Stockholm, Sweden, 13th - 15th June 2001

3GPP TS 29.198-12 V2.0.0 (2001-06)

Technical Specification

3rd Generation Partnership Project;
Technical Specification Group Core Network;
Open Service Access (OSA);
Application Programming Interface (API);
Part 12: Charging;
(Release 4)

Keywords

UMTS, API, OSA

Contents

Foreword

This Technical Specification has been produced by the 3rd Generation Partnership Project (3GPP).

The contents of the present document are subject to continuing work within the TSG and may change following formal TSG approval. Should the TSG modify the contents of the present document, it will be re-released by the TSG with an identifying change of release date and an increase in version number as follows:

Version x.y.z

where:

- x the first digit:
  - 1 presented to TSG for information;
  - 2 presented to TSG for approval;
  - 3 or greater indicates TSG approved document under change control.
- y the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.
- z the third digit is incremented when editorial only changes have been incorporated in the document.

Introduction

The present document is part 12 of a multi-part TS covering the 3rd Generation Partnership Project: Technical Specification Group Core Network; Open Service Access (OSA); Application Programming Interface (API), as identified below. The **API specification** (3GPP TS 29.198) is structured in the following Parts:

Part 1:	Overview	
Part 2:	Common Data Definitions	
Part 3:	Framework	
Part 4:	Call Control SCF	
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	(not part of 3GPP Release 4)
Part 10:	Connectivity Manager SCF	(not part of 3GPP Release 4)
Part 11:	Account Management SCF	
Part 12:	**Charging SCF**	

The **Mapping specification of the OSA APIs and network protocols** (3GPP TR 29.998) is also structured as above. A mapping to network protocols is however not applicable for all Parts, but the numbering of Parts is kept. Also in case a Part is not supported in a Release, the numbering of the parts is maintained.

OSA API specifications 29.198-family		OSA API Mapping - 29.998-family	
29.198-1	**Part 1: Overview**	**29.998-1**	**Part 1: Overview**
29.198-2	**Part 2: Common Data Definitions**	29.998-2	Not Applicable
29.198-3	**Part 3: Framework**	29.998-3	Not Applicable
29.198-4	**Part 4: Call Control SCF**	**29.998-4-1**	**Subpart 1: Generic Call Control – CAP mapping**
		29.998-4-2	
29.198-5	**Part 5: User Interaction SCF**	**29.998-5-1**	**Subpart 1: User Interaction – CAP mapping**
		29.998-5-2	
		29.998-5-3	
		29.998-5-4	**Subpart 4: User Interaction – SMS mapping**
29.198-6	**Part 6: Mobility SCF**	**29.998-6**	**User Status and User Location – MAP mapping**
29.198-7	**Part 7: Terminal Capabilities SCF**	29.998-7	Not Applicable
29.198-8	**Part 8: Data Session Control SCF**	**29.998-8**	**Data Session Control – CAP mapping**
29.198-9	Part 9: Generic Messaging SCF	29.998-9	Not Applicable
29.198-10	Part 10: Connectivity Manager SCF	29.998-10	Not Applicable
29.198-11	**Part 11: Account Management SCF**	29.998-11	Not Applicable
29.198-12	**Part 12: Charging SCF**	29.998-12	Not Applicable

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 concepts and the functional architecture for the OSA are contained in 3GPP TS 23.127 [3]. The requirements for OSA are contained in 3GPP TS 22.127 [2].

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

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

This specification has been defined jointly between 3GPP TSG CN WG5, ETSI SPAN 12 and the Parlay Consortium, in co-operation with the JAIN consortium.

2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document.*

[1] 3GPP TS 29.198-1 "Open Service Access; Application Programming Interface; Part 1: Overview".

[2] 3GPP TS 22.127: "Stage 1 Service Requirement for the Open Service Access (OSA) (Release 4)".

[3] 3GPP TS 23.127: "Virtual Home Environment (Release 4)".

[4] World Wide Web Consortium Composite Capability/Preference Profiles (CC/PP): A user side framework for content negotiation (www.w3.org).

[5] Wireless Application Protocol (WAP), Version 1.2, UAProf Specification (www.wapforum.org).

[6] ISO 4217:1995: "".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in TS 29.198-1 [1] apply.

3.2 Abbreviations

For the purposes of the present document, the abbreviations given in TS 29.198-1 [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 show 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 section show 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 of this specification.

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.

