ETSI ES 204 915-12 V1.1.1 (2008-05)

ETSI Standard

Open Service Access (OSA);
Application Programming Interface (API);
Part 12: Charging SCF
(Parlay 6)

Reference
DES/TISPAN-01032-12-OSA

Keywords
API, IDL, OSA, UML

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Contents

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (http://webapp.etsi.org/IPR/home.asp).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Foreword

This ETSI Standard (ES) has been produced by ETSI Technical Committee Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN).

The present document is part 12 of a multi-part deliverable covering Open Service Access (OSA); Application Programming Interface (API), as identified below. The API specification (ES 204 915) is structured in the following parts:

Part 1: "Overview";

Part 2: "Common Data Definitions";

Part 3: "Framework";

Part 4: "Call Control";

Part 5: "User Interaction SCF";

Part 6: "Mobility SCF";

Part 7: "Terminal Capabilities SCF";

Part 8: "Data Session Control SCF";

Part 9: "Generic Messaging SCF";

Part 10: "Connectivity Manager SCF";

Part 11: "Account Management SCF";

Part 12: "Charging SCF";

Part 13: "Policy Management SCF";

Part 14: "Presence and Availability Management SCF";

Part 15: "Multi-Media Messaging SCF";

Part 16: "Service Broker SCF".

The present document has been defined jointly between ETSI, The Parlay Group (http://www.parlay.org) and the 3GPP, in co-operation with a number of JAIN™ Community (http://www.java.sun.com/products/jain) member companies.

The present document forms part of the Parlay 6.0 set of specifications.

The present document is equivalent to 3GPP TS 29.198-12 V7.0.0 (Release 7).

1 Scope

The present document is part 12 of the Stage 3 specification for an Application Programming Interface (API) for Open Service Access (OSA).

The OSA specifications define an architecture that enables application developers to make use of network functionality through an open standardised interface, i.e. the OSA APIs.

The present document specifies the Charging Service Capability Feature (SCF) aspects of the interface. All aspects of the Charging SCF are defined here, these being:

- Sequence Diagrams.
- Class Diagrams.
- Interface specification plus detailed method descriptions.
- State Transition diagrams.
- Data Definitions.
- IDL Description of the interfaces.
- WSDL Description of the interfaces.
- Reference to the Java™ API description of the interfaces.

The process by which this task is accomplished is through the use of object modelling techniques described by the Unified Modelling Language (UML).

2 References

The references listed in clause 2 of ES 204 915-1 contain provisions which, through reference in this text, constitute provisions of the present document.

ETSI ES 204 915-1: "Open Service Access (OSA); Application Programming Interface (API); Part 1: Overview (Parlay 6)".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in ES 204 915-1 apply.

3.2 Abbreviations

For the purposes of the present document, the abbreviations given in ES 204 915-1 apply.

4 Charging SCF

The following clauses describe each aspect of the Charging Service Capability Feature (SCF).

The order is as follows:

- The Sequence diagrams give the reader a practical idea of how each of the SCF is implemented.
- The Class relationships clause shows how each of the interfaces applicable to the SCF, relate to one another.
- The Interface specification clause describes in detail each of the interfaces shown within the Class diagram part.
- The State Transition Diagrams (STD) show the transition between states in the SCF. The states and transitions are well-defined; either methods specified in the Interface specification or events occurring in the underlying networks cause state transitions.
- The Data Definitions clause shows a detailed expansion of each of the data types associated with the methods within the classes. Note that some data types are used in other methods and classes and are therefore defined within the Common Data types part ES 204 915-2.

4.1 General requirements on support of methods

An implementation of this API which supports or implements a method described in the present document, shall support or implement the functionality described for that method, for at least one valid set of values for the parameters of that method.

Where a method is not supported by an implementation of a Service interface, the exception P_METHOD_NOT_SUPPORTED shall be returned to any call of that method.

Where a method is not supported by an implementation of an Application interface, a call to that method shall be possible, and no exception shall be returned.

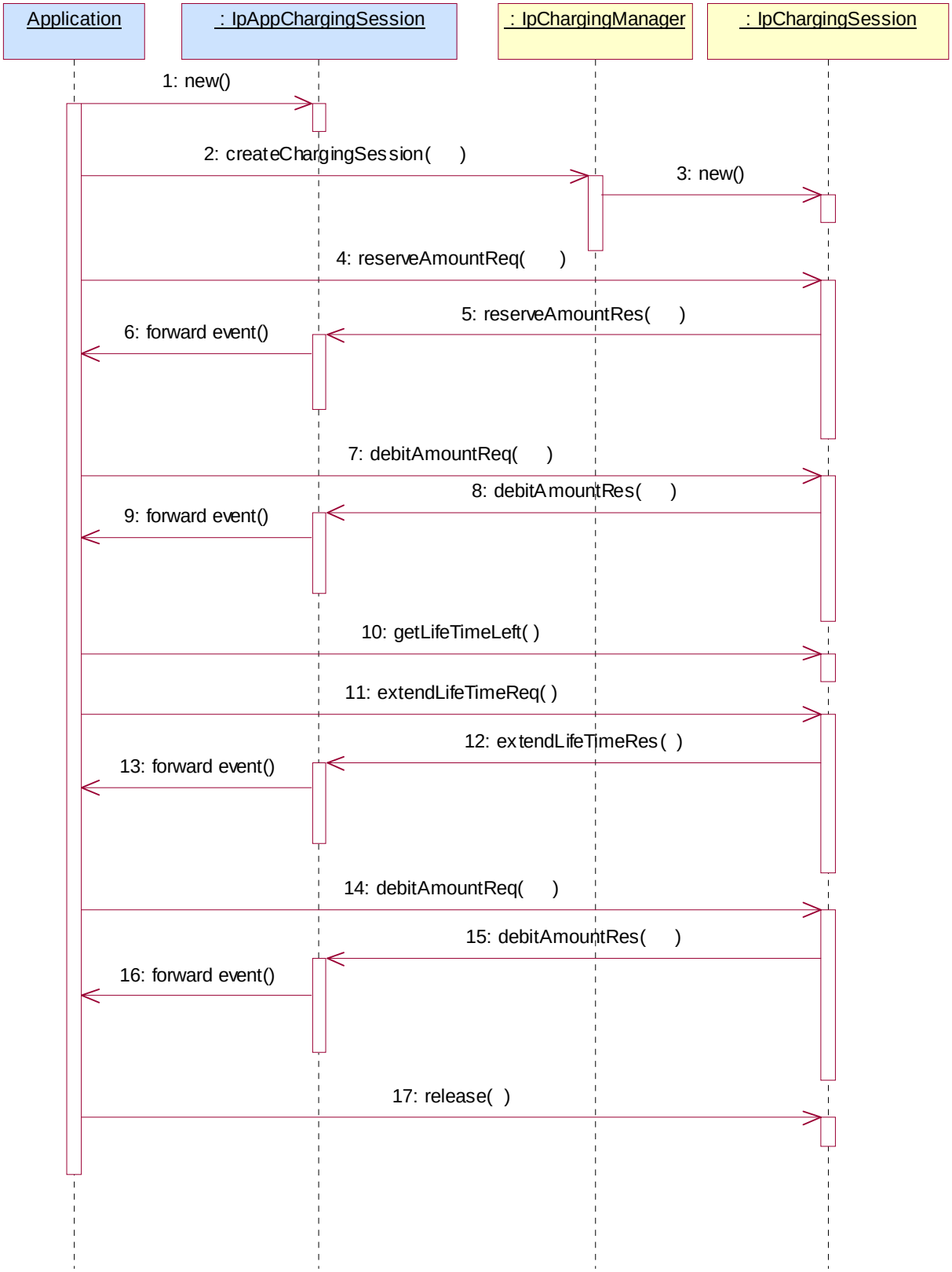
5 Sequence Diagrams

5.1 Reservation / payment in parts

The sequence diagram illustrates how to request a reservation and how to charge a user from the reserved amount, for instance to charge a user for a streamed video which lasts 10 minutes and costs a total of $2.00. The operations and interfaces that do not provide rating are employed throughout this sequence diagram.

We assume the application has already discovered the Charging SCF. As a result, the application received an object reference pointing to an object that implements the IpChargingManager interface.

The operations which handle units are used exactly the same, except that the amount of application usage is indicated instead of a price.



1: The application creates a local object implementing the IpAppChargingSession interface. This object will receive response messages from the IpChargingSession object.

2: The application opens a charging session, a reference to a new or existing object implementing IpChargingSession is returned together with a unique session ID.

3: In this case a new object is used.

4: The application requests the reservation of $2.00.

5: Assuming the criteria for requesting a reservation are met (the application provider has permission to charge the requested amount, the charged user has agreed to pay the requested amount), the amount is reserved in the session. At this point, the application provider knows that the network operator will accept later debit requests up to the reserved amount. So, the application may start serving the user, for instance by sending the video stream.

6: The successful reservation is reported back to the application.

After half of the video has been sent to the user, the application may choose to capture half of the price already:

7: The application requests to debit $1.00 from the reservation.

8: The successful debit is reported back to the application.

9: The acknowledge is forwarded to the application.

10: The application checks if the remaining lifetime of the reservation will cover the remaining 5 minutes of video. Let us assume, it does not.

11: The application asks the IpChargingSession object to extend the lifetime of the reservation.

12: Assuming that the application provider is allowed to keep reservations open for longer than 10 minutes, the extendLifeTimeReq() will be honoured and confirmed properly.

13: The confirmation is forwarded to the application.

14: When the complete video has been transmitted to the user without errors, the application charges another $1.00.

15: The IpChargingSession object acknowledges the successful debit at the IpAppChargingSession callback object.

