optional	NO	NO
Redirect-Host	NO	Optional	YES	YES
Redirect-Host-Usage	NO	Optional	NO	YES
Redirect-Max-Cache-Time	NO	Optional	NO	YES
Proxy-Info	NO	Optional	YES	YES
Route-Record	NO	Optional	YES	YES
Failed-AVP	NO	Optional	YES	YES
Service-Information]	NO	Optional	NO	YES
AVP	NO	Optional	YES	YES

The CCA message content is based on RFC 4006 definitions and is modified/enhanced by 3GPP as per their application needs.

Configurable Parameters

The PS DCI has a configuration file that contains various configuration parameters. Each configurable parameter is described in the following sections. Refer to the section "Format of Configurable Parameters," on page 193 for the format of the configurable parameters.

Authorization-only

A configurable parameter specifies whether authorization-only is enabled. When this parameter is enabled, upon receipt of the CCR-Initial, the PS DCI performs authorization only and does not perform any rating or charging. The default setting of this parameter is disabled.

CDR Information

One configuration item is a parameter to specify what information, if any, to put into the CDR Information field in the CDR. Possible choices of information include the values from the following AVPs:

- One or more AVPs received inside the PS-Information AVP
- Multiple-Services-Credit-Control.Rating-Group AVP
- Multiple-Services-Credit-Control.Service-Identifier AVP

Any valid AVP name(s) can be specified as values for this configurable parameter. The default setting is for no AVPs to appear in the CDR Information field in the CDR.

Subscriber Location

When tariff determination is based on the location of the subscriber or destination, then location parameters must be extracted from the received Diameter message and transferred to the Comverse ONE solution. There are two pairs of parameters used for location, one for subscriber location and the other for the location of the destination or other location. The PS DCI uses default values for the other location.

One configurable item is a parameter to specify where to obtain the Subscriber-Location and Subscriber-Location-Type. Possible choices of this information are from the following AVPs:

- Service-Information.PS-Information.3GPP-User-Location-Info
- Service-Information.PS-Information.SGSN-Address
- Service-Information.PS-Information.3GPP-SGSN-MCC-MNC

A preference order is specified in the configurable parameter. For example, use the value from PS-Information.3GPP-User-Location-Info if that AVP is present. If the PS-Information.3GPP-User-Location-Info AVP is not present, use the value from PS-Information.SGSN-Address if that AVP is present. If the PS-Information.SGSN-Address AVP is not present, use the value from the PS-Information.3GPP-SGSN-MCC-MNC AVP. If none of the above three AVPs are present, the PS DCI does not provide location information to the PS DCI core application.

There is an optional parameter for the location-type. The possible choices for location-type are as follows:

- Handset
- Dialed-Number
- MSRN
- Cell-ID
- MSC-ID
- SGSN-ID
- IP-Address

For the 3GPP-User-Location-Info AVP, the default location-type is Cell-ID. For the SGSN-Address AVP, the default location-type is IP-Address. For the 3GPP-SGSN-MCC-MNC AVP, the default location-type is SGSN-ID.

If an element with the subscriber-location name does not appear in the configuration file, the PS DCI does not provide location information to the PS DCI application. If the charging activity depends on location data then the default location is used for the A party.

Default Data

There is a section in the PS DCI configuration file that specifies default data.

Some parameters are constant values that calculate values of specific AVPs. The value for Time-Quota-Threshold-Percentage is used in the calculation of the Multiple-Services-Credit-Control.Time-Quota-Threshold AVP. The value for Volume-Quota-Threshold-Percentage is used in the calculation of the Multiple-Services-Credit-Control.Volume-Quota-Threshold AVP. If the Percentage parameter is present in the configuration file, the corresponding Threshold AVP is present. If the Percentage parameter is not present in the configuration file, the corresponding Threshold AVP is not present

Other parameters specify the value of specific AVPs, including the MSCC.Quota-Holding-Time AVP, the MSCC.Quota-Consumption-Time AVP, the MSCC.Trigger-Type AVP, and the MSCC.Final-Unit-Indication.Final-Unit-Action AVP. If the parameter is present in the configuration file, that AVP is present in the CCA message. If the parameter is not present in the configuration file, that AVP is not present in the CCA message. The default behavior is for the AVPs to be absent in the CCA message.

The DCCC delivers the QOS information to the PS DCI as an information element, which is a UTF8String of hexadecimal digits. The QOS information element is described in 3GPP TS 29.061. A mechanism in the PS DCI translates the mean throughput to a string value. The string value in the configuration file corresponds to a string value provisioned in the database for the QOS parameter. The mean throughput value begins at the eighth digit in the UTF8String that is received from the DCCC. The mean throughput value is two digits long. The valid values for the integer value of QOS are 01–18, 30–31.

Table 87, “Example Mapping of Mean Throughput Value to String Value” shows an example mapping from mean throughput values to string values provisioned in the database. This is only an example. The string values provisioned in the database can be any string values.

Table 87 Example Mapping of Mean Throughput Value to String Value

Mean Throughput Value	Example string value provisioned in database
01	Low Speed
02	Low Speed
03	Low Speed
04	Low Speed
05	Low Speed
06	Low Speed
07	Medium Speed
08	Medium Speed
09	Medium Speed
10	Medium Speed
11	Medium Speed
12	Medium Speed
13	Medium Speed
14	High Speed
15	High Speed

Table 87 Example Mapping of Mean Throughput Value to String Value (Continued)

Mean Throughput Value	Example string value provisioned in database
16	High Speed
17	High Speed
18	High Speed
30	High Speed
31	High Speed

Diameter Request Handling

The Comverse ONE solution system contains a parameter called Application Usage Type (AUT). An AUT consists of Application, Sub Type, and Unit Type.

The most important action of Diameter request handling is the selection of the Application and Sub Type parameters used to initiate the billing process. This selection is based on a comparison of several AVPs to columns in the PS DCI Service Mapping Table. The result of the table lookup is the Application and Sub Type parameters from this table.

Application Sub Type Determination

Application and Sub Type are internal factors used in rating in the Comverse ONE solution. The Application and Sub Type are derived for each MSCC based on matching against the input values received. The PS DCI determines the Application and Sub Type using a table that contains these values, which can be provisioned in the Product Catalog. Table 88, “Rating Parameters Used in PS DCI” describes this table.

Table 88 Rating Parameters Used in PS DCI

Column	Type	Description
SCID	Mandatory	The Service-Context-ID AVP value of the CCR-Initial or CCR-Event message
APN	Optional	Service-Information.PS-Information:Called-Station-ID AVP
QOS	Optional	Service-Information.PS-Information:3GPP-GPRS-Negotiated-QoS-Profile AVP
RADIO_TYPE	Optional	Service-Information.PS-Information:RAT Type AVP
RATING_GROUP	Optional	MSCC: Rating-Group AVP
SERVICE_IDENTIFIER	Optional	MSCC: Service Identifier AVP

Unit Determination

In addition to selecting the Application and Sub Type, if the received message contains the Requested-Service-Unit AVP, the PS DCI must also inspect this AVP to determine how to perform unit determination. If the Requested-Service-Unit AVP is empty, the Comverse PS DCI reverts to Centralized Unit Determination to select the correct unit type. This is the most likely scenario. With Centralized Unit Determination, the Diameter Unit Mapping Table determines the unit type(s). One Application/Sub Type can map to multiple unit types. The Diameter Unit Mapping Table is used by all Diameter Charging Applications in Comverse ONE solution.

Table 89, “Requested-Service-Unit, Granted-Service-Unit, and Used-Service-Unit Member AVPs” identifies the possible member AVPs of the Requested-Service-Unit AVP. These member AVPs are also used within the Granted-Service-Unit AVP and Used-Service-Unit AVP.

Table 89 Requested-Service-Unit, Granted-Service-Unit, and Used-Service-Unit Member AVPs

AVP	Comverse ONE Solution Unit Type
CC-Time	SECONDS
CC-Total-Octets	OCTETS
CC-Input-Octets	INPUT_OCTETS
CC-Output-Octets	OUTPUT_OCTETS
CC-Service-Specific-Units	For this AVP, the Diameter Unit Mapping Table correlates the Application and Sub Type to the Comverse ONE solution unit type.

The following quota combinations allocated by Comverse ONE solution are possible for PS DCI:

- Time
- Service-Specific Units
- Total-Octets
- Input-Octets
- Output-Octets
- Input-Octets and Output-Octets
- Time and Total Octets
- Time and Input-Octets
- Time and Output-Octets
- Time, Input-Octets, and Output-Octets
- Time and Service-Specific Units

Diameter Answer Handling

All Diameter Requests sent to the PS DCI are acknowledged with a Diameter Answer, however receipt of a Diameter Answer does not constitute a successful operation. The Diameter Answer contains the Result-Code AVP that must be inspected by the DCCC to determine success of the message.