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.

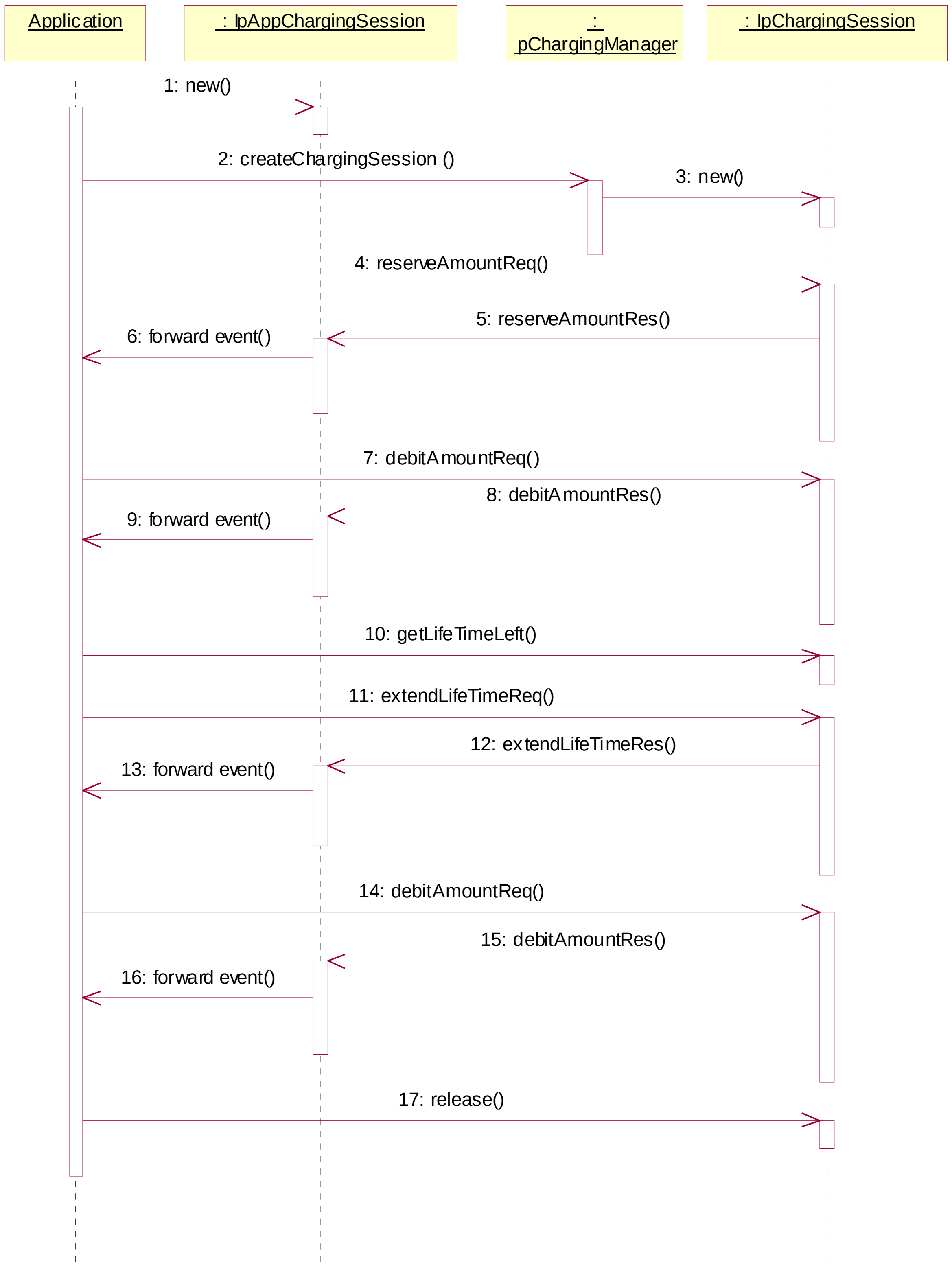
14. When the complete video has been transmitted to the user without errors, the application charges another $1.00.

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.

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


Application
: IpAppChargingSession
: pChargingManager
: IpChargingSession
1: new()
2: createChargingSession ()
3: new()
4: reserveAmountReq()
5: reserveAmountRes()
6: forward event()
7: debitAmountReq()
8: debitAmountRes()
9: forward event()
10: getLifeTimeLeft()
11: extendLifeTimeReq()
12: extendLifeTimeRes()
13: forward event()
14: debitAmountReq()
15: debitAmountRes()
16: forward event()
17: release()

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.