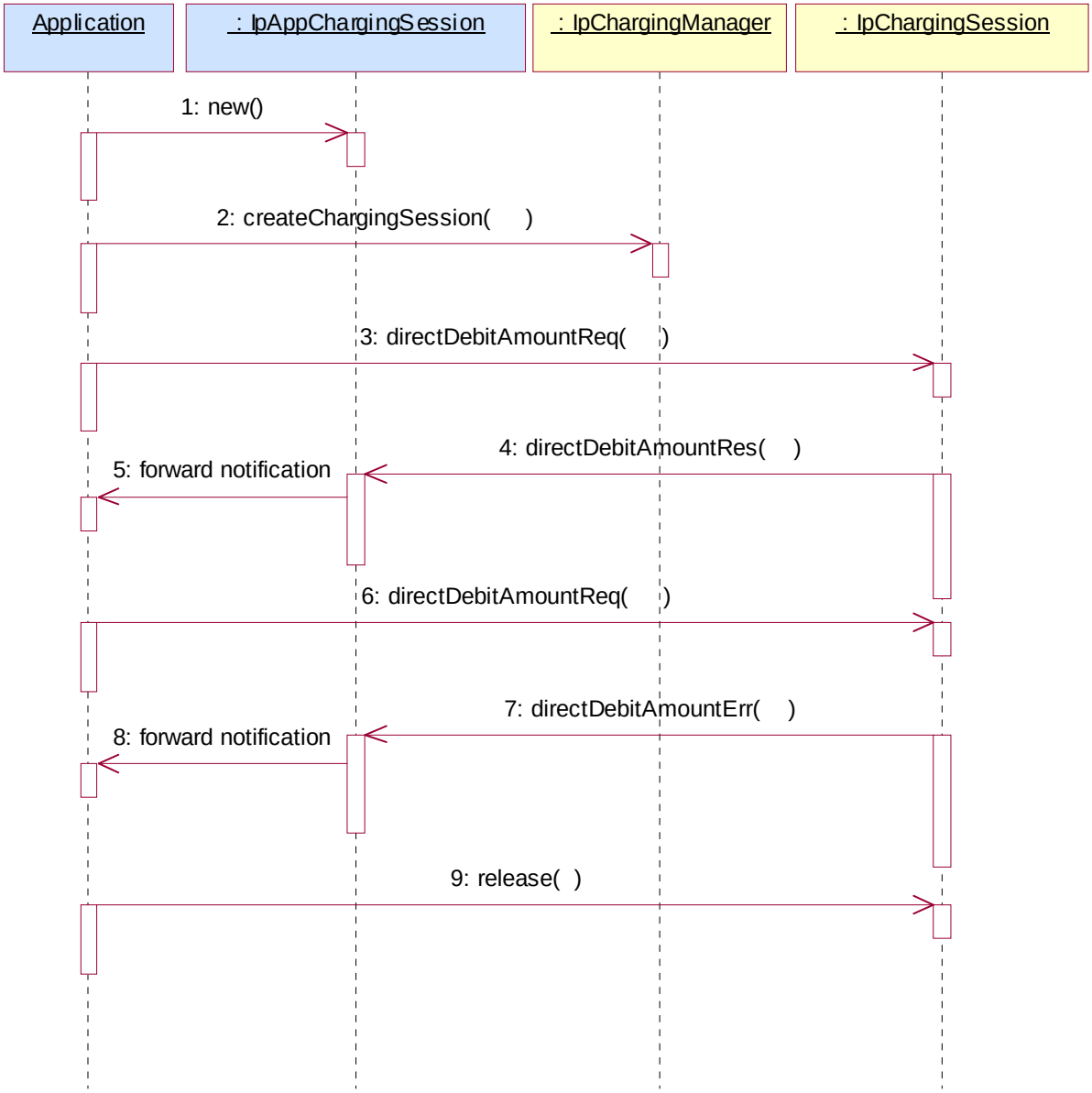
16: The IpAppChargingSession object forwards the acknowledge to the application.

17: Since the service is complete, the application frees all resources associated with the reservation and session.

5.2 Immediate Charge

This sequence diagram illustrates how immediate charging is used. Assume a WAP gateway that charges the user $0.01 per requested URL. Since it is acceptable to loose one tick worth $0.01, no prior reservations are made. The WAP gateway sends an immediate debit for each requested URL, and should a payment have as result failure, the user is disconnected.

The operations which handle units are used exactly the same, except that the amount of application usage is indicated instead of a price.



1: The application creates a local object implementing the IpAppChargingSession interface. This object will receive response messages from the IpChargingSession object.

2: The application orders the creation of a session. No new object is created for the charging session handling in this example implementation.

3: The application requests to charge the user $0.01.

4: The payment is acknowledged.

5: The acknowledgement is forwarded in the application.

6: The application requests to charge the user $0.01.

7: The payment is reported to fail.

8: The failure report is forwarded in the application.

(repeat steps 3 to 5 and 6 to 8 as long as you want to in any order you want to).

9: The application releases the session.

6 Class Diagrams

This class diagram shows the application interfaces for charging and their relations to the service interfaces.

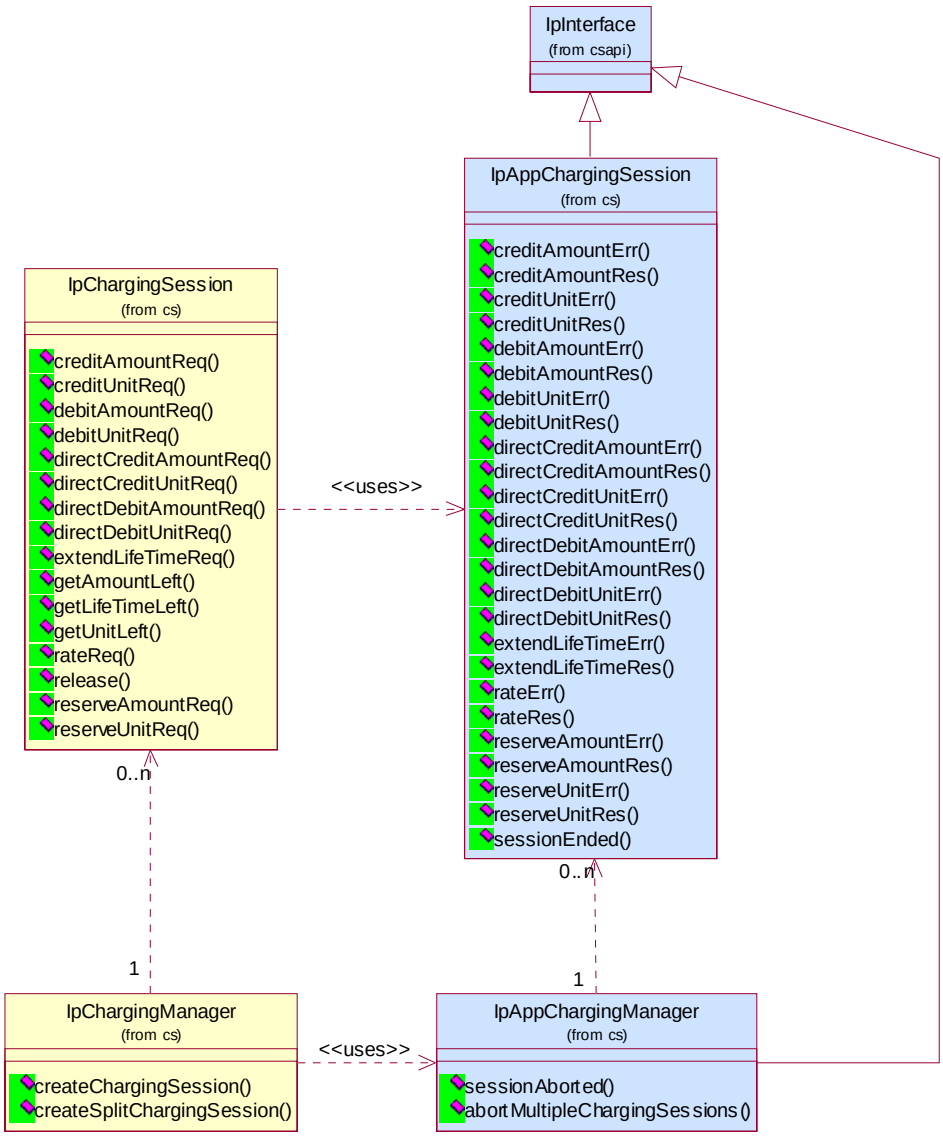


Figure 1: Application Interfaces

This class diagram shows the interfaces of the charging SCF.