In addition to the Result-Code AVP at the command level, each Multiple-Services-Credit-Control AVP contains a Result-Code AVP. The Result-Code AVP within the MSCC indicates the success or failure of that particular service.

Any Result-Code that is not DIAMETER_SUCCESS or DIAMETER_LIMITED_SUCCESS indicates that there was an error in the operation. In addition, certain errors indicate that the reservation based session is now closed.

In general DIAMETER_SUCCESS indicates that the operation was completely successful and DIAMETER_LIMITED_SUCCESS indicates that some part of the operation did not succeed. For instance, if the subscriber has limited funds and the granted amount is only a portion of the requested amount DIAMETER_LIMITED_SUCCESS is returned. Regardless, the Granted-Service-Units AVP must be consulted in all cases to determine the allowed quota.

Table 90, “Result Codes” lists possible values for the Result Code AVP in the Answer message.

Table 90 Result Codes

Enumerated Token	Value	Description	Session Closed?
DIAMETER_SUCCESS	2001	The operation was completely successful.	No
DIAMETER_LIMITED_SUCCESS	2002	Some part of the operation did not succeed. For instance, if the subscriber has limited funds and the granted amount is only a portion of the requested amount.	No
DIAMETER_UNABLE_TO_DELIVER	3002	The PS DCI was unable to process the message. This can occur because there was a problem decoding the message or there is no connection to the peer host.	No
DIAMETER_REALM_NOT_SERVED	3003	The PS DCI received a message that did not contain an Origin-Host AVP.	No
DIAMETER_TOO_BUSY	3004	The PS DCI cannot process any more transactions because it has reached its maximum number of concurrent transactions.	No
DIAMETER_INVALID_AVP_BITS	3009	The Origin-Host AVP is malformed.	No
DIAMETER_UNKNOWN_PEER	3010	The configured peer host does not match the value in the Origin-Host AVP in the Capabilities Exchange Request or Capabilities Exchange Answer.	No
DIAMETER_AUTHENTICATION_ REJECTED	4001	The PS DCI determined that the user account is invalid.	No
DIAMETER_OUT_OF_SPACE	4002	The PS DCI could not allocate space for a session.	Yes
DIAMETER_END_USER_SERVICE_ DENIED	4010	The PS DCI determined that the service was denied. Possible reasons include: the activity is not allowed in the current subscriber state, cannot load account data, or an exception condition occurred while processing an event or session.	No
DIAMETER_CREDIT_LIMIT_ REACHED	4012	The reservation limit for the user was reached.	No
DIAMETER_UNKNOWN_SESSION_ ID	5002	The PS DCI received an unknown session ID. For example, it received a session ID in a CCR-Update or CCR-Terminate which does not match the session ID for any currently open sessions.	Yes
DIAMETER_INVALID_AVP_VALUE	5004	The PS DCI was unable to retrieve data from an AVP.	No
DIAMETER_MISSING_AVP	5005	A mandatory AVP is missing from the message.	No

Client Requirements for Sessions Using Multiple Transactions

When the charging scenario requires multiple Diameter transactions for the same session ID, there is one additional requirement for the DCCC application when interfacing to the Comverse PS DCI. This includes any scenario that starts with a CCR with the CC-Request-Type AVP as INITIAL_REQUEST and is followed by a CCR with the CC-Request-Type AVP set to UPDATE_REQUEST or TERMINATION_ REQUEST.

To insure proper operation, the client application must populate the Destination-Host AVP in subsequent CCR messages sent to the Comverse PS DCI for the same Session-ID. The value for the Destination-Host AVP is reported to the client application in Origin-Host AVP from the last CCA message received from the Comverse PS DCI for the corresponding Session-ID. The value of the Origin-Host AVP in the CCA identifies the subsystem that is handling the session. Providing Destination-Host AVP in the CCR for

UPDATE_REQUEST and TERMINATION_REQUEST insures that the correct subsystem is addressed for these subsequent requests.

Charging Scenarios

The Charging scenarios are specified in this section along with the appropriate mapping of the AVP to the appropriate Comverse ONE solution factor. The scenarios supported are one of the following:

- Event with Reservation
- Session Charge

Each scenario can have the following variations:

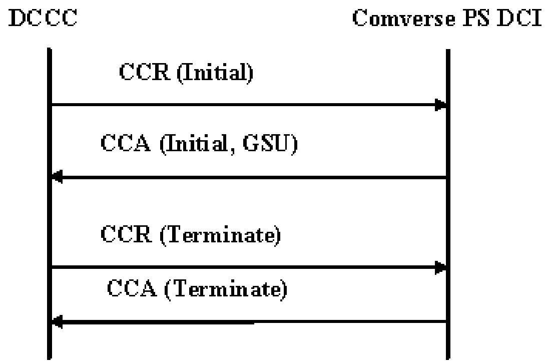
- Decentralized Unit determination/Centralized Rating
- Centralized Unit determination/Centralized Rating

The expected AVPs in the initial request are dependent on the charging variations. However, the rules are applied consistently across scenarios.

Event with Reservation Charge

The Event with Reservation scenario, shown in Figure 66, “Event with Reservation”, is characterized by an initial reservation with a CCR (Initial) followed by a debit with the CCR Terminate. Functionally, there is very little difference between this scenario and the Session reservation scenario without an update. This scenario is typically used when the consumed amount is known at the beginning of the session and there is a need to authenticate the subscriber before the service is actually delivered.

Figure 66 Event with Reservation



Initial Exchange

Table 91 Credit Control Request – Event with Reservation – Initial Exchange

AVP	Mandatory?	Description
Session-Id	Mandatory	Session ID
Origin-Host	Mandatory	Origin Host
Origin-Realm	Mandatory	Origin Realm
Destination-Realm	Mandatory	Destination Realm
Auth-Application-Id	Mandatory	4

Table 91 Credit Control Request – Event with Reservation – Initial Exchange (Continued)

AVP	Mandatory?	Description
Service-Context-Id	Mandatory	Used to identify the service logic at the PS DCI. Also, this is one of the parameters used to determine the Application and Sub Type.
CC-Request-Type	Mandatory	INITIAL_REQUEST
CC-Request-Number	Mandatory	Request Number
Subscription-Id: Subscription-Id-Type	Mandatory	Uses the values MSISDN, IMSI, or SIP URI. Refer to Table 83, "Subscription ID AVP Fields"
Subscription-Id: Subscription-Id-Data	Mandatory	UTF8String
Multiple-Services-Indicator	Mandatory	Indicates whether multiple services are supported or not. For PS Domain Charging, the value is MULTIPLE_SERVICES_SUPPORTED.
Multiple-Services-Credit-Control.Requested-Service-Unit.	Optional	Can be empty or can contain some combination of the AVPs listed in table Table 85, "PS-Information AVPs Important for PS Charging".
Multiple-Services-Credit-Control.Service-Identifier	Optional	Identifies the service. This is one of the parameters used to determine the Application and Sub Type.
Multiple-Services-Credit-Control.Rating-Group	Optional	Identifies the rating group consisting of multiple services. This is one of the parameters used to determine the Application and Sub Type.
Event-Timestamp	Optional	Timestamp of the charge event
Service-Information.PS-Information.3GPP-GPRS-Negotiated-QoS-Profile	Optional	The QOS profile applied by the GGSN. This is one of the parameters used to determine the Application and Sub Type.
Service-Information.PS-Information.SGSN-Address	Optional	Holds the SGSN IP address that is used by the GTP control plane for the handling of control messages.
Service-Information.PS-Information.Called-StationId.	Optional	Contains the identifier of the access point name (APN) the user is connected to. This is one of the parameters used to determine the Application and Sub Type.
Service-Information.PS-Information.3GPP-SGSN-MCC-MNC	Optional	Holds the MCC and MNC extracted from the RAI within the Create PDP Context Request or Update PDP Context Request message.
Service-Information.PS-Information.3GPP-MS-Timezone	Optional	Indicates the offset between universal time and local time in steps of 15 minutes of where the MS currently resides.
Service-Information.PS-Information.3GPP-User-Location-Info	Optional	Indicates details of where the User Equipment is currently located (for example, RAI, SAI, or CGI).
Service-Information.PS-Information.3GPP-RAT-Type	Optional	Indicates which Radio Access Technology is currently serving the User Equipment. This is one of the parameters used to determine the Application and Sub Type.