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

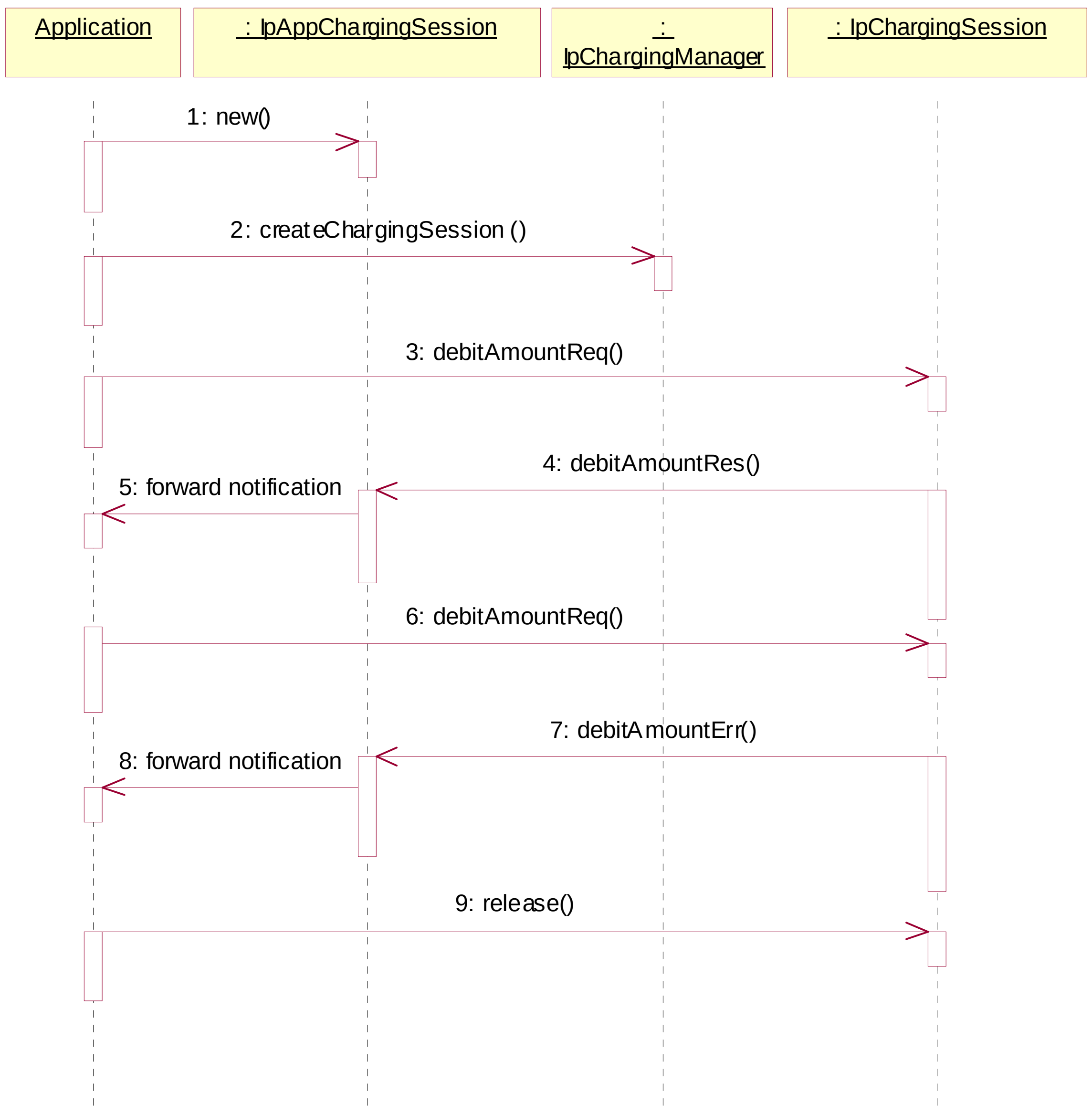
7. The payment is reported to fail.