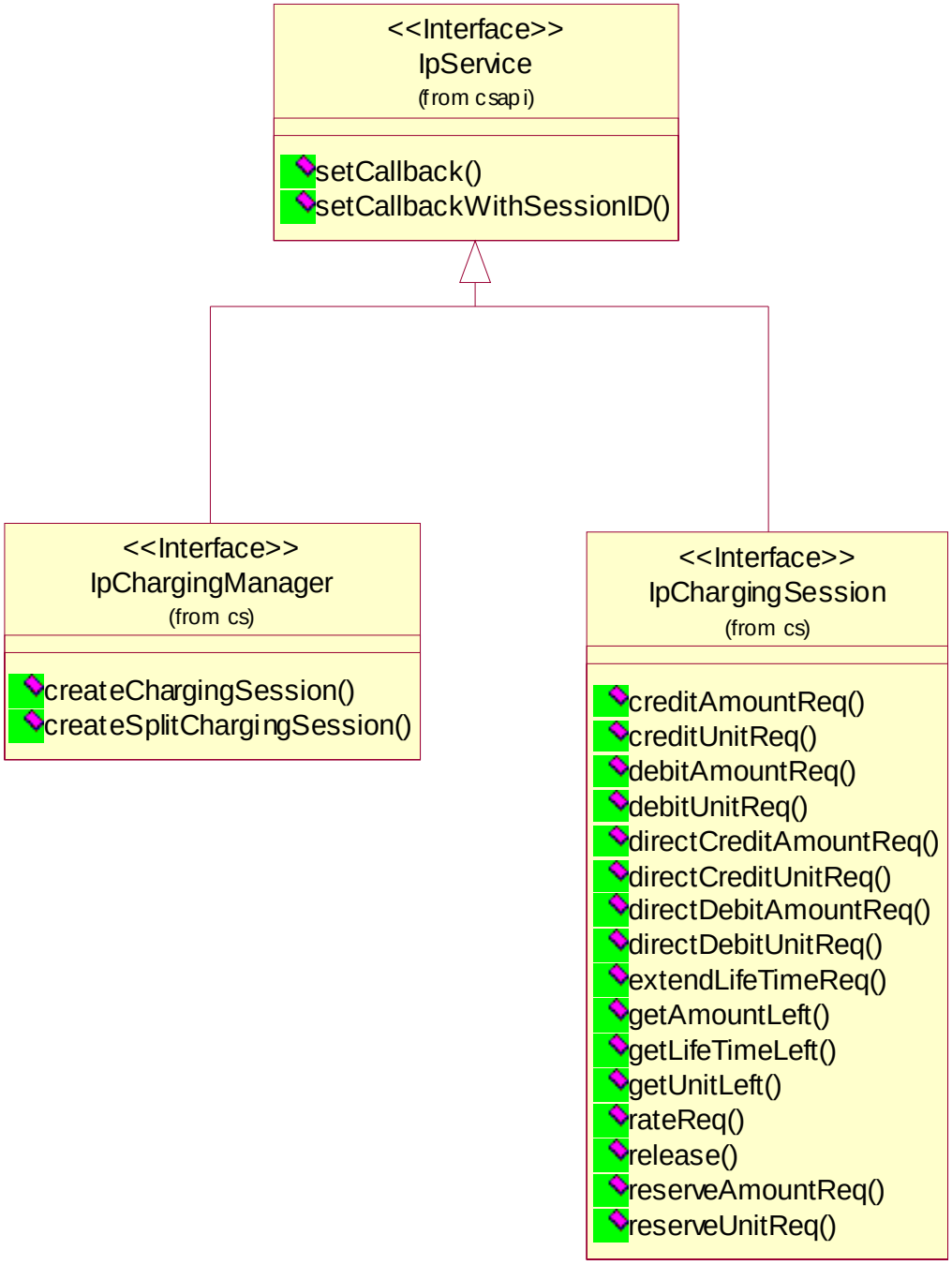


Figure 2: Service Interfaces

7 The Service Interface Specifications

7.1 Interface Specification Format

This clause defines the interfaces, methods and parameters that form a part of the API specification. The Unified Modelling Language (UML) is used to specify the interface classes. The general format of an interface specification is described below.

7.1.1 Interface Class

This shows a UML interface class description of the methods supported by that interface, and the relevant parameters and types. The Service and Framework interfaces for enterprise-based client applications are denoted by classes with name `Ip<name>`. The callback interfaces to the applications are denoted by classes with name `IpApp<name>`. For the interfaces between a Service and the Framework, the Service interfaces are typically denoted by classes with name IpSvc<name>, while the Framework interfaces are denoted by classes with name IpFw<name>.

7.1.2 Method descriptions

Each method (API method "call") is described. Both synchronous and asynchronous methods are used in the API. Asynchronous methods are identified by a '`Req`' suffix for a method request, and, if applicable, are served by asynchronous methods identified by either a '`Res`' or '`Err`' suffix for method results and errors, respectively. To handle responses and reports, the application or service developer must implement the relevant `IpApp<name> or IpSvc<name>` interfaces to provide the callback mechanism.

7.1.3 Parameter descriptions

Each method parameter and its possible values are described. Parameters described as 'in' represent those that must have a value when the method is called. Those described as 'out' are those that contain the return result of the method when the method returns.

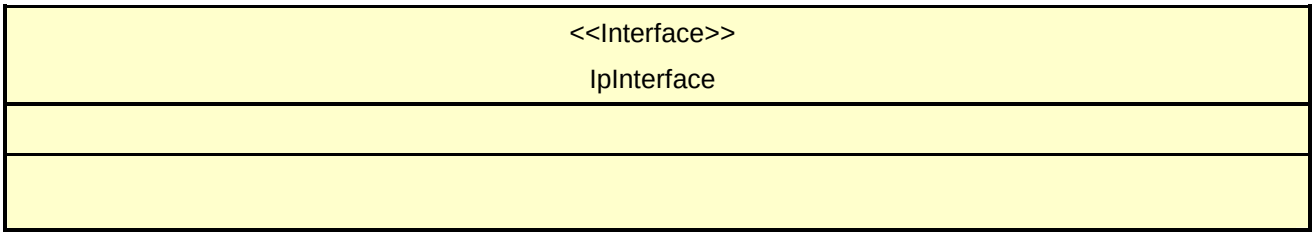
7.1.4 State Model

If relevant, a state model is shown to illustrate the states of the objects that implement the described interface.

7.2 Base Interface

7.2.1 Interface Class IpInterface

All application, framework and service interfaces inherit from the following interface. This API Base Interface does not provide any additional methods.



7.3 Service Interfaces

7.3.1 Overview

The Service Interfaces provide the interfaces into the capabilities of the underlying network - such as call control, user interaction, messaging, mobility and connectivity management.

The interfaces that are implemented by the services are denoted as 'Service Interface'. The corresponding interfaces that must be implemented by the application (e.g. for API callbacks) are denoted as 'Application Interface'.

7.4 Generic Service Interface

7.4.1 Interface Class IpService

Inherits from: IpInterface;

All service interfaces inherit from the following interface.

<<Interface>> IpService
setCallback (appInterface : in IpInterfaceRef) : void setCallbackWithSessionID (appInterface : in IpInterfaceRef, sessionID : in TpSessionID) : void

7.4.1.1 Method setCallback()

This method specifies the reference address of the callback interface that a service uses to invoke methods on the application. It is not allowed to invoke this method on an interface that uses SessionIDs. Multiple invocations of this method on an interface shall result in multiple callback references being specified. The SCS shall use the most recent callback interface provided by the application using this method. In the event that a callback reference fails or is no longer available, the next most recent callback reference available shall be used.

Parameters

appInterface : in IpInterfaceRef

Specifies a reference to the application interface, which is used for callbacks.

Raises

TpCommonExceptions, P_INVALID_INTERFACE_TYPE

7.4.1.2 Method setCallbackWithSessionID()

This method specifies the reference address of the application's callback interface that a service uses for interactions associated with a specific session ID: e.g. a specific call, or call leg. It is not allowed to invoke this method on an interface that does not use SessionIDs. Multiple invocations of this method on an interface shall result in multiple callback references being specified. The SCS shall use the most recent callback interface provided by the application using this method. In the event that a callback reference fails or is no longer available, the next most recent callback reference available shall be used.

Parameters

appInterface : in IpInterfaceRef

Specifies a reference to the application interface, which is used for callbacks.

sessionID : in TpSessionID

Specifies the session for which the service can invoke the application's callback interface.

Raises

TpCommonExceptions, P_INVALID_SESSION_ID, P_INVALID_INTERFACE_TYPE

8 Charging Interface Classes

The Charging SCF is used by applications to charge for the usage of the applications. The charged user can be the same user as that uses the application. It is also possible that another user will pay the charge.

In the interfaces of the Charging SCF a "Request Number" is used when invoking operations that operate on the user's account (directly or indirectly via reservations) in order to make retries possible after application, service, or communication errors. A retry of these operations can be done by invoking the same operation with the same Request Number.

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 can be used. This mechanism ensures that an application retries an operation when it does not receive an answer.

The use of the Request Number ensures that there can only be one outstanding request per Charging Session. Only after an answer is received (result or error), the next request can be made. Note however that only asynchronous operations that could lead to over or under charging of the user require a request number.

Because responses from the Charging SCF can be delayed in the network the Charging SCF shall guarantee that Request Numbers are unique in a timespan where delayed responses can arrive. Suppose, for example, that the response from a retried request is received indicating the next request number to use is 1 000. During the period that the response to the original request (which also carries the next request number to use equal to 1 000) can arrive, this request number may not be used again.

The units (of different types) that are used in a TpVolumeSet are NOT consolidated by the charging SCF. The application must use the same units when making the reservation and when debiting the amount. For example, when after a reservation of 10 minutes a debit request for 5 seconds is done, an error will be returned.

Split Charging Functionality.

There are cases where a single instance of the merchant application may serve more than a one service user. Examples are multi-user games or conferences. Typically, the costs for the resources consumed by the single service instance will be split among all service users.

On the other hand, a merchant application may show advertisements within its application, and in turn the company that is advertised may subside a certain percentage of the application cost. A consumer connecting to the merchant application pays only part of the costs, while the remainder is paid by the advertised company.

To support this kind of application, multiple users can be specified when a charging session is created. The charging session interface itself is the same no matter if the split charging feature is used or not.

It is subject to service level agreements that are negotiated between the OSA client provider and the network operator how the charge is split between the users.

8.1 Interface Class IpChargingManager

Inherits from: IpService.

This interface is the 'service manager' interface for the Charging Service. The Charging manager interface provides management functions to the charging service. The application programmer can use this interface to start charging sessions.

This interface shall be implemented by a Charging SCF. As a minimum requirement, at least one of createChargingSession() or createSplitChargingSession() shall be implemented.