Table 92 Credit Control Answer – Event with Reservation – Initial Exchange

AVP	Mandatory?	Description
Session-Id	Mandatory	Session ID
Result-Code	Mandatory	Result of the operation
Origin-Host	Mandatory	Origin Host
Origin-Realm	Mandatory	Origin Realm
Auth-Application-Id	Mandatory	4
CC-Request-Type	Mandatory	INITIAL_REQUEST
CC-Request-Number	Mandatory	From the request
Origin-State-Id	Optional	Origin State ID
Event-Timestamp	Optional	Timestamp of the charge event
Multiple-Services-Credit-Control.Granted-Service-Unit	Mandatory	Number of units that were granted. Contains some combination of the AVPs listed in Table 89, "Requested-Service-Unit, Granted-Service-Unit, and Used-Service-Unit Member AVPs"
Multiple-Services-Credit-Control.Service-Identifier	Optional	Identifies the service.
Multiple-Services-Credit-Control.Rating-Group	Optional	Identifies the rating group consisting of multiple services.
Multiple-Services-Credit-Control.Result-Code	Optional	Result code for the MSCC
Multiple-Services-Credit-Control.Time-Quota-Threshold	Optional	Informs the Credit Control Client to seek reauthorization from the server for the quota when the quota contents fall below the supplied threshold.
Multiple-Services-Credit-Control.Volume-Quota-Threshold	Optional	Informs the Credit Control Client to seek reauthorization from the server for the quota when the quota contents fall below the supplied threshold. The volume quota used in the calculation is one of the following possibilities: ■ CC-Total-Octets ■ CC-Input-Octets ■ CC-Output-Octets ■ CC-Input-Octets + CC-Output-Octets
Multiple-Services-Credit-Control.Quota-Holding-Time	Optional	Contains the quota holding time in seconds. The Credit Control Client deems a quota to have expired when no traffic associated with the quota is observed for the value indicated by this AVP. This AVP applies equally to the granted time quota and the granted volume quota.
Multiple-Services-Credit-Control.Quota-Consumption-Time	Optional	Contains an idle threshold time in seconds. This is applicable when the granted quota is a time quota.
Multiple-Services-Credit-Control.Validity-Time	Optional	Time duration in seconds that the reservation is valid.

Table 92 Credit Control Answer – Event with Reservation – Initial Exchange (Continued)

AVP	Mandatory?	Description
Multiple-Services-Credit-Control.Final-Unit-Indication.Final-Unit-Action	Optional	Populated when a partial reservation was done or the Rating database was down. Action is either Redirect or Terminate. The default is Terminate.
Multiple-Services-Credit-Control.Final-Unit-Indication.Redirect-Server.Redirect-Address-Type	Optional	Address Type of the Redirect Server. Used if Final-Unit-Action is Redirect.
Multiple-Services-Credit-Control.Final-Unit-Indication.Redirect-Server.Redirect-Server-Address	Optional	Address of the Redirect Server. Used if Final-Unit-Action is Redirect.

Terminate Exchange

Table 93 Credit Control Request – Event with Reservation – Terminate Exchange

AVP	Mandatory?	Description
Session-Id	Mandatory	Session ID
Origin-Host	Mandatory	Origin Host
Origin-Realm	Mandatory	Origin Realm
Destination-Host	Mandatory	Destination Host. The value of Destination-Host comes from the Origin-Host AVP in the CCA-Initial.
Destination-Realm	Mandatory	Destination Realm
Auth-Application-Id	Mandatory	4
Service-Context-Id	Mandatory	Used to identify the service logic at the PS DCI.
CC-Request-Type	Mandatory	TERMINATION_REQUEST
CC-Request-Number	Mandatory	Request Number
Multiple-Services-Credit-Control.Used-Service-Unit	Mandatory	Number of units that were used. Contains some combination of the AVPs listed in Table 89, “Requested-Service-Unit, Granted-Service-Unit, and Used-Service-Unit Member AVPs”
Multiple-Services-Credit-Control.Service-Identifier	Optional	Identifies the service. This is one of the parameters used to determine the Application and Sub Type.
Multiple-Services-Credit-Control.Rating-Group	Optional	Identifies the rating group consisting of multiple services. This is one of the parameters used to determine the Application and Sub Type.
Event-Timestamp	Optional	Timestamp of the charge event

Table 94 Credit Control Answer– Event with Reservation – Terminate Exchange

AVP	Mandatory?	Description
Session-Id	Mandatory	Session ID
Result-Code	Mandatory	Result of the operation
Origin-Host	Mandatory	Origin Host

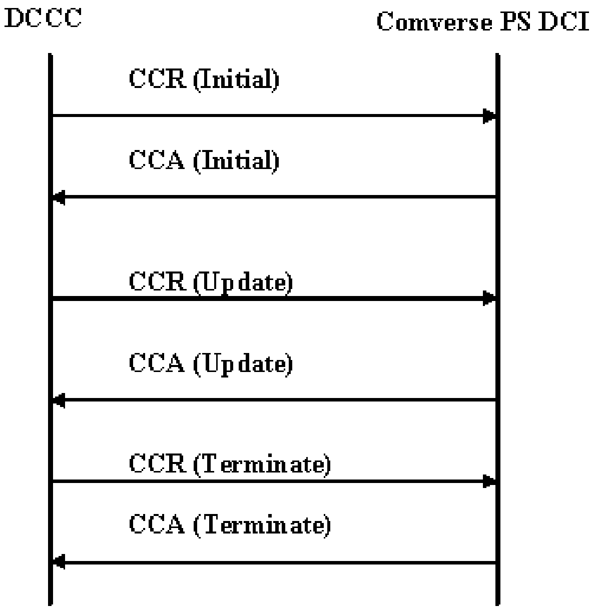
Table 94 Credit Control Answer– Event with Reservation – Terminate Exchange (Continued)

AVP	Mandatory?	Description
Origin-Realm	Mandatory	Origin Realm
Auth-Application-Id	Mandatory	4
CC-Request-Type	Mandatory	TERMINATION_REQUEST
CC-Request-Number	Mandatory	From the request
Origin-State-Id	Optional	Origin State ID
Event-Timestamp	Optional	Timestamp of the charge event

Session-Based Charge

The Session-Based Charge scenario, shown in Figure 67, "Session-Based Charge", is used when the consumed units are not predetermined. In this kind of exchange, the subscriber is authenticated and some initial value is reserved. At some later point, the DCCC notifies the PS DCI that some consumption has occurred and additional balance is reserved. At the close of the session, the final consumption amount is reported.

Figure 67 Session-Based Charge



NOTE The CCR/CCA exchange for Update is repeated as often as necessary to continue the session.

Initial Exchange

Table 95 Credit Control Request – Session-Based Charge – Initial Exchange

AVP	Mandatory?	Description
Session-Id	Mandatory	Session ID
Origin-Host	Mandatory	Origin Host
Origin-Realm	Mandatory	Origin Realm
Destination-Realm	Mandatory	Destination Realm
Auth-Application-Id	Mandatory	4
Service-Context-Id	Mandatory	Used to identify the service logic at the PS DCI. Also, this is one of the parameters used to determine the Application and Sub Type.
CC-Request-Type	Mandatory	INITIAL_REQUEST
CC-Request-Number	Mandatory	Request Number
Subscription-Id: Subscription-Id-Type	Mandatory	Uses the values MSISDN, IMSI, or SIP URI. Refer to Table 83, "Subscription ID AVP Fields".
Subscription-Id: Subscription-Id-Data	Mandatory	UTF8String
Multiple-Services-Indicator	Mandatory	Indicates whether multiple services are supported or not. For PS Domain Charging, the value is MULTIPLE_SERVICES_SUPPORTED.
Multiple-Services-Credit-Control.Requested-Service-Unit.	Optional	Can be empty or can contain some combination of member AVPs listed in table Table 85, "PS-Information AVPs Important for PS Charging"
Multiple-Services-Credit-Control.Service-Identifier	Optional	Identifies the service. This is one of the parameters used to determine the Application and Sub Type.
Multiple-Services-Credit-Control.Rating-Group	Optional	Identifies the rating group consisting of multiple services. This is one of the parameters used to determine the Application and Sub Type.
Event-Timestamp	Optional	Timestamp of the charge event
Service-Information.PS-Information.3GPP-GPRS-Negotiated-QoS-Profile	Optional	The QOS profile applied by the GGSN. This is one of the parameters used to determine the Application and Sub Type.
Service-Information.PS-Information.SGSN-Address	Optional	Holds the SGSN IP address that is used by the GTP control plane for the handling of control messages.
Service-Information.PS-Information.Called-StationId.	Optional	Contains the identifier of the access point name (APN) the user is connected to. This is one of the parameters used to determine the Application and Sub Type.
Service-Information.PS-Information.3GPP-SGSN-MCC-MNC	Optional	Holds the MCC and MNC extracted from the RAI within the Create PDP Context Request ou Update PDP Context Request message.
Service-Information.PS-Information.3GPP-MS-Timezone	Optional	Indicates the offset between universal time and local time in steps of 15 minutes of where the MS currently resides.
Service-Information.PS-Information.3GPP-User-Location-Info	Optional	Indicates details of where the User Equipment is currently located (for example, RAI, SAI, or CGI).
Service-Information.PS-Information.3GPP-RAT-Type	Optional	Indicates which Radio Access Technology is currently serving the User Equipment. This is one of the parameters used to determine the Application and Sub Type.