8. The failure report is forwarded in the application.

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

9. The application releases the session

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



6 Class Diagrams

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

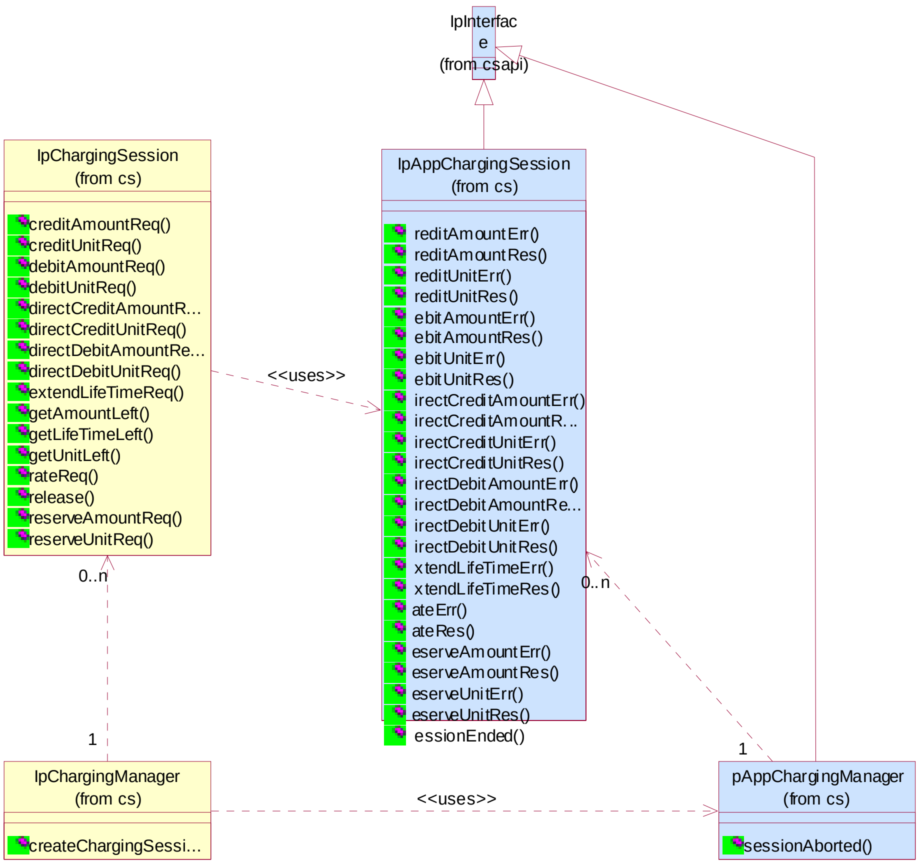


Figure: Application Interfaces

This class diagram shows the interfaces of the charging SCF.