<<Interface>> IpChargingManager
createChargingSession (appChargingSession : in IpAppChargingSessionRef, sessionDescription : in TpString, merchantAccount : in TpMerchantAccountID, user : in TpAddress, correlationID : in TpCorrelationID) : TpChargingSessionID createSplitChargingSession (appChargingSession : in IpAppChargingSessionRef, sessionDescription : in TpString, merchantAccount : in TpMerchantAccountID, users : in TpAddressSet, correlationID : in TpCorrelationID) : TpChargingSessionID

8.1.1 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 should already have been passed to the IpChargingManager, otherwise the charging manager will not be able to report a sessionAborted() to the application (the application should invoke setCallback() if it wishes to ensure this).

Returns chargingSession: Defines the session.

Parameters

appChargingSession : in IpAppChargingSessionRef

Callback interface for the session in the application.

sessionDescription : in TpString

Descriptive text for informational purposes.

merchantAccount : in TpMerchantAccountID

Identifies the account of the party providing the application to be used.

user : in TpAddress

Specifies the user that is using the application. This may or may not be the user that will be charged. The Charging service will determine the charged user. When this method is invoked the Charging service shall determine if charging is allowed for this application for this subscriber. An exception shall be thrown if this type of charging is not allowed.

correlationID : in TpCorrelationID

This value can be used to correlate the charging to network activity.

Returns

TpChargingSessionID

Raises

TpCommonExceptions, P_INVALID_USER, P_INVALID_ACCOUNT

8.1.2 Method createSplitChargingSession()

This method creates an instance of the IpChargingSession interface to handle the charging events related to the specified users and to the application invoking this method. This method differs from createChargingSession() in that it allows to specify multiple users to be charged. The SCS implementation is responsible to figure out how later reserve and charge operations are split among these subscribers. The algorithm may be selected and controlled e.g. through the chargingParameter argument in the respective methods. The algorithms provided and the details how they interpret any parameters are vendor specific.

Returns chargingSession: Defines the session.

Parameters

appChargingSession : in IpAppChargingSessionRef

Callback interface for the session in the application.

sessionDescription : in TpString

Descriptive text for informational purposes.

merchantAccount : in TpMerchantAccountID

Identifies the account of the party providing the application to be used.

users : in TpAddressSet

Specifies the users that are involved in using the application. This could be all users in a multi-party application (conference call, multi-user-game).

correlationID : in TpCorrelationID

This value can be used to correlate the charging to network activity.

Returns

TpChargingSessionID

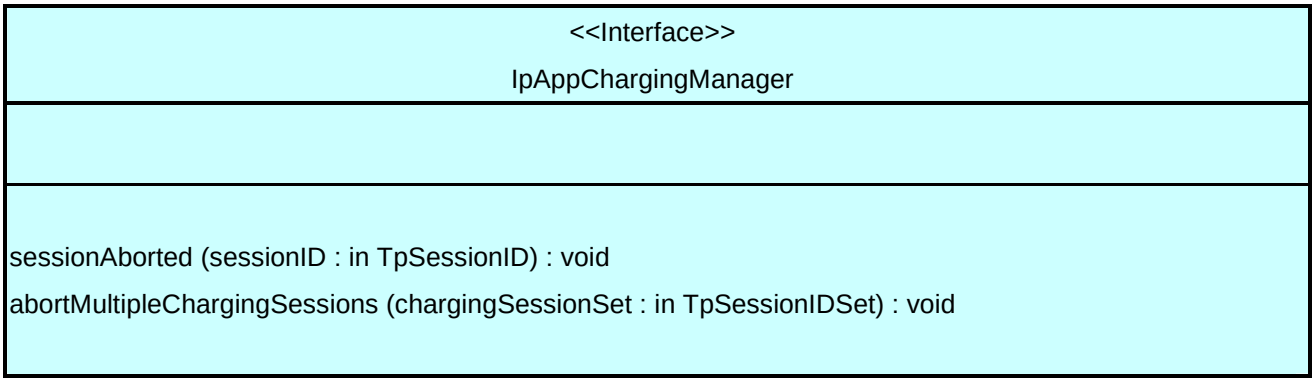
Raises

TpCommonExceptions, P_INVALID_USER, P_INVALID_ACCOUNT

8.2 Interface Class IpAppChargingManager

Inherits from: IpInterface.

This interface is the manager application interface for the Charging Service. The Charging manager interface provides the application Charging Session Management functions to the charging service.



8.2.1 Method sessionAborted()

This method indicates to the application that the charging session object (at the gateway) has aborted or terminated abnormally. No further communication will be possible between the charging session and application.

Parameters

`sessionID : in TpSessionID`

Specifies the sessionID of the charging session that has aborted or terminated abnormally.

8.2.2 Method abortMultipleChargingSessions()

The service may invoke this method on the IpAppChargingManager interface to indicate that a number of ongoing charging sessions have aborted or terminated abnormally. No further communication will be possible between the application and the charging sessions. This may be used for example in the event of service failure and recovery in order to instruct the application that a number of charging sessions have failed. The service shall provide a set of charging sessionIDs indicating to the application the charging sessions that have aborted. In the case that the service invokes this method and provides an empty set of sessionIDs, this shall be used to indicate that all charging sessions previously active on the IpChargingManager interface have been aborted.

Parameters

`chargingSessionSet : in TpSessionIDSet`

Specifies the set of sessionIDs of charging sessions that have aborted or terminated abnormally. The empty set shall be used to indicate that all charging sessions have aborted.

8.3 Interface Class IpChargingSession

Inherits from: IpService.

The Charging Session interface provides operations to facilitate transactions between a merchant and a user. The application programmer can use this interface to debit or credit amounts and/or units towards a user, to create and extend the lifetime of a reservation and to get information about what is left of the reservation.

This interface shall be implemented by a Charging SCF. As a minimum requirement, the release() method shall be implemented. If the reserveAmountReq() method is implemented, at least one of the debitAmountReq() or the creditAmountReq() methods shall also be implemented. If the reserveUnitReq() method is implemented, at least one of the debitUnitReq() or the creditUnitReq() methods shall also be implemented. If neither the reserveAmountReq() nor the reserveUnitReq() method is implemented, then at least one of the directDebitAmountReq() or the directDebitUnitReq(), or the directCreditAmountReq(), or the directCreditUnitReq() methods shall be implemented.

<<Interface>> IpChargingSession
creditAmountReq (sessionID : in TpSessionID, applicationDescription : in TpApplicationDescription, amount : in TpChargingPrice, closeReservation : in TpBoolean, requestNumber : in TpInt32) : void creditUnitReq (sessionID : in TpSessionID, applicationDescription : in TpApplicationDescription, volumes : in TpVolumeSet, closeReservation : in TpBoolean, requestNumber : in TpInt32) : void debitAmountReq (sessionID : in TpSessionID, applicationDescription : in TpApplicationDescription, amount : in TpChargingPrice, closeReservation : in TpBoolean, requestNumber : in TpInt32) : void debitUnitReq (sessionID : in TpSessionID, applicationDescription : in TpApplicationDescription, volumes : in TpVolumeSet, closeReservation : in TpBoolean, requestNumber : in TpInt32) : void directCreditAmountReq (sessionID : in TpSessionID, applicationDescription : in TpApplicationDescription, chargingParameters : in TpChargingParameterSet, amount : in TpChargingPrice, requestNumber : in TpInt32) : void directCreditUnitReq (sessionID : in TpSessionID, applicationDescription : in TpApplicationDescription, chargingParameters : in TpChargingParameterSet, volumes : in TpVolumeSet, requestNumber : in TpInt32) : void directDebitAmountReq (sessionID : in TpSessionID, applicationDescription : in TpApplicationDescription, chargingParameters : in TpChargingParameterSet, amount : in TpChargingPrice, requestNumber : in TpInt32) : void directDebitUnitReq (sessionID : in TpSessionID, applicationDescription : in TpApplicationDescription, chargingParameters : in TpChargingParameterSet, volumes : in TpVolumeSet, requestNumber : in TpInt32) : void extendLifeTimeReq (sessionID : in TpSessionID) : void getAmountLeft (sessionID : in TpSessionID) : TpChargingPrice getLifeTimeLeft (sessionID : in TpSessionID) : TpInt32 getUnitLeft (sessionID : in TpSessionID) : TpVolumeSet rateReq (sessionID : in TpSessionID, chargingParameters : in TpChargingParameterSet) : void release (sessionID : in TpSessionID, requestNumber : in TpInt32) : void reserveAmountReq (sessionID : in TpSessionID, applicationDescription : in TpApplicationDescription, chargingParameters : in TpChargingParameterSet, preferredAmount : in TpChargingPrice, minimumAmount : in TpChargingPrice, requestNumber : in TpInt32) : void reserveUnitReq (sessionID : in TpSessionID, applicationDescription : in TpApplicationDescription, chargingParameters : in TpChargingParameterSet, volumes : in TpVolumeSet, requestNumber : in TpInt32) : void

8.3.1 Method creditAmountReq()

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

The amount left in the reservation will be increased by this amount.

Each request to debit / credit an amount towards a reservation is handled separately. For example, two requests for a payment of EUR 1,- will give a total payment of EUR 2,-.

A credit of EUR 1,- and a debit of EUR 1 will give a total payment of EUR 0,-.

Parameters

sessionID : in TpSessionID

The ID of the session.

applicationDescription : in TpApplicationDescription

Descriptive text for informational purposes (e.g. text presented on the bill and used in communication towards the user).

amount : in TpChargingPrice

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

closeReservation : in TpBoolean

If set to true, this parameter indicates that the remaining part of the reservation can be freed. This may also mean addition of currency to the subscriber's account if more credits than debits have been made. The session is not released, this has to 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, P_INVALID_SESSION_ID, P_INVALID_AMOUNT, P_INVALID_CURRENCY, P_INVALID_REQUEST_NUMBER

8.3.2 Method creditUnitReq()

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

The volumes left in the reservation of this will be increased by this amount.