Table 96 Credit Control Answer – Session-Based Charge – Initial Exchange

AVP	Mandatory?	Description
Session-Id	Mandatory	Session ID
Result-Code	Mandatory	Result of the operation
Origin-Host	Mandatory	Origin Host
Origin-Realm	Mandatory	Origin Realm
Auth-Application-Id	Mandatory	4
CC-Request-Type	Mandatory	INITIAL_REQUEST
CC-Request-Number	Mandatory	From the request
Origin-State-Id	Optional	Origin State ID
Event-Timestamp	Optional	Timestamp of the charge event
Multiple-Services-Credit-Control.Granted-Service-Unit	Mandatory	Number of units that were granted
Multiple-Services-Credit-Control.Service-Identifier	Optional	Identifies the service.
Multiple-Services-Credit-Control.Rating-Group	Optional	Identifies the rating group consisting of multiple services.
Multiple-Services-Credit-Control.Result-Code	Optional	Result code for the MSCC
Multiple-Services-Credit-Control.Time-Quota-Threshold	Optional	Informs the Credit Control Client to seek reauthorization from the server for the quota when the quota contents fall below the supplied threshold.
Multiple-Services-Credit-Control.Volume-Quota-Threshold	Optional	Informs the Credit Control Client to seek reauthorization from the server for the quota when the quota contents fall below the supplied threshold. The volume quota used in the calculation is one of the following possibilities: ▪ CC-Total-Octets ▪ CC-Input-Octets ▪ CC-Output-Octets ▪ CC-Input-Octets + CC-Output-Octets
Multiple-Services-Credit-Control.Quota-Holding-Time	Optional	Contains the quota holding time in seconds. The Credit Control Client deems a quota to have expired when no traffic associated with the quota is observed for the value indicated by this AVP. This AVP applies equally to the granted time quota and the granted volume quota.
Multiple-Services-Credit-Control.Quota-Consumption-Time	Optional	Contains an idle threshold time in seconds. This is applicable when the granted quota is a time quota.
Multiple-Services-Credit-Control.Validity-Time	Optional	Time duration in seconds that the reservation is valid.

Table 96 Credit Control Answer – Session-Based Charge – Initial Exchange (Continued)

AVP	Mandatory?	Description
Multiple-Services-Credit-Control.Final-Unit-Indication.Final-Unit-Action	Optional	Populated when a partial reservation was done or the rating database was down. Action is either Redirect or Terminate. The default is Terminate.
Multiple-Services-Credit-Control.Final-Unit-Indication.Redirect-Server.Redirect-Address-Type	Optional	Address Type of the Redirect Server. Used if Final-Unit-Action is Redirect.
Multiple-Services-Credit-Control.Final-Unit-Indication.Redirect-Server.Redirect-Server-Address	Optional	Address of the Redirect Server. Used if Final-Unit-Action is Redirect.

Update Exchange

Table 97 Credit Control Request – Session-Based Charge – Update Exchange

AVP	Mandatory?	Description
Session-Id	Mandatory	Session ID
Origin-Host	Mandatory	Origin Host
Origin-Realm	Mandatory	Origin Realm
Destination-Host	Mandatory	Destination Host. The value of Destination-Host comes from the Origin-Host AVP in the CCR-Initial.
Destination-Realm	Mandatory	Destination Realm
Auth-Application-Id	Mandatory	4
Service-Context-Id	Mandatory	Used to identify the service logic at the PS DCI.
CC-Request-Type	Mandatory	UPDATE_REQUEST
CC-Request-Number	Mandatory	Request Number
Multiple-Services-Credit-Control.Requested-Service-Unit	Mandatory	Can be either not, present, empty or can contain some combination of member AVPs listed in Table 89, “Requested-Service-Unit, Granted-Service-Unit, and Used-Service-Unit Member AVPs”
Multiple-Services-Credit-Control.Used-Service-Unit	Mandatory	Contains some combination of member AVPs listed in Table 89, “Requested-Service-Unit, Granted-Service-Unit, and Used-Service-Unit Member AVPs”
Multiple-Services-Credit-Control.Service-Identifier	Optional	Identifies the service. This is one of the parameters used to determine the Application and Sub Type.
Multiple-Services-Credit-Control.Rating-Group	Optional	Identifies the rating group consisting of multiple services. This is one of the parameters used to determine the Application and Sub Type.
Event-Timestamp	Optional	Timestamp of the charge event

Table 98 Credit Control Answer – Session-Based Charge – Update Exchange

AVP	Mandatory?	Description
Session-Id	Mandatory	Session ID
Result-Code	Mandatory	Result of the operation
Origin-Host	Mandatory	Origin Host
Origin-Realm	Mandatory	Origin Realm
Auth-Application-Id	Mandatory	4
CC-Request-Type	Mandatory	UPDATE_REQUEST
CC-Request-Number	Mandatory	Request Number
Origin-State-Id	Optional	Origin State ID
Event-Timestamp	Optional	Timestamp of the charge event
Multiple-Services-Credit-Control.Granted-Service-Unit	Mandatory	Number of units that were granted Contains some combination of member AVPs listed in Table 89, “Requested-Service-Unit, Granted-Service-Unit, and Used-Service-Unit Member AVPs”
Multiple-Services-Credit-Control.Service-Identifier	Optional	Identifies the service.
Multiple-Services-Credit-Control.Rating-Group	Optional	Identifies the rating group consisting of multiple services.
Multiple-Services-Credit-Control.Result-Code	Optional	Result code for the MSCC
Multiple-Services-Credit-Control.Time-Quota-Threshold	Optional	Informs the Credit Control Client to seek reauthorization from the server for the quota when the quota contents fall below the supplied threshold
Multiple-Services-Credit-Control.Volume-Quota-Threshold	Optional	Informs the Credit Control Client to seek reauthorization from the server for the quota when the quota contents fall below the supplied threshold The volume quota used in the calculation is one of the following possibilities. ■ CC-Total-Octets ■ CC-Input-Octets ■ CC-Output-Octets ■ CC-Input-Octets + CC-Output-Octets
Multiple-Services-Credit-Control.Quota-Holding-Time	Optional	Contains the quota holding time in seconds. The Credit Control Client deems a quota to have expired when no traffic associated with the quota is observed for the value indicated by this AVP. This AVP applies equally to the granted time quota and the granted volume quota.
Multiple-Services-Credit-Control.Quota-Consumption-Time	Optional	Contains an idle threshold time in seconds. This is applicable when the granted quota is a time quota.
Multiple-Services-Credit-Control.Validity-Time	Optional	Time duration in seconds that the reservation is valid.

Table 98 Credit Control Answer – Session-Based Charge – Update Exchange (Continued)

AVP	Mandatory?	Description
Multiple-Services-Credit-Control.Final-Unit-Indication.Final-Unit-Action	Optional	Populated when a partial reservation was done or the rating database was down. Action is either Redirect or Terminate. The default is Terminate.
Multiple-Services-Credit-Control.Final-Unit-Indication.Redirect-Server.Redirect-Address-Type	Optional	Address Type of the Redirect Server. Used if Final-Unit-Action is Redirect.
Multiple-Services-Credit-Control.Final-Unit-Indication.Redirect-Server.Redirect-Server-Address	Optional	Address of the Redirect Server. Used if Final-Unit-Action is Redirect.

Terminate Exchange

Table 99 Credit Control Request – Session-Based Charge – Terminate Exchange

AVP	Mandatory?	Description
Session-Id	Mandatory	Session ID
Origin-Host	Mandatory	Origin Host
Origin-Realm	Mandatory	Origin Realm
Destination-Host	Mandatory	Destination Host. The value of Destination-Host comes from the Origin-Host AVP in previous CCR.
Destination-Realm	Mandatory	Destination Realm
Auth-Application-Id	Mandatory	4
Service-Context-Id	Mandatory	Used to identify the service logic at the PS DCI.
CC-Request-Type	Mandatory	TERMINATION_REQUEST
CC-Request-Number	Mandatory	Request Number
Multiple-Services-Credit-Control.Used-Service-Unit	Mandatory	Contains some combination of member AVPs listed in Table 89, “Requested-Service-Unit, Granted-Service-Unit, and Used-Service-Unit Member AVPs”
Multiple-Services-Credit-Control.Service-Identifier	Optional	Identifies the service. This is one of the parameters used to determine the Application and Sub Type.
Multiple-Services-Credit-Control.Rating-Group	Optional	Identifies the rating group consisting of multiple services. This is one of the parameters used to determine the Application and Sub Type.
Event-Timestamp	Optional	Timestamp of the charge event

Table 100 Credit Control Answer – Session-Based Charge – Terminate Exchange

AVP	Mandatory?	Description
Session-Id	Mandatory	Session ID
Result-Code	Mandatory	Result of the operation
Origin-Host	Mandatory	Origin Host
Origin-Realm	Mandatory	Origin Realm
Auth-Application-Id	Mandatory	4

Table 100 Credit Control Answer – Session-Based Charge – Terminate Exchange (Continued)

AVP	Mandatory?	Description
CC-Request-Type	Mandatory	TERMINATION_REQUEST
CC-Request-Number	Mandatory	From the request
Origin-State-Id	Mandatory	Origin State ID
Event-Timestamp	Optional	Timestamp of the charge event

Limitations

- Advice of Charge (Price Inquiry Request), Balance Check, Direct Debit, and Refund are not supported.
- Tariff switching is not supported.
- Trigger Type is not supported.
- CCR-Update based on Trigger is not supported.