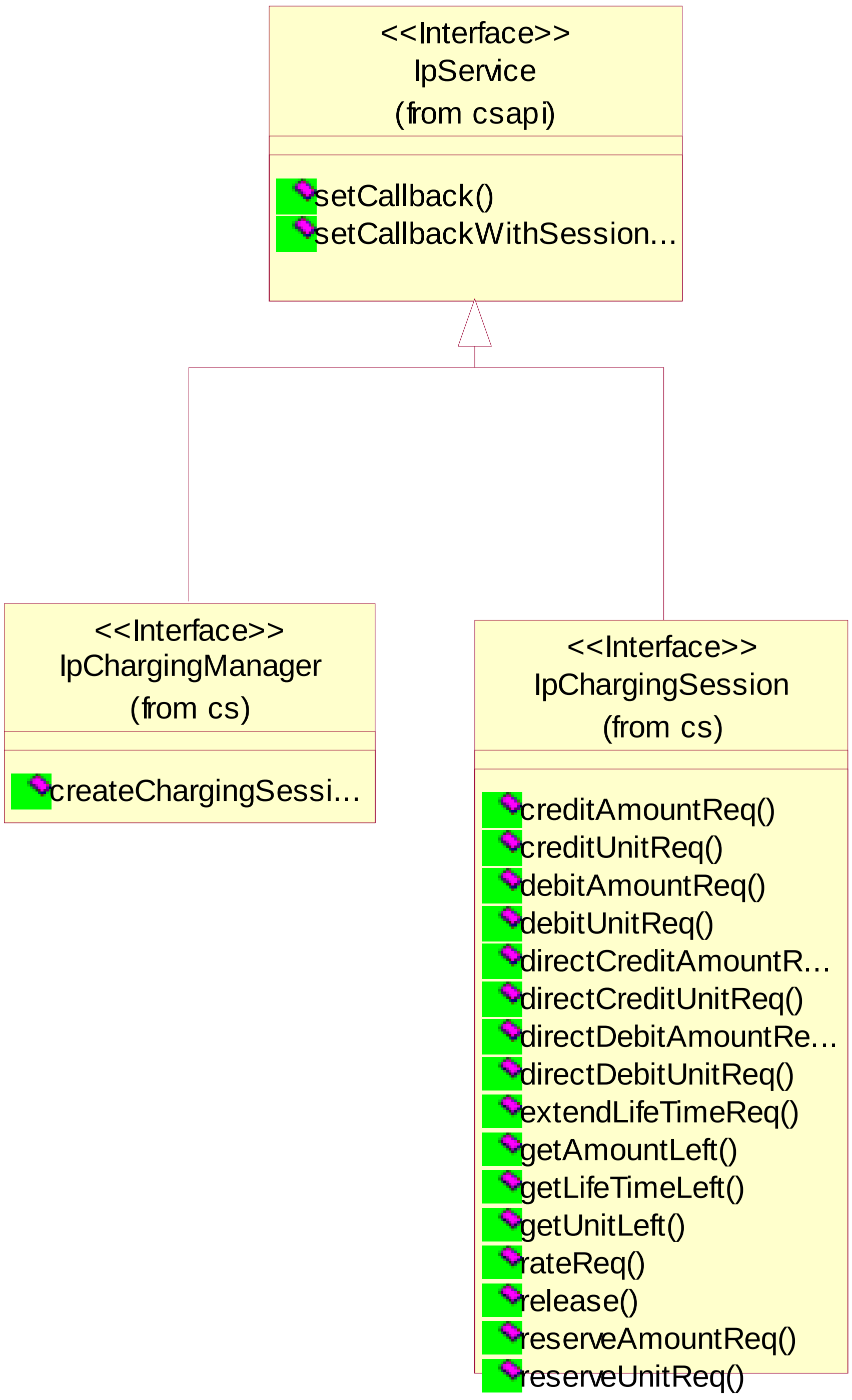


Figure: Service Interfaces

7 The Service Interface Specifications

7.1 Interface Specification Format

This section 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. All methods in the API return a value of type `TpResult`, indicating, amongst other things, if the method invocation was sucessfully executed or not.

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.

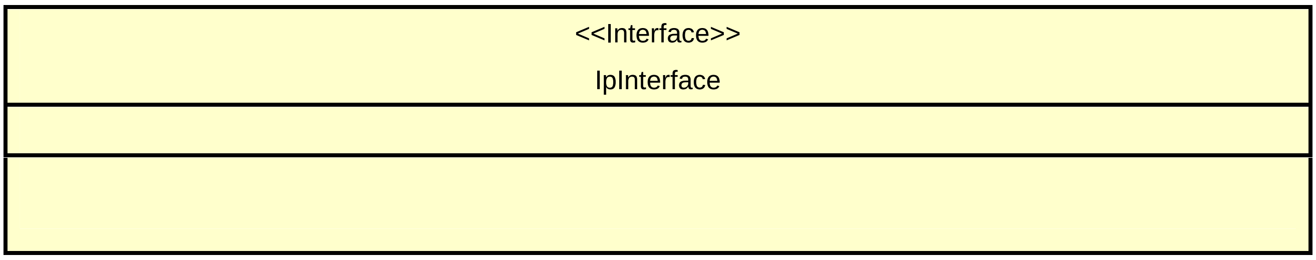
7.1.4 State Model

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.

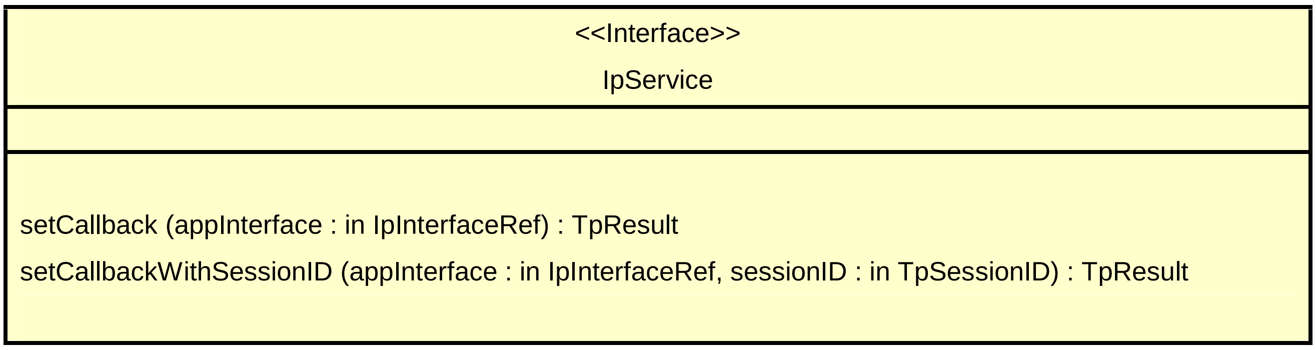
The interfaces that are implemented by the services are denoted as 'Service Interface'. The corresponding interfaces that must be implemented by the application (e.g. for API callbacks) are denoted as 'Application Interface'.