Each request to debit / credit a volume towards a reservation is handled separately. For example, two requests for a payment for 10 kilobytes will give a total payment for 20 kilobytes.

Parameters

sessionID : in TpSessionID

The ID of the session.

applicationDescription : in TpApplicationDescription

Descriptive text for informational purposes (e.g. text presented on the bill and used in communication towards the user).

volumes : in TpVolumeSet

Specifies the credited volumes in different units, more specifically 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 has to 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, P_INVALID_SESSION_ID, P_INVALID_VOLUME, P_INVALID_REQUEST_NUMBER

8.3.3 Method debitAmountReq()

This method debits an amount from the reservation.

The amount left in the reservation will be decreased by this amount.

Each request to debit / credit an amount towards a reservation is handled separately. For example, two requests for a payment of EUR 1,- will give a total payment of EUR 2,-.

A credit of EUR 1,- and a debit of EUR 1 will give a total payment of EUR 0,-.

When a debit operation would exceed the limit of the reservation, the debit operation fails.

Parameters

sessionID : in TpSessionID

The ID of the session.

applicationDescription : in TpApplicationDescription

Descriptive text for informational purposes (e.g. text presented on the bill and used in communication towards the user).

amount : in TpChargingPrice

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

closeReservation : in TpBoolean

If set to true, this parameter indicates that the reservation can be freed. The session is not released, this has to 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, P_INVALID_SESSION_ID, P_INVALID_AMOUNT, P_INVALID_CURRENCY, P_INVALID_REQUEST_NUMBER

8.3.4 Method debitUnitReq()

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

The volumes left in the reservation will be decreased by this amount.

Each request to debit / credit a volume towards a reservation is handled separately. For example, two requests for a payment for 10 kilobytes will give a total payment for 20 kilobytes.

When a debit operation would exceed the limit of the reservation, the debit operation succeeds, and the debited volumes will be the rest of the volumes in the reservation.

Parameters

sessionID : in TpSessionID

The ID of the session.

applicationDescription : in TpApplicationDescription

Descriptive text for informational purposes (e.g. text presented on the bill and used in communication towards the user).

volumes : in TpVolumeSet

Specifies the charged volumes in different units, more specifically 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 has to 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, P_INVALID_SESSION_ID, P_INVALID_VOLUME, P_INVALID_REQUEST_NUMBER

8.3.5 Method directCreditAmountReq()

This method directly credits an amount towards the user.

A possible reservation associated with this session is not influenced.

Parameters

sessionID : in TpSessionID

The ID of the session.

applicationDescription : in TpApplicationDescription

Descriptive text for informational purposes (e.g. text presented on the bill and used in communication towards 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.

amount : in TpChargingPrice

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

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, P_INVALID_SESSION_ID, P_INVALID_AMOUNT, P_INVALID_CURRENCY, P_INVALID_REQUEST_NUMBER

8.3.6 Method directCreditUnitReq()

This method directly credits a volume of application usage towards the user.

The volumes in a possible reservation associated with this session are not influenced.

Parameters

sessionID : in TpSessionID

The ID of the reservation.

applicationDescription : in TpApplicationDescription

Descriptive text for informational purposes (e.g. text presented on the bill and used in communication towards 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.

volumes : in TpVolumeSet

Specifies the credited volumes in different units, more specifically a sequence of data elements each containing the amount and applied unit.

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, P_INVALID_SESSION_ID, P_INVALID_VOLUME, P_INVALID_REQUEST_NUMBER

8.3.7 Method directDebitAmountReq()

This method directly debits an amount towards the user.

A possible reservation associated with this session is not influenced.

Parameters

sessionID : in TpSessionID

The ID of the session.

applicationDescription : in TpApplicationDescription

Descriptive text for informational purposes (e.g. text presented on the bill and used in communication towards 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.

amount : in TpChargingPrice

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

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, P_INVALID_SESSION_ID, P_INVALID_AMOUNT, P_INVALID_CURRENCY, P_INVALID_REQUEST_NUMBER

8.3.8 Method directDebitUnitReq()

This method directly credits a volume of application usage towards the user.

The volumes in a possible reservation associated with this session are not influence.

Parameters

sessionID : in TpSessionID

The ID of the reservation.

applicationDescription : in TpApplicationDescription

Descriptive text for informational purposes (e.g. text presented on the bill and used in communication towards 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.

volumes : in TpVolumeSet

Specifies the charged volumes in different units, more specifically a sequence of data elements each containing the amount and applied unit.

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, P_INVALID_SESSION_ID, P_INVALID_VOLUME, P_INVALID_REQUEST_NUMBER

8.3.9 Method extendLifeTimeReq()

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

Parameters

sessionID : in TpSessionID

The ID of the session.

Raises

TpCommonExceptions, P_INVALID_SESSION_ID

8.3.10 Method getAmountLeft()

With this method an application can request the remaining amount of the reservation.

Returns amountLeft: Gives the amount left in the reservation.

Parameters

sessionID : in TpSessionID

The ID of the session.

Returns

TpChargingPrice

Raises

TpCommonExceptions, P_INVALID_SESSION_ID

8.3.11 Method getLifeTimeLeft()

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 an exception (P_TASK_REFUSED).

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

Parameters

sessionID : in TpSessionID

The ID of the session.

Returns

TpInt32

Raises

TpCommonExceptions, P_INVALID_SESSION_ID

8.3.12 Method getUnitLeft()

With this method an application can request the remaining amount of the reservation.

Returns volumesLeft: Specifies the remaining volumes in different units, more specifically a sequence of data elements each containing the amount and applied unit.

Parameters

sessionID : in TpSessionID

The ID of the session.

Returns

TpVolumeSet

Raises

TpCommonExceptions, P_INVALID_SESSION_ID

8.3.13 Method rateReq()

This method is used when the application wants to have an item rated by the charging service. The result can be used to present pricing information to the end-user before the end-user actually wants to start using the service.

Parameters

sessionID : in TpSessionID

The ID of the session.

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.

Raises

TpCommonExceptions, P_INVALID_SESSION_ID

8.3.14 Method release()

This method releases the session, no operations can be done towards this session anymore (not even retries). Unused parts of a reservation are freed.

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.

Raises

TpCommonExceptions, P_INVALID_SESSION_ID, P_INVALID_REQUEST_NUMBER

8.3.15 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.

Parameters

sessionID : in TpSessionID

The ID of the session.

applicationDescription : in TpApplicationDescription

Descriptive text for informational purposes (e.g. text presented on the bill and used in communication towards 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.

preferredAmount : in TpChargingPrice

The amount of specified currency that the application wants to be reserved.

minimumAmount : in TpChargingPrice

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

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, P_INVALID_SESSION_ID, P_INVALID_AMOUNT, P_INVALID_CURRENCY, P_INVALID_REQUEST_NUMBER

8.3.16 Method reserveUnitReq()

This method is used when an application wants to reserve volumes of application usage to be delivered 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.

Parameters

sessionID : in TpSessionID

The ID of the session.

applicationDescription : in TpApplicationDescription

Descriptive text for informational purposes (e.g. text presented on the bill and used in communication towards 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.

volumes : in TpVolumeSet

Specifies the reserved volumes in different units, more specifically a sequence of data elements each containing the amount and applied unit. It is e.g. 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, P_INVALID_SESSION_ID, P_INVALID_VOLUME, P_INVALID_REQUEST_NUMBER

8.4 Interface Class IpAppChargingSession

Inherits from: IpInterface.

This application interface must be implemented by the client application to handle callbacks from the IpChargingSession.

<<Interface>> IpAppChargingSession
creditAmountErr (sessionID : in TpSessionID, requestNumber : in TpInt32, error : in TpChargingError, requestNumberNextRequest : in TpInt32) : void creditAmountRes (sessionID : in TpSessionID, requestNumber : in TpInt32, creditedAmount : in TpChargingPrice, reservedAmountLeft : in TpChargingPrice, requestNumberNextRequest : in TpInt32) : void creditUnitErr (sessionID : in TpSessionID, requestNumber : in TpInt32, error : in TpChargingError, requestNumberNextRequest : in TpInt32) : void creditUnitRes (sessionID : in TpSessionID, requestNumber : in TpInt32, creditedVolumes : in TpVolumeSet, reservedUnitsLeft : in TpVolumeSet, requestNumberNextRequest : in TpInt32) : void debitAmountErr (sessionID : in TpSessionID, requestNumber : in TpInt32, error : in TpChargingError,

requestNumberNextRequest : in TpInt32) : void

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

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

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

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

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

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

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

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

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

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

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

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

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

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

rateRes (sessionID : in TpSessionID, rates : in TpPriceVolumeSet, validityTimeLeft : in TpDuration) : void

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

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

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

reserveUnitRes (sessionID : in TpSessionID, requestNumber : in TpInt32, reservedUnits : in TpVolumeSet, sessionTimeLeft : in TpInt32, requestNumberNextRequest : in TpInt32) : void

sessionEnded (sessionID : in TpSessionID, report : in TpSessionEndedCause) : void

8.4.1 Method creditAmountErr()

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

Parameters

`sessionID : in TpSessionID`

This is the ID of the session for which the operation was called.

requestNumber : in TpInt32
This is the 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.

8.4.2 Method creditAmountRes()

This method indicates that the corresponding request was successful.

Parameters

sessionID : in TpSessionID
This is the ID of the session for which the operation was called.

requestNumber : in TpInt32
This is the 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.