Unsupported AVPs

The following tables list the unsupported AVPs in the PS DCI.

Table 101 Unsupported AVPs within the Credit Control Request Message

Credit-Control Request Message Unsupported AVPs
User-Name
Origin-State-Id
Termination-Cause
Reporting-Reason (within MSCC)
Tariff-Change-Usage (within MSCC)
Trigger (within MSCC)
Trigger-Type
User-Equipment-Info
User-Equipment-Info-Type
User-Equipment-Info-Value
Route-Record

Table 102 Unsupported AVPs within the Credit Control Answer Message

Credit-Control Answer Message Unsupported AVPs
Tariff-Time-Change (within MSCC.GSU)
G-S-U-Pool-Reference (within MSCC.GSU)
G-S-U-Pool-Identifier
CC-Unit-Type
Unit-Value

Table 102 Unsupported AVPs within the Credit Control Answer Message (Continued)

Credit-Control Answer Message Unsupported AVPs
Value-Digits
Exponent
Unit-Quota-Threshold (within MSCC)
Trigger (within MSCC)
Trigger-Type
PS-Furnish-Charging-Information (within MSCC)
Redirect-Host
Redirect-Host-Usage
Redirect-Max-Cache-Time

Table 103 Unsupported AVPs within the PS-Information AVP

PS-Information Unsupported AVPs
3GPP-Charging-Id
3GPP-PDP-Type
PDP-Address
GGSN-Address
CG-Address
3GPP-IMSI-MCC-MNC
3GPP-GGSN-MCC-MNC
3GPP-NSAPI
3GPP-Session-Stop-Indicator
3GPP-Selection-Mode
3GPP-Charging-Characteristics
3GPP-MS-TimeZone
Charging-Rule-Base-Name
PS-Furnish-Charging-Information
PDP-Context-Type

Format of Configurable Parameters

This section describes the format of the configurable parameters within the configuration file.

Authorization-only

The format of this configurable parameter is:

```
<authorization-policy authorization-only="false"/>
```

CDR Information

The format of this configurable parameter is:

```
<cdr-information>
    <cdr-info-avp name="CCR.Service-Information.PS-Information.Called-Station-Id"
          token="ps-cld-st-id"/>
    <cdr-info-avp name="CCR.Service-Information.PS-Information._3GPP-GPRS-
          Negotiated-QoS-Profile" token="ps-qos"/>
    <cdr-info-avp name="CCR.Multiple-Services-Credit-Control.Rating-Group"
          token="rg"/>
    <cdr-info-avp name="CCR.Multiple-Services-Credit-Control.Service-Identifier"
          token="si"/>
</cdr-information>
```

In the example above, values from four AVPs are used to populate the CDR Information field of the CDR. The name of the token is also specified.

Subscriber Location

The format of this configurable parameter is:

```
<subscriber-location>
    <avp name=”CCR.Service-Information.PS-Information._3GPP-User-Location-Info”/>
    <avp name=”CCR.Service-Information.PS-Information.SGSN-Address”/>
    <avp name=”CCR.Service-Information.PS-Information._3GPP-SGSN-MCC-MNC” location-
          type="MSC-ID"/>
</subscriber-location >
```

In the example above, if the PS-Information.3GPP-User-Location-Info AVP is present, its value is used. If the PS-Information.3GPP-User-Location-Info AVP is not present, the value from the PS-Information.SGSN-Address AVP is used. If the PS-Information.SGSN-Address AVP is not used, the value from the PS-Information.3GPP-SGSN-MCC-MNC AVP is used. If none of the above three AVPs are present, the PS DCI does not provide location information to the PS DCI core application.

Default Data

The format of the default data is:

```
<sle-default-data>
    <Time-Quota-Threshold-Percentage>80</Time-Quota-Threshold-Percentage>
    <Volume-Quota-Threshold-Percentage>90</Volume-Quota-Threshold-Percentage>
    <sle-default-avp name="CCR.Multiple-Services-Credit-Control.Quota-Holding-
          Time">60</sle-default-avp>
    <sle-default-avp name="CCR.Multiple-Services-Credit-Control.Quota-Consumption-
          Time">300</sle-default-avp>
    <sle-default-avp name="CCR.Multiple-Services-Credit-Control.Trigger.Trigger-
          Type">CHANGE_IN_SGSN_IP_ADDRESS</sle-default-avp>
    <sle-default-avp name="CCR.Multiple-Services-Credit-Control.Trigger.Trigger-
          Type">CHANGE_IN_QOS</sle-default-avp>
    <sle-default-avp name="CCR.Multiple-Services-Credit-Control.Trigger.Trigger-
          Type">CHANGE_IN_LOCATION</sle-default-avp>
    <sle-default-avp name="CCR.Multiple-Services-Credit-Control.Trigger.Trigger-
          Type">CHANGE_IN_RAT</sle-default-avp>
    <sle-default-avp name="CCR.Multiple-Services-Credit-Control.Final-Unit-
          Indication.Final-Unit-Action"> REDIRECT</sle-default-avp>
```

```
  <sle-default-avp name="CCR.Multiple-Services-Credit-Control.Final-Unit-
     Indication.Redirect-Server.Redirect-Address-Type">IPV4_ADDRESS</sle-
     default-avp>
  <sle-default-avp name="CCR.Multiple-Services-Credit-Control.Final-Unit-
     Indication.Redirect-Server.Redirect-Server-Address">123.45.67.89</sle-
     default-avp>
   <qos-information qos="Low Speed">01</qos-information>
   <qos-information qos="Low Speed">02</qos-information>
   <qos-information qos="Low Speed">03</qos-information>
   <qos-information qos="Low Speed">04</qos-information>
   <qos-information qos="Low Speed">05</qos-information>
   <qos-information qos="Low Speed">06</qos-information>
   <qos-information qos="Medium Speed">07</qos-information>
   <qos-information qos="Medium Speed">08</qos-information>
   <qos-information qos="Medium Speed">09</qos-information>
   <qos-information qos="Medium Speed">10</qos-information>
   <qos-information qos="Medium Speed">11</qos-information>
   <qos-information qos="Medium Speed">12</qos-information>
   <qos-information qos="Medium Speed">13</qos-information>
   <qos-information qos="High Speed">14</qos-information>
   <qos-information qos="High Speed">15</qos-information>
   <qos-information qos="High Speed">16</qos-information>
   <qos-information qos="High Speed">17</qos-information>
   <qos-information qos="High Speed">18</qos-information>
   <qos-information qos="High Speed">30</qos-information>
   <qos-information qos="High Speed">31</qos-information>
</sle-default-data>
```

```
http://www.3gpp.org/ftp/Specs/html-info/32299.htm
```

```
http://www.3gpp.org/ftp/Specs/html-info/32296.htm
http://www.3gpp.org/ftp/Specs/html-info/32251.htm
http://www.3gpp.org/ftp/Specs/html-info/23125.htm
```

References

Table 104 IETF and 3GPP References

Document Title	Document Location
IETF RFC 3588 Diameter Base Protocol	http://www.rfc-editor.org/rfc/rfc3588.txt
3GPP TS 32.299 Diameter Charging Application	http://www.3gpp.org/ftp/Specs/html-info/32299.htm
3GPP TS 32.296 Online Charging System (OCS): applications and interfaces	http://www.3gpp.org/ftp/Specs/html-info/32296.htm
3GPP TS 32.251 Packet-Switched (PS) domain charging	http://www.3gpp.org/ftp/Specs/html-info/32251.htm

Table 104 IETF and 3GPP References (Continued)

Document Title	Document Location
3GPP TS 23.125 Overall high-level functionality and architecture impacts of Flow-Based charging	http://www.3gpp.org/ftp/Specs/html-info/23125.htm
3GPP TS 29.060 GPRS Tunneling Protocol (GTP) across the Gn and Gp interface	http://www.3gpp.org/ftp/Specs/html-info/29060.htm
3GPP TS 29.061 Internetworking between the Public Land Mobile Network (PLMN) supporting packet based services and Packet Data Networks (PDN)	http://www.3gpp.org/ftp/Specs/html-info/29061.htm

Chapter 8

Comverse IMS Charging Interface

IP Multimedia Subsystem (IMS)

This chapter defines the Diameter Credit Control Application (DCCA) DCI as implemented by Comverse ONE OCS. This interface utilizes the Diameter real time credit control application as defined in IETF RFC 4006, 3GPP TS 32.299 and 3GPP TS 32.260, so as to provide the rating parameters required by Comverse ONE for session-based rating and charging. The document 3GPP 32.260 defines the specific requirements associated with the IMS charging model.