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) : TpResult
setCallbackWithSessionID (appInterface : in IpInterfaceRef, sessionID : in TpSessionID) : TpResult

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 SessionID's.

Parameters

`appInterface : in IpInterfaceRef`

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

Raises

`TpCommonExceptions`

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 uses SessionID's.

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`

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.

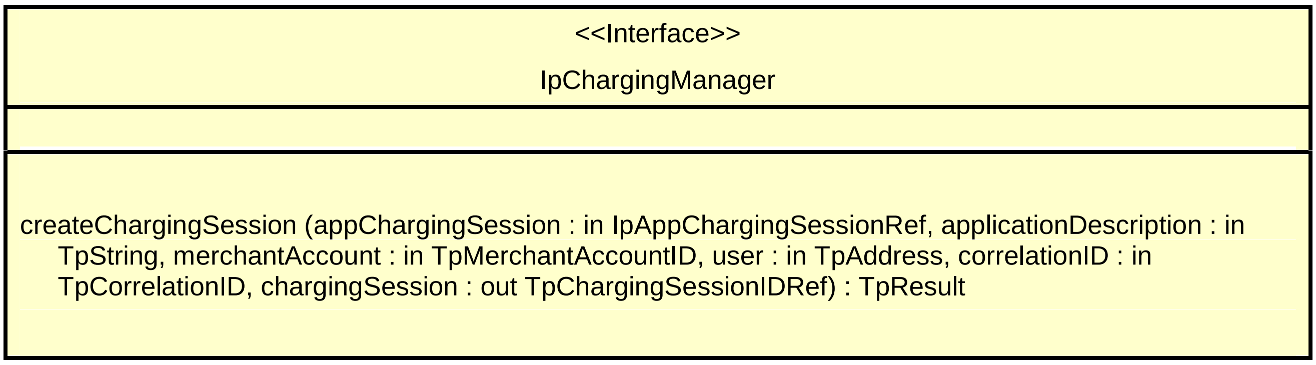
In the interfaces of the Charging SCF a "Request Number" is used when invoking operations that operate on the user's account (directly or indirectly via reservations) in order to make retries possible after application, service, or communication errors. A retry of these operations can be done by invoking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

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.



<<Interface>>

IpChargingManager

createChargingSession (appChargingSession : in IpAppChargingSessionRef, applicationDescription : in TpString, merchantAccount : in TpMerchantAccountID, user : in TpAddress, correlationID : in TpCorrelationID, chargingSession : out TpChargingSessionIDRef) : TpResult

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.

Parameters

`appChargingSession : in IpAppChargingSessionRef`

Callback interface for the session in the application

`applicationDescription : in TpString`

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

`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.

`chargingSession : out TpChargingSessionIDRef`

Defines the session.

Raises

`TpCommonExceptions, P_INVALID_USER, P_INVALID_ACCOUNT`

8.2 Interface Class IpAppChargingSession

Inherits from: IpInterface.

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

<<Interface>> IpAppChargingSession
extendLifeTimeRes (sessionID : in TpSessionID, SessionTimeLeft : in TpInt32) : TpResult extendLifeTimeErr (sessionID : in TpSessionID, error : in TpChargingError) : TpResult creditAmountRes (sessionID : in TpSessionID, creditedAmount : in TpChargingPrice, reservedAmountLeft : in TpChargingPrice, requestNumberNextRequest : in TpInt32) : TpResult creditAmountErr (sessionID : in TpSessionID, error : in TpChargingError, requestNumberNextRequest : in TpInt32) : TpResult debitAmountRes (sessionID : in TpSessionID, debitedAmount : in TpChargingPrice, reservedAmountLeft : in TpChargingPrice, requestNumberNextRequest : in TpInt32) : TpResult debitAmountErr (sessionID : in TpSessionID, error : in TpChargingError, requestNumberNextRequest : in

TpInt32) : TpResult

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

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

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

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

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

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

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

creditUnitErr (sessionID : in TpSessionID, error : in TpChargingError, requestNumberNextRequest : in TpInt32) : TpResult

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

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

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

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

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

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

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

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

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

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

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

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.