8.4.3 Method creditUnitErr()

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

Parameters

sessionID : in TpSessionID
This is the ID of the session for which the operation was called.

requestNumber : in TpInt32
This is the request number for this request.

error : in TpChargingError
Indicates the reason for failure. Possible errors are: P_CHS_ERR_VOLUMES 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.

8.4.4 Method creditUnitRes()

This method indicates that the corresponding request was successful.

Parameters

sessionID : in TpSessionID

This is the ID of the session for which the operation was called.

requestNumber : in TpInt32

This is the 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.

8.4.5 Method debitAmountErr()

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

Parameters

sessionID : in TpSessionID

This is the ID of the session for which the operation was called.

requestNumber : in TpInt32

This is the 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.

8.4.6 Method debitAmountRes()

This method indicates that the corresponding request was successful.

Parameters

sessionID : in TpSessionID

This is the ID of the session for which the operation was called.

requestNumber : in TpInt32

This is the request number for this request.

debitedAmount : in TpChargingPrice

Indicates the debited 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.

8.4.7 Method debitUnitErr()

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

Parameters

sessionID : in TpSessionID

This is the ID of the session for which the operation was called.

requestNumber : in TpInt32

This is the 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.

8.4.8 Method debitUnitRes()

This method indicates that the corresponding request was successful.

Parameters

sessionID : in TpSessionID

This is the ID of the session for which the operation was called.

requestNumber : in TpInt32

This is the 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.

8.4.9 Method directCreditAmountErr()

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

Parameters

sessionID : in TpSessionID

This is the ID of the session for which the operation was called.

requestNumber : in TpInt32

This is the request number for this request.

error : in TpChargingError

Indicates the reason for failure. Possible errors are: P_CHS_ERR_PARAMETER, P_CHS_ERR_NO_CREDIT, P_CHS_ERR_CURRENCY.

requestNumberNextRequest : in TpInt32

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

8.4.10 Method directCreditAmountRes()

This method indicates that the corresponding request was successful.

Parameters

sessionID : in TpSessionID

This is the ID of the session for which the operation was called.

requestNumber : in TpInt32

This is the request number for this request.

creditedAmount : in TpChargingPrice

Indicates the credited amount.

requestNumberNextRequest : in TpInt32

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

8.4.11 Method directCreditUnitErr()

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

Parameters

sessionID : in TpSessionID

This is the ID of the session for which the operation was called.

requestNumber : in TpInt32

This is the request number for this request.

error : in TpChargingError

Indicates the reason for failure. Possible errors are: P_CHS_ERR_PARAMETER, P_CHS_ERR_NO_CREDIT, P_CHS_ERR_VOLUMES.

requestNumberNextRequest : in TpInt32

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

8.4.12 Method directCreditUnitRes()

This method indicates that the corresponding request was successful.

Parameters

sessionID : in TpSessionID

This is the ID of the session for which the operation was called.

requestNumber : in TpInt32

This is the request number for this request.

creditedVolumes : in TpVolumeSet

Indicates the credited volumes of application usage.

requestNumberNextRequest : in TpInt32

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

8.4.13 Method directDebitAmountErr()

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

Parameters

sessionID : in TpSessionID

This is the ID of the session for which the operation was called.

requestNumber : in TpInt32

This is the request number for this request.

error : in TpChargingError

Indicates the reason for failure. Possible errors are: P_CHS_ERR_PARAMETER, P_CHS_ERR_NO_DEBIT, P_CHS_ERR_CURRENCY, 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.

8.4.14 Method directDebitAmountRes()

This method indicates that the corresponding request was successful.

Parameters

sessionID : in TpSessionID

This is the ID of the session for which the operation was called.

requestNumber : in TpInt32

This is the request number for this request.

debitedAmount : in TpChargingPrice

Indicates the debited amount.

requestNumberNextRequest : in TpInt32

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

8.4.15 Method directDebitUnitErr()

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

Parameters

sessionID : in TpSessionID

This is the ID of the session for which the operation was called.

requestNumber : in TpInt32

This is the request number for this request.

error : in TpChargingError

Indicates the reason for failure. Possible errors are: P_CHS_ERR_PARAMETER, P_CHS_ERR_NO_DEBIT, P_CHS_ERR_VOLUMES, 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.

8.4.16 Method directDebitUnitRes()

This method indicates that the corresponding request was successful.

Parameters

sessionID : in TpSessionID

This is the ID of the session for which the operation was called.

requestNumber : in TpInt32

This is the 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.

8.4.17 Method extendLifeTimeErr()

This method indicates that the corresponding request failed.

Parameters

sessionID : in TpSessionID

This is the ID of the session for which the operation was called.

error : in TpChargingError

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

8.4.18 Method extendLifeTimeRes()

This method indicates that the corresponding request was successful.

Parameters

sessionID : in TpSessionID

This is the ID of the session for which the operation was called.

sessionTimeLeft : in TpInt32

Indicates the number of seconds that the session remains valid.

8.4.19 Method rateErr()

This method indicates that the corresponding request failed.

Parameters

sessionID : in TpSessionID

This is the ID of the session for which the operation was called.

error : in TpChargingError

Indicates the reason for failure. Possible errors are: P_CHS_ERR_PARAMETER.

8.4.20 Method rateRes()

This method indicates that the corresponding request was successful.

Parameters

sessionID : in TpSessionID

This is the ID of the session for which the operation was called.

rates : in TpPriceVolumeSet

The applicable rates.

validityTimeLeft : in TpDuration

Indicates the number of milliseconds that this information remains valid.

8.4.21 Method reserveAmountErr()

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

Parameters

sessionID : in TpSessionID

This is the same as the session ID returned in the request.

requestNumber : in TpInt32

This is the 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.

8.4.22 Method reserveAmountRes()

This method indicates that the corresponding request was successful.

Parameters

sessionID : in TpSessionID

This is the same as the session ID returned in the request.

requestNumber : in TpInt32

This is the 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 therein remain valid.

requestNumberNextRequest : in TpInt32

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

8.4.23 Method reserveUnitErr()

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

Parameters

sessionID : in TpSessionID

This is the same as the session ID returned in the request.

requestNumber : in TpInt32

This is the 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.

8.4.24 Method reserveUnitRes()

This method indicates that the corresponding request was successful.

Parameters

sessionID : in TpSessionID

This is the same as the session ID returned in the request.

requestNumber : in TpInt32

This is the 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. E.g. a pending reservation of 25 charging units and a new reservation of 1 000 octets and 10 charging units will result 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 therein remain valid.

requestNumberNextRequest : in TpInt32

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

8.4.25 Method sessionEnded()

This method indicates to the application that the charging session has terminated in the charging server. The application is expected to deassign the charging session object after having received the sessionEnded.

Parameters

sessionID : in TpSessionID

Specifies the charging sessionID.

report : in TpSessionEndedCause

Specifies the cause the charging session is terminated.

9 State Transition Diagrams

9.1 State Transition Diagrams for IpChargingSession

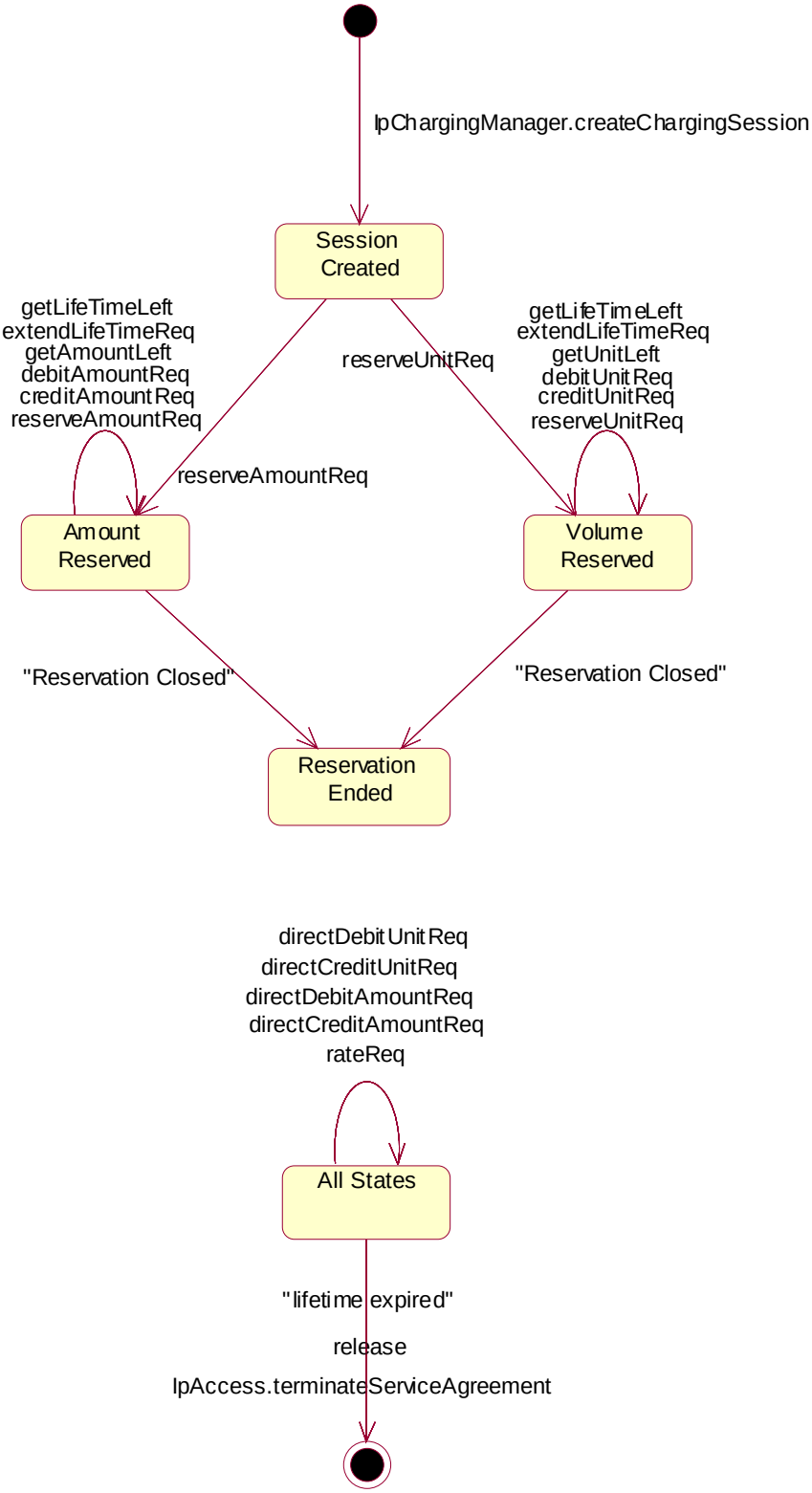


Figure 3: Charging Session Handling

9.1.1 Session Created State

In this state the Charging Session is created. No reservations have been made. In this state, the applications have the possibility to perform direct debits and credits on the user's account and to request rating.