Using the IMS DCCA, DCCC applications can invoke existing Comverse ONE Rating and Charging capabilities.

Support for IP Multimedia Subsystem DCCA allows the Comverse ONE solution to act as a Diameter Credit Control Server and provide real time rating and charging capabilities to Diameter Credit Control Clients.

IMS DCCA is specified in IETF RFC 4006 and is a general real time charging interface specification. It provides a framework on the way a charging client and server communicate and how the quota for a service is requested, granted, and reported. DCCA also categorizes the types of charging into event charging and session charging. DCCA does not specify which rating parameters are used to rate the specific application or service accessed by the user. For example, a ringtone download rating may be based on the type of ringtone downloaded but a video call depends on the video quality, location of participants, and so on. The charging client and server must agree on the rating AVPs used for the service. Each service may have a separate set of rating AVPs. The application/service-related information is added by the vendor (or standards) thus defining an extension to DCCA.

IMS DCCA Message Details

3GPP DCCA defines the following message to support real time charging capabilities:

- **Credit Control Request (CCR):** Message is sent from DCC client to request charging and/or rating from the DCC server.
- **Credit Control Answer (CCA):** Message is sent as a response from the DCC server to the DCC client.

NOTE Details of the AVPs in CCR messages are defined in Diameter Base RFC 3588 and 3GPP TS 32.299.

Application and Sub Type Determination

Application and sub type are used to perform rating in the Comverse ONE solution. Application and sub type are derived from input parameters. One of the functions of the IMSDCCA is to determine the application and sub type. This is performed through the use of a table accissible via the Product Catalog. The table contains columns as outlined below and the application and sub type are derived from the available input parameters for the other columns. The IMS DCCA Service Mapping table is as follows:

Table 105 IMS DCCA Service Mapping Table

Column	Type	Description
Service-Context-Id	M	The [Service-Context-Id] AVP value of the CCR-Initial message.
Audio	M	A Boolean value. “True” if one of the values of the AVP [Service-Information.IMS-Information.SDP-Media-Component.SDP-Media-Name] contains “audio.”
Audio_QOS	O	The QoS of the media type found above. It is obtained from the same SDP-Media-Component but under the [Service-Information.IMS-Information.SDP-Media-Component.Authorized-QoS] AVP.
Video	M	A Boolean value. “True” if one of the values of the AVP [Service-Information.IMS-Information.SDP-Media-Component.SDP-Media-Name] contains “video.”
Video_QoS	O	The QoS of the media type found above. It is obtained from the same SDP-Media-Component but under the [Service-Information.IMS-Information.SDP-Media-Component.Authorized-QoS] AVP.
Application	M	A Boolean value. “True” if one of the values of the AVP [Service-Information.IMS-Information.SDP-Media-Component.SDP-Media-Name] contains “application.”
Application_QoS	O	The QoS of the media type found above. It is obtained from the same SDP-Media-Component but under the [Service-Information.IMS-Information.SDP-Media-Component.Authorized-QoS] AVP.
Data	M	A Boolean value. “True” if one of the values of the AVP [Service-Information.IMS-Information.SDP-Media-Component.SDP-Media-Name] contains “data.”
Data_QoS	O	The QoS of the media type found above. It is obtained from the same SDP-Media-Component but under the [Service-Information.IMS-Information.SDP-Media-Component.Authorized-QoS] AVP.
Application Name	R	Application identifier assigned to a specific type of activity, such as IMS.
Application Sub Type	R	Identifier of the sub type associated with this application activity, such as IMS Audio or IMS Video.

O = Optional, M = Mandatory, R = Result

Literals used are True and False and must be an exact match.

Diameter Request Handling

The most important action of the Diameter Request handling is the selection of the application and sub type parameters used to initiate the billing process. This selection is based on a comparison of several AVPs to columns in the IMS DCCA Service Mapping Table. The result of the table lookup is the application and sub type parameter from this table.

Subscriber ID and Subscriber ID Type

The Subscriber ID is obtained from the [CCR.Subscription-Id.Subscription-Id-Data] AVP. The Subscriber ID Type is determined by the [CCR.Subscription-Id.Subscription-Id-Type] AVP. The value of the AVP is used to map to existing vlaues in the External Id Type table. When the External ID Type is not MSISDN, the Subscriber-Id-Data and the External Id Type are then used as inputs to the External Id Equipment Map table to determine the Subscriber ID.

Unit Determination

In addition to selection of the application and sub type, the IMS DCCA also inspects the received message Requested-Service-Unit AVP, if present, to determine how to perform unit determination. If the Requested-Service-Unit AVP is empty, then the DCCA reverts to Centralized Unit Determination to select the correct unit.

The following table identifies the possible member AVPs of the Requested-Service-Unit AVP. These member AVPs are also used within the Granted-Service-Unit AVP and Used-Service-Unit AVP.

Table 106 Requested-Service-Unit, Granted-Service-Unit, and Used-Service-Unit Member AVPs

Requested-Service-Unit Member AVP	Comverse ONE Unit Type
CC-Time	SECONDS
CC-Service-Specific-Units	Utilizes centralized unit determination.

Location Parameters

In cases where tariff determination is based on the location of the subscriber or destination, then location parameters are extracted from the received Diameter message and transferred to the URE. There are two pairs of parameters used for location: one for subscriber location and the other for the location of the destination or other location.

The IMS DCCA uses the value of the AVP [CCR.Service-Information.IMS-Information.Role-Of-Node] to determine the value of the other-location:

- If [Role-Of-Node] is ORIGINATING_ROLE (0), then derive the value from [CCR.Service-Information.IMS-Information.Called-Party-Address].
- If [Role-Of-Node] is TERMINATING_ROLE (1), then derive the value from [CCR.Service-Information.IMS-Information.Calling-Party-Address.

The subscriber location information comes from the following AVP:

- Service-Information.PS-Information.3GPP_User-Location-Info

Diameter Answer Handling

All Diameter Requests sent to the OCS are acknowledged with a Diameter Answer but receipt of a Diameter Answer does not constitute a successful operation. The Diameter Answer contains the Result-Code AVP which must be inspected by the Diameter Client to determine success of the message.

The IMS-SLE CCA AVPs are defined in the ims_answer_template.xml file. The table below represents the CCAs and the contents that are defined for the IMS-SLE service. The OCS process builds the CCA message and, if the SLE registers for further manipulation of the CCA, conditionally hands the CCA over to the SLE for further processing. The following table provides CCA message types and formats:

Table 107 CCA Message Types and Formats

CCA Type	AVP Contents
CCA-Initial-OK	Session-Id Result-Code Origin-Host Origin-Realm Auth-Application-Id (4) CC-Request-Type (INITIAL_REQUEST) CC-Request-Number Origin-State-Id Event-Timestamp Multiple-Service-Credit-Control Granted-Service-Unit CC-Time (Reserved-Amount) Validity-Time (Reservation-Lifetime) Time-Quota-Threshold (Reservation-Lifetime) *Quota-Holding-Time (n/a for IMS-SLE)* Quota-Consumption-Time (n/a for IMS-SLE)
CCA-Initial-Final	Session-Id Result-Code Origin-Host Origin-Realm Auth-Application-Id (4) CC-Request-Type (INITIAL_REQUEST) CC-Request-Number Origin-State-Id Event-Timestamp Multiple-Service-Credit-Control Granted-Service-Unit CC-Time (Reserved-Amount) Validity-Time (Reservation-Lifetime) Time-Quota-Threshold (Reservation-Lifetime) Final-Unit-Indication Final-Unit-Action (TERMINATE)
CCA-Initial-Error	Session-Id Result-Code Origin-Host Origin-Realm Auth-Application-Id (4) CC-Request-Type (INITIAL_REQUEST) CC-Request-Number Origin-State-Id Event-Timestamp

Table 107 CCA Message Types and Formats (Continued)

CCA Type	AVP Contents
CCA-Update-OK	Session-Id Result-Code Origin-Host Origin-Realm Auth-Application-Id (4) CC-Request-Type (UPDATE_REQUEST) CC-Request-Number Origin-State-Id Event-Timestamp Multiple-Service-Credit-Control Granted-Service-Unit CC-Time (Reserved-Amount) Validity-Time (Reservation-Lifetime) Time-Quota-Threshold (Reservation-Lifetime) *Quota-Holding-Time (n/a for IMS-SLE)* Quota-Consumption-Time (n/a for IMS-SLE)
CCA-Update-Final	Session-Id Result-Code Origin-Host Origin-Realm Auth-Application-Id (4) CC-Request-Type (UPDATE_REQUEST) CC-Request-Number Origin-State-Id Event-Timestamp Multiple-Service-Credit-Control Granted-Service-Unit CC-Time (Reserved-Amount) Validity-Time (Reservation-Lifetime) Time-Quota-Threshold (Reservation-Lifetime) Final-Unit-Indication Final-Unit-Action (TERMINATE)
CCA-Update-Error	Session-Id Result-Code Origin-Host Origin-Realm Auth-Application-Id (4) CC-Request-Type (UPDATE_REQUEST) CC-Request-Number Origin-State-Id Event-Timestamp