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

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.

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.

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.

error : in TpChargingError

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

requestNumberNextRequest : in TpInt32

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

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.

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.

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.

error : in TpChargingError

Indicates the reason for failure. Possible errors are: P_CHS_ERR_CURRENCY

requestNumberNextRequest : in TpInt32

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

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.

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 remains valid.

requestNumberNextRequest : in TpInt32

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

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.

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

requestNumberNextRequest : in TpInt32

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

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.

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.

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.

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.

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.

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.

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.

`error : in TpChargingError`

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

`requestNumberNextRequest : in TpInt32`

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

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.

`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.

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.

`error : in TpChargingError`

Indicates the reason for failure. Possible errors are: P_CHS_ERR_VOLUMES

`requestNumberNextRequest : in TpInt32`

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

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.

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.

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.

error : in TpChargingError

Indicates the reason for failure. Possible errors are: P_CHS_ERR_VOLUMES

requestNumberNextRequest : in TpInt32

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

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.

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 1000 octets and 10 charging units will result in two TpVolume elements for this parameter: 1000 octets and 35 charging units.

sessionTimeLeft : in TpInt32

Indicates the number of seconds that the session and the reservation therein remains valid.

requestNumberNextRequest : in TpInt32

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

Method
sessionEnded()

This method indicates to the application that the charging session has terminated in the charging server. The application is expected to de-assign 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.

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.

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

requestNumberNextRequest : in TpInt32

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

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 milli-seconds that this information remains valid.

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

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.

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.

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.

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.

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.

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.

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.