9.1.2 Amount Reserved State

In this state a reservation for a certain maximum amount has been made. This reservation has succeeded and the application has the possibility to perform incremental debits/credits on this reserved amount until either the application chooses to close the reservation or the reservation limit is reached, or the Charging Session is released (either explicitly by the application or implicitly when the lifetime of the session has expired). The application can also extend the reservation and control its lifetime.

If the application chooses to close the reservation or the original reservation limit is reached, a transition to the 'Reservation Ended' state results.

9.1.3 Volume Reserved State

In this state a reservation for a certain maximum volume (kilobytes, emails, html-pages, etc) has been made. This reservation has succeeded and the application has the possibility to perform incremental debits/credits on this reserved volume until either the application chooses to close the reservation or the reservation limit is reached, or the Charging Session is released (either explicitly by the application or implicitly when the lifetime of the session has expired). The application can also extend the reservation and control its lifetime.

If the application chooses to close the reservation or the original reservation limit is reached, a transition to the 'Reservation Ended' state results.

9.1.4 Reservation Ended State

In this state an amount or volume reservation has been closed by the application, or the reservation limit has been reached. The charging session may remain active in order to carry out non-reservation related tasks such direct credit or debit operations. No further charging reservations shall be possible for this session, a new session is therefore required for future charging reservations. The charging session is closed on lifetime expiry or application 'release'.

10 Content Based Charging Service Properties

The following table lists properties relevant for CBC SCF.

Property	Type	Description/Interpretation
P_ADDRESSPLAN	INTEGER_SET	Indicates the supported address plan (defined in TpAddressPlan.) E.g. {P_ADDRESS_PLAN_E164, P_ADDRESS_PLAN_IP}).
P_SUPPORTED_UNITS	INTEGER_SET	Indicates the unit-types that are supported, e.g. {P_CHS_UNIT_OCTETS, P_CHS_UNIT_SECONDS}.
P_SUPPORTED_CURRENCIES	STRING_SET	Indicates the currency-types that are supported according to ISO 4217, e.g. {"EUR", "DEM", "NLG"}.
P_UNIT_CHARGING	BOOLEAN_SET	Indicates if charging based on units (rather then amounts) is supported. Value = TRUE: unit based charging is supported Value = FALSE: unit based charging is not supported If unit charging is supported or not or is selected or not does not tell anything about amount charging.

Property	Type	Description/Interpretation
P_AMOUNT_CHARGING	BOOLEAN_SET	Indicates if charging based on amounts (rather then units) is supported. Value = TRUE: amount based charging is supported Value = FALSE: amount based charging is not supported If amount charging is supported or not or is selected or not does not tell anything about unit charging.
P_SPLIT_CHARGING	BOOLEAN_SET	Indicates if split charging feature is available. Value = TRUE: split charging is supported Value = FALSE: split charging is not supported.
P_DEBITING	BOOLEAN_SET	Upon service registration, this property describes if the SCS supports debiting at all and if it can be turned off by the application. Upon service instantiation, it describes which mode(s) the client has selected.
P_CREDITING	BOOLEAN_SET	Upon service registration, this property describes if the SCS supports crediting at all and if it can be turned off by the application. Upon service instantiation, it describes which mode(s) the client has selected.

The previous table lists properties related to the capabilities of the SCS itself. The following table lists properties that are used in the context of the Service Level Agreement, e.g. to restrict the access of applications to the capabilities of the SCS.

Property	Type	Description/Interpretation
P_DEFAULT_LIFETIME	INTEGER_INTERVAL	Defines the default lifetime for a charging reservation in milliseconds.
P_LIFETIME_INCREMENT	INTEGER_INTERVAL	Defines the duration in milliseconds by which the lifetime of a charging reservation can be extended.
P_MAX_LIFETIME	INTEGER_INTERVAL	Defines the maximum lifetime for a charging reservation in milliseconds.
P_MIN_DEBIT_AMOUNT	STRING_SET	Defines the minimum amounts for a debit operation, depending on currency. Each set element is a string that contains the amount, formatted as a string and followed by the currency. Example: {'1.00 EUR', '0.5 GBP'} means that the minimum amount in Euro is 1.00, while in Pound Sterling it is 0.5.
P_MAX_DEBIT_AMOUNT	STRING_SET	Defines the maximum amount for a debit operation, similar to P_MIN_DEBIT_AMOUNT.
P_CREDIT_AMOUNT	INTEGER_INTERVAL	Defines the range for amounts that can be credited. Valid for any allowed currency.
P_PARALLEL_SESSIONS	INTEGER_INTERVAL	Defines the range for the allowed amount of parallel charging sessions.
P_SESSIONS_HOUR	INTEGER_INTERVAL	Defines the range for the allowed number of charging sessions per hour.

11 Data Definitions

11.1 Charging Data Definitions

This clause provides the Charging specific data definitions necessary to support the OSA interface specification.

The general format of a data definition specification is the following:

- Data type, that shows the name of the data type.

- Description, that describes the data type.
- Tabular specification, that specifies the data types and values of the data type.
- Example, if relevant, shown to illustrate the data type.

All data types referenced but not defined in this clause are common data definitions which may be found in ES 204 915-2.

11.1.1 IpChargingManager

Defines the address of an `IpChargingManager` Interface.

11.1.2 IpChargingManagerRef

Defines a Reference to type `IpChargingManager`.

11.1.3 IpAppChargingManager

Defines the address of an `IpAppChargingManager` Interface.

11.1.4 IpAppChargingManagerRef

Defines a Reference to type `IpAppChargingManager`.

11.1.5 IpChargingSession

Defines the address of an `IpChargingSession` Interface.

11.1.6 IpChargingSessionRef

Defines a Reference to type `IpChargingSession`.

11.1.7 IpAppChargingSession

Defines the address of an `IpAppChargingSession` Interface.

11.1.8 IpAppChargingSessionRef

Defines a Reference to type `IpAppChargingSession`.

11.1.9 TpApplicationDescription

Defines a Sequence of Data Elements that specifies what is about to be charged.

Sequence Element Name	Sequence Element Type
Text	TpString
AppInformation	TpAppInformationSet

11.1.10 TpAppInformationSet

Defines a Numbered Set of Data Elements that further describe what is about to be charged. The data elements are of type TpAppInformation.

11.1.11 TpAppInformation

Defines a Tagged Choice of Data Elements that comprise an individual application information.

	Tag Element Type	
	TpAppInformationType	

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

11.1.12 TpAppInformationType

Defines the possible information items.

Name	Value	Description
P_APP_INF_TIMESTAMP	0	The information item contains a timestamp.

11.1.13 TpSessionEndedCause

Defines the reason for which a charging session is released.

Name	Value	Description
P_CHS_CAUSE_UNDEFINED	0	The reason of release is not known, because no info was received from the network.
P_CHS_CAUSE_TIMER_EXPIRED	1	The session lifetime has expired.

11.1.14 TpMerchantAccountID

Defines a `Sequence of Data Elements` that defines the used service.

Sequence Element Name	Sequence Element Type
MerchantID	TpString
AccountID	TpInt32

11.1.15 TpCorrelationID

Defines the `Sequence of Data Elements` that identify a correlation.

Sequence Element Name	Sequence Element Type
CorrelationID	TpSessionID
CorrelationType	TpCorrelationType

11.1.16 TpCorrelationType

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

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

11.1.17 TpChargingPrice

Defines the `Sequence of Data Elements` that identify a price.

Sequence Element Name	Sequence Element Type
Currency	TpString
Amount	TpAmount
NOTE: Currencies as defined by ISO 4217.	

11.1.18 TpAmount

Defines the `Sequence of Data Elements` that define an amount in integers as "Number × 10 ^ Exponent" (i.e. if Number = 6 543 and Exponent = -2 then the amount is 65,43). This representation avoids unwanted rounding off.

Sequence Element Name	Sequence Element Type
Number	TpInt32
Exponent	TpInt32

11.1.19 TpChargingParameterSet

Defines a `Numbered Set of Data Elements` of TpChargingParameter.

11.1.20 TpChargingParameter

Defines a `Sequence of Data Elements` that defines the used service.