Table 107 CCA Message Types and Formats (Continued)

CCA Type	AVP Contents
CCA-Terminate-OK	Session-Id Result-Code Origin-Host Origin-Realm Auth-Application-Id (4) CC-Request-Type (TERMINATE_REQUEST) CC-Request-Number Origin-State-Id Event-Timestamp
CCA-Terminate-OK-AOC (for Advice of Charge)	Session-Id Result-Code Origin-Host Origin-Realm Auth-Application-Id (4) CC-Request-Type (TERMINATE_REQUEST) CC-Request-Number Origin-State-Id Event-Timestamp Cost-Information Unit-Value Value-Digits (Charged-Amount) Exponent (Charged-Amount-Exponent)
CCA-Terminate-Error	Session-Id Result-Code Origin-Host Origin-Realm Auth-Application-Id (4) CC-Request-Type (TERMINATE_REQUEST) CC-Request-Number Origin-State-Id Event-Timestamp

The Flex SLE framework provides the scripting capability to further modify the value of the AVP in question. For the IMS SLE, special treatment is required for the Time-Quota-Threshold AVP in the CCA-Initial-OK and CCA-Update-OK messages. The value of this AVP is calculated in the script by subtracting the original value from the provisional threshold value defined in the ims_entity_def.xml file.

Any Result-Code that is not DIAMETER_SUCCESS or DIAMETER_LIMITED_SUCCESS indicates that there was an error in the operation. In addition, certain errors indicate that the reservation-based session is now closed.

In general, DIAMETER_SUCCESS indicates that the operation was completely successful and DIAMETER_LIMITED_SUCCESS indicates that some part of the operation did not succeed. For instance, if the subscriber has limited funds and the granted amount is only a portion of the requested amount, DIAMETER_LIMITED_SUCCESS is returned. Regardless, the Granted-Service-Units AVP are consulted in all cases to determine the allowed quota.

The following table lists possible values for the Result Code AVP in the Answer message.

Table 108 Result Code AVP Values in Answer Message

Enumerated Token	Value	Description	Session Closed?
DIAMETER_SUCCESS	2001	The operation was completely successful.	No
DIAMETER_LIMITED_SUCCESS	2002	Some part of the operation did not succeed. For example, if the subscriber has limited funds and the granted amount is only a portion of the requested amount.	No
DIAMETER_UNABLE_TO_DELIVER	3002	The DCCS was unable to process the message. This may occur because there was a problem decoding the message or there is no connection to the peer host.	No
DIAMETER_REALM_NOT_SERVED	3003	The DCCS received a message that did not contain an Origin-Host AVP.	No
DIAMETER _TOO_BUSY	3004	The DCCS cannot process any more transactions because it has reached its maximum number of concurrent transactions.	No
DIAMETER_INVALID_AVP_BITS	3009	The Origin-Host AVP is malformed.	No
DIAMETER_UNKNOWN_PEER	3010	The configured peer host does not match the value in the Origin-Host AVP in the Capabilities Exchange Request or Capabilities Exchange Answer.	No
DIAMETER_AUTHENTICATION_REJECTED	4001	The DCCS determined that the user account is invalid.	No
DIAMETER_OUT_OF_SPACE	4001	The DCCS could not allocate space for a session	Yes
DIAMETER_END_USER_SERVICE_DENIED	4010	The DCCS determined that the service was denied. Possible reasons include: ■ the activity is not allowed in the current subscriber state ■ cannot load account data ■ an exception condition occurred while processing an event or session	No
DIAMETER_CREDIT_LIMIT_REACHED	4012	The reservation limit for the user was reached.	No
DIAMETER_UNKNOWN_SESSION_ID	5002	The DCCS received an unknown session ID. For example, it received a session ID in a CCR-Update or CCR-Terminate which does not match the session ID for any currently open sessions.	Yes
DIAMETER_INVALID_AVP_VALUE	5004	The DCCS was unable to retrieve data from an AVP.	No
DIAMETER_MISSING_AVP	5005	A mandatory AVP is missing from the message.	No

Table 108 Result Code AVP Values in Answer Message (Continued)

Enumerated Token	Value	Description	Session Closed?
DIAMETER_NO_COMMON_APPLICATION	5010	The Auth-Application-Id AVP or the Acct-Application-Id AVP has one of these issues: ■ does not match the values configured in the DCCS ■ there is no service logic to handle the message ■ there was a problem decoding the message	No
DIAMETER_UNABLE_TO_COMPLY	5012	The DCCS was unable to process the message. This may occur when the AVP(s) is missing or an application and sub type cannot be determined. Another possibility is a Capabilities Exchange Request was received with an Application ID not equal to 0.	No
DIAMETER_USER_UNKNOWN	5030	The DCCS was unable to access the user’s account.	No
DIAMETER_RATING_FAILED	5031	Failed for one of three reasons: ■ the DCCS had an error with a reservation on the user’s account ■ there was no tariff plan ■ the amount or units were invalid	No

Client Requirements for Sessions Using Multiple Transactions

In the situation where the charging scenario requires multiple Diameter transactions for the same session ID, there is one additional requirement for the Diameter Client application when interfacing to the Comverse ONE OCS. This includes any scenario that starts with a CCR with the CC-Request-Type AVP as INITIAL_REQUEST and is followed by a CCR with the CC-Request-Type AVP set to UPDATE_REQUEST or TERMINATE_REQUEST.

In order to ensure proper operation, it is necessary for the Client application to populate the Destination-Host AVP in subsequent CCR messages sent to the Comverse ONE OCS for the same Session-ID. The value for the Destination-Host AVP is reported to the Client application in Origin-Host AVP from the last CCA message received from the Comverse ONE OCS for the corresponding Session-ID. The value of the Origin-Host AVP in the CCA identifies the subsystem that is handling the session. Providing Destination-Host AVP in the CCR for UPDATE_REQUEST and TERMINATE_REQUEST ensures that the correct subsystem is addressed for these subsequent requests.

OCS Service Abort Request Handling

During the IMS session, the OCS may encounter a situation that requires the session to abort. The OCS sends a Service Abort Request to the DCCC to indicate the inability to hold the session on the DCCS side. Although the DCCC replies with a Service Abort Answer, the session termination doesn’t occur until a CCR-Terminate message is received from the DCCC. The following table describes the message format of the Abort-Session-Request (ASR).

Table 109 CCR Abort Session Request Format

CCA Type	AVP Contents
CCR-SVC-Abort-Request	Session-Id Origin-Host Origin-Realm Destination-Realm Destination-Host Auth-Application-Id (4)

Charging Scenario

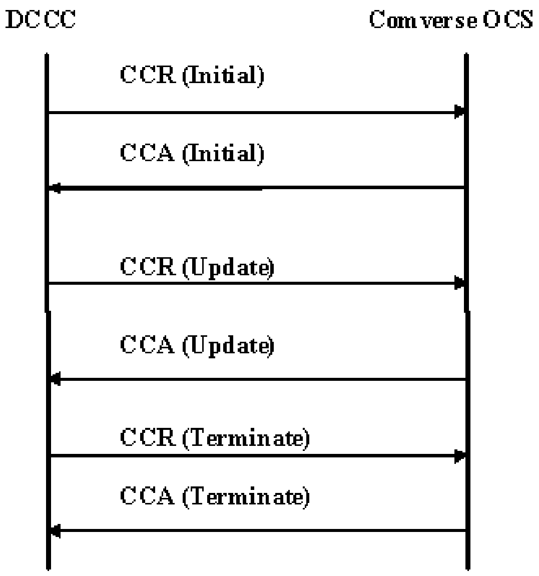
The IMS DCCA only supports the Session Charge with Reservation. This scenario has the following variations:

- **Decentralized Unit determination / Centralized Rating:** The Multiple-Services-CreditControl.Requested-Service-Unit AVP is present and contains unit(s).
- **Centralized Unit determination / Centralized Rating:** The Multiple-Services-Credit-Control. Requested-Service-Unit AVP is empty.

The expected AVPs in the initial request are dependent on the charging variations. The rules are applied consistently across scenarios.

The following scenario is used when the consumed units are not predetermined. The subscriber is authenticated and some initial value is reserved. Later, the DCCC notifies the DCCS that some consumption has occurred and additional balance is reserved. At the close of the session, the final consumption amount is reported.

Figure 68 Session Based Charge



NOTE The CCR/CCA exchange for Update is repeated as often as necessary to continue the session.

In the tables below, the following key applies:

Key Type:

M = Mandatory. Must be present based on message syntax.

R = Semantically Required. Required as part of the scenario.

O = Optional. May be present.

CCR for Initial Request

The following table provides the CCR needed for the Initial Request.

Table 110 Credit Control Request for Initial Request

AVP	Type	Description
Session-Id	M	Session ID
Origin-Host	M	Origin Host
Origin-Realm	M	Origin Realm
Destination-Realm	M	Destination Realm
Auth-Application-Id	M	4
Service-Context-Id	M	Used to identify the service logic at the DCCS. For IMS SLE, set to 32260@3gpp.org
CC-Request-Type	M	INITIAL_REQUEST
CC-Request-Number	M	Request Number

Table 110 Credit Control Request for Initial Request

AVP	Type	Description
Subscription-Id. Subscription-ID-Type	R	Uses the values E164, IMSI, SIP URI, NAI or PRIVATE.
Subscription-Id. Subscription-ID-Data	R	UTF8String. Identifies the subscriber.
Multiple-Services-Credit-Control. Requested-Service-Unit. CC-Time	O	Contains the amount of requested time in the unit of SECONDS.
Event-Timestamp	O	Timestamp of the charge event.
Service-Information. IMS-Information. SDP-Media-Component. SDP-Media-Name	R	Contains information used to derive AUT charging factor for SIP sessions. There can be multiple occurrences of this AVP if more than one media is being used in the IMS session. The supported media names are: ■ audio ■ video ■ application ■ data
Service-Information. IMS-Information. SDP-Media-Component. Authorized-QoS	R	Contains the Authorized Quality of Service as defined in TS 23.207 / TS 29.207. It is used to derive AUT factor for SCUR.
Service-Information. IMS-Information. Role-Of-Node.	O	This AVP specifies the role of the AS/CSCF. For IMS, the identifier is either ORIGINATING_ROLE (0) or TERMINATING_ROLE (1). Except when Role-Of-Node is set to (1), the IMS SLE treats the session as Originating.
Service-Information. IMS-Information. Called-Party-Address	O	In IMS charging, it holds the address (SIP URI or TEL URI) of the party to whom the SIP transaction is posted. The IMS SLE supports only the Tel URI.
Service-Information. IMS-Information. Calling-Party-Address	O	In IMS charging, it holds the address (SIP URI or TEL URI) of the party initiating the SIP transaction. The IMS SLE supports only the Tel URI.
Service-Information. PS-Information. 3GPP-User-Location-Info	O	Indicates details of where the User Equipment is currently located.

CCA for Initial Request

The following table provides the CCA needed for the Initial Request.

Table 111 Credit Control Answer for Initial Request

AVP	Type	Description
Session-Id	M	Session ID
Result-Code	M	Result of the operation
Origin-Host	M	Origin Host
Origin-Realm	M	Origin Realm
Auth-Application-Id	M	4
CC-Request-Type	M	INITIAL_REQUEST

Table 111 Credit Control Answer for Initial Request (Continued)

AVP	Type	Description
CC-Request-Number	M	The number from the request.
Origin-State-Id	R	Origin State ID
Event-Timestamp	R	Timestamp of the charge event.
Multiple-Services-Credit-Control. Granted-Service-Unit. CC-Time	R	Contains the amount of the granted time.
Multiple-Services-Credit-Control. Validity-Time	R	Defines the time in order to limit the validity of the granted quota for a given category instance.
Multiple-Services-Credit-Control. Time-Quota-Threshold	O	Informs the Credit Control Client to seek reauthorization from the server for the quota when the quota contents fall below the supplied threshold.
Multiple-Services-Credit-Control. *Quota-Holding-Time*	O	Note: This AVP is not applicable for the IMS use case. Contains the quota holding time in seconds. The CCC deems a quota to have expired when no traffic associated with the quota is observed for the value indicated by this AVP.
Multiple-Services-Credit-Control. *Quota-Consumption-Time*	O	Note: This AVP is not applicable for the IMS use case. Contains an idle threshold time in seconds. This is applicable when the granted quota is a time quota.
Multiple-Services-Credit-Control. Final-Unit-Indication. Final-Unit-Action	O	Populated when a partial reservation was done or the SDP was down. Action is either Redirect or Terminate. The default is TERMINATE.
Multiple-Services-Credit-Control. *Final-Unit-Indication.* *Redirect-Server.* *Redirect-Address-Type*	O	Note: A placeholder is defined in the IMS SLE template configuration file. This AVP contains the Address Type of the Redirect Server. Used if Final-Unit-Action is Redirect.
Multiple-Services-Credit-Control. *Final-Unit-Indication.* *Redirect-Server.* *Redirect-Server-Address*	O	Note: A placeholder is defined in the IMS SLE template configuration file. This AVP contains the Address of the Redirect Server. Used if Final-Unit-Action is Redirect.

CCA for Update Request

The following table provides the CCA needed for the Update Request.

Table 112 Credit Control Answer for Update Request

AVP	Type	Description
Session-Id	M	Session ID
Result-Code	M	Result of the operation
Origin-Host	M	Origin Host
Origin-Realm	M	Origin Realm
Auth-Application-Id	M	4
CC-Request-Type	M	UPDATE_REQUEST
CC-Request-Number	M	Number from the request
Origin-State-Id	M	Origin State ID
Event-Timestamp	M	Timestamp of the charge event

Table 112 Credit Control Answer for Update Request

AVP	Type	Description
Multiple-Services-Credit-Control. Granted-Service-Unit. CC-Time	M	Contains the amount of the granted time.
Multiple-Services-Credit-Control. Validity-Time	O	Defines the time in order to limit the validity of the granted quota for a given category instance.
Multiple-Services-Credit-Control. Time-Quota-Threshold	O	Informs the Credit Control Client to seek reauthorization from the server for the quota when the quota contents fall below the supplied threshold.
Multiple-Services-Credit-Control. *Quota-Holding-Time*	O	Note: This AVP is not applicable for the IMS use case. Contains the quota holding time in seconds. The CCC deems a quota to have expired when no traffic associated with the quota is observed for the value indicated by this AVP.
Multiple-Services-Credit-Control. *Quota-Consumption-Time*	O	Note: This AVP is not applicable for the IMS use case. Contains an idle threshold time in seconds. This is applicable when the granted quota is a time quota.
Multiple-Services-Credit-Control. Final-Unit-Indication. Final-Unit-Action	R	Populated when a partial reservation was done or the SDP was down. Action is either Redirect or Terminate. The default is TERMINATE.
Multiple-Services-Credit-Control. *Final-Unit-Indication.* *Redirect-Server.* *Redirect-Address-Type*	R	Note: A placeholder is defined in the IMS SLE template configuration file. This AVP contains the Address Type of the Redirect Server. Used if Final-Unit-Action is Redirect.
Multiple-Services-Credit-Control. *Final-Unit-Indication.* *Redirect-Server.* *Redirect-Server-Address*	O	Note: A placeholder is defined in the IMS SLE template configuration file. This AVP contains the Address of the Redirect Server. Used if Final-Unit-Action is Redirect.

CCR for Terminate Request

The following table provides the CCR needed for the Terminate Request.

Table 113 Credit Control Request for Terminate Request

AVP	Type	Description
Session-Id	M	Session ID
Origin-Host	M	Origin Host
Origin-Realm	M	Origin Realm
Destination-Realm	M	Destination Realm
Auth-Application-Id	M	4
Service-Context-Id	M	Used to identify the service logic at the DCCS. For IMS SLE, set to 32260@3gpp.org
CC-Request-Type	M	TERMINATE_REQUEST
CC-Request-Number	M	Request Number

Table 113 Credit Control Request for Terminate Request

AVP	Type	Description
Destination-Host	R	This AVP contains the value from the Origin-Host AVP of the last received CCA.
Multiple-Services-Credit-Control. Used-Service-Unit. CC-Time	R	Contains the amount of used time in the unit of SECONDS.
Event-Timestamp	O	Timestamp of the charge event..

CCA for Terminate Request

The following table provides the CCA needed for the Terminate Request.

Table 114 Credit Control Answer for Terminate Request

AVP	Type	Description
Session-Id	M	Session ID
Origin-Host	M	Origin Host
Origin-Realm	M	Origin Realm
Destination-Realm	M	Destination Realm
Auth-Application-Id	M	4
Service-Context-Id	M	Used to identify the service logic at the DCCS. For IMS SLE, set to 32260@3gpp.org
CC-Request-Type	M	TERMINATE_REQUEST
CC-Request-Number	M	Request Number
Event-Timestamp	O	Timestamp of the charge event
Cost-Information. Unit-Value. Value-Digits (Charged-Amount)	O	This AVP is applicable for Advice of Charge. The digits portion of the Charged-Amount of the session from the OCS is loaded into this AVP.
Cost-Information. Unit-Value. Exponent (Charged-Amount-Exponent)	O	This AVP is loaded with the Charged-Amount_ Exponent from the OCS.

Index