Sequence Element Name	Sequence Element Type
ParameterID	TpChargingParameterID
ParameterValue	TpChargingParameterValue

11.1.21 TpChargingParameterID

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

Name	Value	Description
P_CHS_PARAM_UNDEFINED	0	Unknown parameter
P_CHS_PARAM_ITEM	1	Parameter represents kind of service delivered to the end user
P_CHS_PARAM_SUBTYPE	2	Parameter represents subtype / operation of service delivered to the end user
P_CHS_PARAM_CONFIRMATION_ID	3	The ID that references a stored confirmation to authorize the required payment
P_CHS_PARAM_CONTRACT	4	Parameter represents a signed confirmation, which shall be of the type P_CHS_PARAMETER_OCTETSET

11.1.22 TpChargingParameterValue

Defines the `Tagged Choice of Data Elements` that identify a charging parameter.

	Tag Element Type	
	TpChargingParameterValueType	

Tag Element Value	Choice Element Type	Choice Element Name
P_CHS_PARAMETER_INT32	TpInt32	IntValue
P_CHS_PARAMETER_FLOAT	TpFloat	FloatValue
P_CHS_PARAMETER_STRING	TpString	StringValue
P_CHS_PARAMETER_BOOLEAN	TpBoolean	BooleanValue
P_CHS_PARAMETER_OCTETSET	TpOctetSet	OctetValue

11.1.23 TpChargingParameterValueType

Defines the type of charging parameter.

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

11.1.24 TpVolumeSet

Defines the `Numbered Set of Data Elements` that describes list TpVolume.

11.1.25 TpVolume

Defines a volume.

Sequence Element Name	Sequence Element Type
Amount	TpAmount
Unit	TpUnitID

11.1.26 TpUnitID

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

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

11.1.27 TpChargingSessionID

Defines the `Sequence of Data Elements` that unambiguously specify the Charging Session object.

Sequence Element Name	Sequence Element Type	Sequence Element Description
ChargingSessionReference	IpChargingSessionRef	This element specifies the interface reference for the charging session object.
ChargingSessionID	TpSessionID	This element specifies the session ID for the charging session.
RequestNumberFirstRequest	TpInt32	This element specifies the request number to use for the next request.

11.1.28 TpPriceVolumeSet

Defines a `Numbered Set of Data Elements` of TpPriceVolume.

11.1.29 TpPriceVolume

Defines the `Sequence of Data Elements` that identify a price for a volume.

Sequence Element Name	Sequence Element Type
Price	TpChargingPrice
Volume	TpVolume

11.1.30 TpChargingError

Indicates the error that occurred.

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	For some reason the application is not allowed to get money from this user.
P_CHS_ERR_NO_CREDIT	5	For some reason the application is not allowed 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 could not be obtained by the SCS. The SCS expects that the client initiates a stored confirmation scenario.

12 Exception Classes

The following are the list of exception classes, which are used in this interface of the API.

Name	Description
P_INVALID_ACCOUNT	Invalid merchant account specified.
P_INVALID_REQUEST_NUMBER	Invalid request number specified.
P_INVALID_USER	Invalid user specified.
P_INVALID_VOLUME	Invalid volume specified.

Each exception class contains the following structure:

Structure Element Name	Structure Element Type	Structure Element Description
ExtraInformation	TpString	Carries extra information to help identify the source of the exception, e.g. a parameter name

Annex A (normative): OMG IDL Description of Charging SCF

The OMG IDL representation of this interface specification is contained in a text file (cs.idl contained in archive es_20491512IDL.zip) which accompanies the present document.

This archive can be found in es_20491512v010101p0.zip which accompanies the present document.

Annex B (informative): W3C WSDL Description of Charging SCF

The W3C WSDL representation of this interface specification is contained in zip file es_20491512WSDL.zip which accompanies the present document.

This archive can be found in es_20491512v010101p0.zip which accompanies the present document.

Annex C (informative): Java™ API Description of the Charging SCF

The Java™ API realisation of this interface specification is produced in accordance with the Java™ Realisation rules defined in ES 204 915-1. These rules aim to deliver for Java™, a developer API, provided as a realisation, supporting a Java™ API that represents the UML specifications. The rules support the production of both J2SE™ and J2EE™ versions of the API from the common UML specifications.

The J2SE™ representation of this interface specification is provided as Java™ Code, contained in archive 20491512J2SE.zip that accompanies the present document.

The J2EE™ representation of this interface specification is provided as Java™ Code, contained in archive 20491512J2EE.zip that accompanies the present document.

Both these archives can be found in es_20491512v010101p0.zip which accompanies the present document.

Annex D (informative): Contents of 3GPP OSA R7 Charging

All of the present document is relevant for TS 129 198-12 V7 (Release 7).

Annex E (informative): Description of Charging for 3GPP2 cdma2000 networks

This annex is intended to define the OSA API Stage 3 interface definitions and it provides the complete OSA specifications. It is an extension of OSA API specifications capabilities to enable operation in cdma2000 systems environment. They are in alignment with 3GPP2 Stage 1 requirements and Stage 2 architecture defined in [52], [53] and [54] of EN 204 915-1, clause 2. These requirements are expressed as additions to and/or exclusions from the 3GPP Release 7 specification. The information given here is to be used by developers in 3GPP2 cdma2000 network architecture to interpret the 3GPP OSA specifications.

E.1 General Exceptions

The term UMTS is not applicable for the cdma2000 family of standards. Nevertheless these terms are used (TR 121 905) mostly in the broader sense of "3G Wireless System". If not stated otherwise there are no additions or exclusions required.

CAMEL and CAP mappings are not applicable for cdma2000 systems.

E.2 Specific Exceptions

E.2.1 Clause 1: Scope

There are no additions or exclusions.

E.2.2 Clause 2: References

Normative references on TS 123 078 and on TS 129 078 are not applicable for cdma2000 systems.

E.2.3 Clause 3: Definitions and abbreviations

There are no additions or exclusions.

E.2.4 Clause 4: Charging SCF

There are no additions or exclusions.

E.2.5 Clause 5: Sequence Diagrams

There are no additions or exclusions.

E.2.6 Clause 6: Class Diagrams

There are no additions or exclusions.

E.2.7 Clause 7: The Service Interface Specifications

There are no additions or exclusions.

E.2.8 Clause 8: Charging Interface Classes

There are no additions or exclusions.

E.2.9 Clause 9: State Transition Diagrams

There are no additions or exclusions.

E.2.10 Clause 10: Content Based Charging Service Properties

There are no additions or exclusions.

E.2.11 Clause 11: Data Definitions

There are no additions or exclusions.

E.2.12 Clause 12: Exception Classes

There are no additions or exclusions.

E.2.13 Annex A (normative): OMG IDL Description of Charging SCF

There are no additions or exclusions.

E.2.14 Annex B (informative): W3C WSDL Description of Charging SCF

There are no additions or exclusions.

E.2.15 Annex C (informative): Java™ API Description of the Charging SCF

There are no additions or exclusions.

Annex F (informative): Record of changes

The following is a list of the changes made to the present document for each release. The list contains the names of all changed, deprecated, added or removed items in the specifications and not the actual changes. Any type of change information that is important to the reader is put in the final clause of this annex.

Changes are specified as changes to the prior major release, but every minor release will have its own part of the table allowing the reader to know when the actual change was made.

F.1 Interfaces

F.1.1 New

Identifier	Comments
Interfaces added in ES 204 915-12 version 1.1.1 (Parlay 6.0)	

F.1.2 Deprecated

Identifier	Comments
Interfaces deprecated in ES 204 915-12 version 1.1.1 (Parlay 6.0)	

F.1.3 Removed

Identifier	Comments
Interfaces removed in ES 204 915-12 version 1.1.1 (Parlay 6.0)	

F.2 Methods

F.2.1 New

Identifier	Comments
Methods added in ES 204 915-12 version 1.1.1 (Parlay 6.0)	

F.2.2 Deprecated

Identifier	Comments
Methods deprecated in ES 204 915-12 version 1.1.1 (Parlay 6.0)	

F.2.3 Modified

Identifier	Comments
Methods modified in ES 204 915-12 version 1.1.1 (Parlay 6.0)	

F.2.4 Removed

Identifier	Comments
Methods removed in ES 204 915-12 version 1.1.1 (Parlay 6.0)	

F.3 Data Definitions

F.3.1 New

Identifier	Comments
Data Definitions added in ES 204 915-12 version 1.1.1 (Parlay 6.0)	

F.3.2 Modified

Identifier	Comments
Data Definitions modified in ES 204 915-12 version 1.1.1 (Parlay 6.0)	

F.3.3 Removed

Identifier	Comments
Data Definitions removed in ES 204 915-12 version 1.1.1 (Parlay 6.0)	

F.4 Service Properties

F.4.1 New

Identifier	Comments
Service Properties added in ES 204 915-12 version 1.1.1 (Parlay 6.0)	

F.4.2 Deprecated

Identifier	Comments
Service Properties deprecated in ES 204 915-12 version 1.1.1 (Parlay 6.0)	

F.4.3 Modified

Identifier	Comments
Service Properties modified in ES 204 915-12 version 1.1.1 (Parlay 6.0)	

F.4.4 Removed

Identifier	Comments
Service Properties removed in ES 204 915-12 version 1.1.1 (Parlay 6.0)	

F.5 Exceptions

F.5.1 New

Identifier	Comments
Exceptions added in ES 204 915-12 version 1.1.1 (Parlay 6.0)	

F.5.2 Modified

Identifier	Comments
Exceptions modified in ES 204 915-12 version 1.1.1 (Parlay 6.0)	

F.5.3 Removed

Identifier	Comments
Exceptions removed in ES 204 915-12 version 1.1.1 (Parlay 6.0)	

F.6 Others

None.

History

Document history		
V1.1.1	February 2008	Membership Approval Procedure MV 20080425: 2008-02-26 to 2008-04-25
V1.1.1	May 2008	Publication