error : in TpChargingError

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

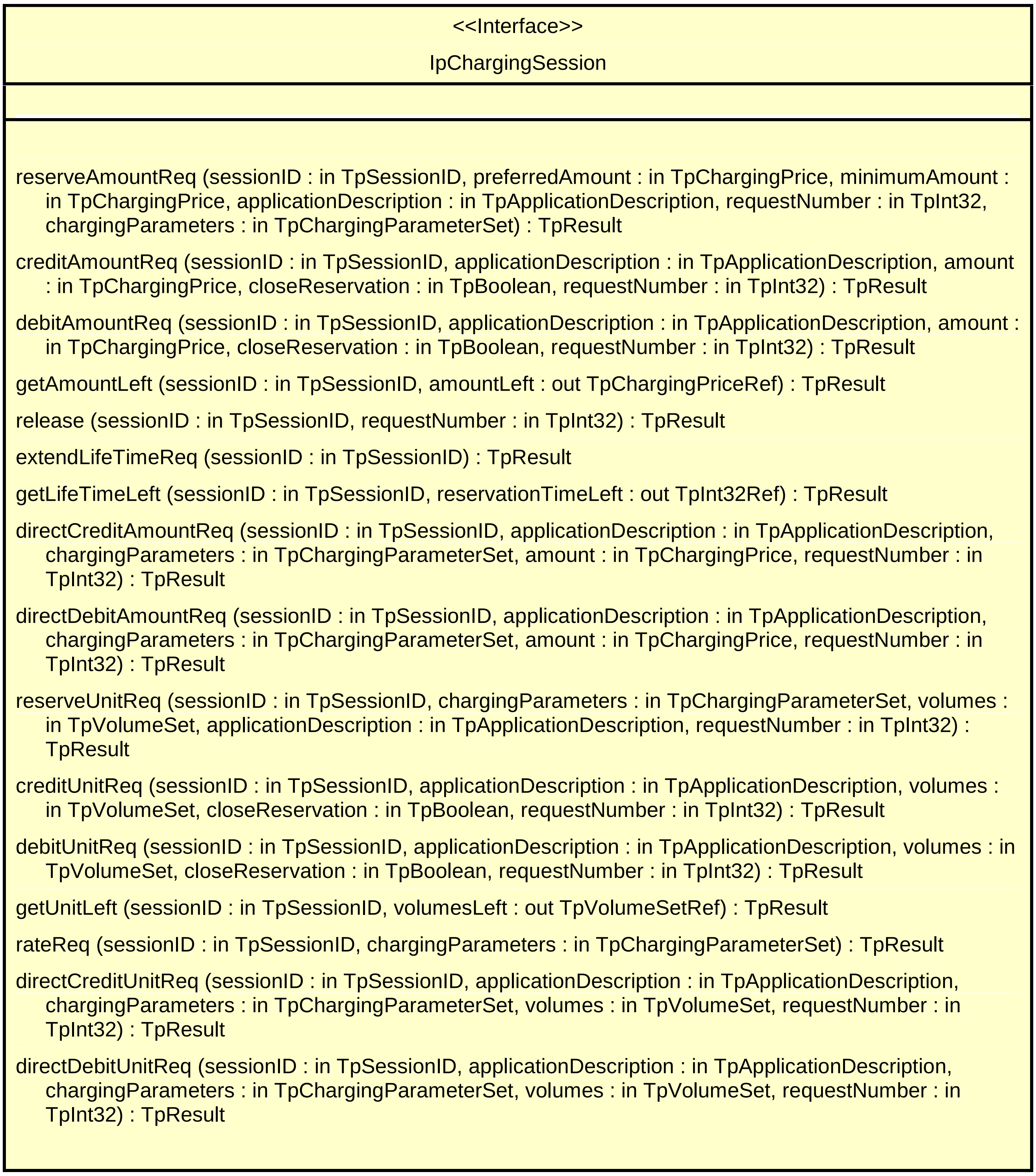
requestNumberNextRequest : in TpInt32

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

8.3 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.



<<Interface>>

IpChargingSession

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

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

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

getAmountLeft (sessionID : in TpSessionID, amountLeft : out TpChargingPriceRef) : TpResult

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

extendLifeTimeReq (sessionID : in TpSessionID) : TpResult

getLifeTimeLeft (sessionID : in TpSessionID, reservationTimeLeft : out TpInt32Ref) : TpResult

directCreditAmountReq (sessionID : in TpSessionID, applicationDescription : in TpApplicationDescription, chargingParameters : in TpChargingParameterSet, amount : in TpChargingPrice, requestNumber : in TpInt32) : TpResult

directDebitAmountReq (sessionID : in TpSessionID, applicationDescription : in TpApplicationDescription, chargingParameters : in TpChargingParameterSet, amount : in TpChargingPrice, requestNumber : in TpInt32) : TpResult

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

creditUnitReq (sessionID : in TpSessionID, applicationDescription : in TpApplicationDescription, volumes : in TpVolumeSet, closeReservation : in TpBoolean, requestNumber : in TpInt32) : TpResult

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

getUnitLeft (sessionID : in TpSessionID, volumesLeft : out TpVolumeSetRef) : TpResult

rateReq (sessionID : in TpSessionID, chargingParameters : in TpChargingParameterSet) : TpResult

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

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

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.

`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.

`applicationDescription : in TpApplicationDescription`

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

`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,P_INVALID_AMOUNT,P_INVALID_CURRENCY,P_INVALID_REQUEST_NUMBER`

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 and the session can be closed. This may also mean addition of currency to the subscriber's account if more credits than debits have been made.

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

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 and the session can be closed.

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

Method
getAmountLeft()

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

Parameters

`sessionID : in TpSessionID`

The ID of the session.

`amountLeft : out TpChargingPriceRef`

Gives the amount left in the reservation.

Raises

`TpCommonExceptions,P_INVALID_SESSION_ID`

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`

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`

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).

Parameters

sessionID : in TpSessionID

The ID of the session.

reservationTimeLeft : out TpInt32Ref

Indicates the number of seconds that the session remains valid.

Raises

TpCommonExceptions,P_INVALID_SESSION_ID

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

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

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.

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 10000 octets and 5 charging units.

applicationDescription : in TpApplicationDescription

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

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 and the session can be closed.

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

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 and the session can be closed.

`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`

Method

`getUnitLeft()`

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

Parameters

`sessionID : in TpSessionID`

The ID of the session.

`volumesLeft : out TpVolumeSetRef`

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

Raises

`TpCommonExceptions, P_INVALID_SESSION_ID`

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 want 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`

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`

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.

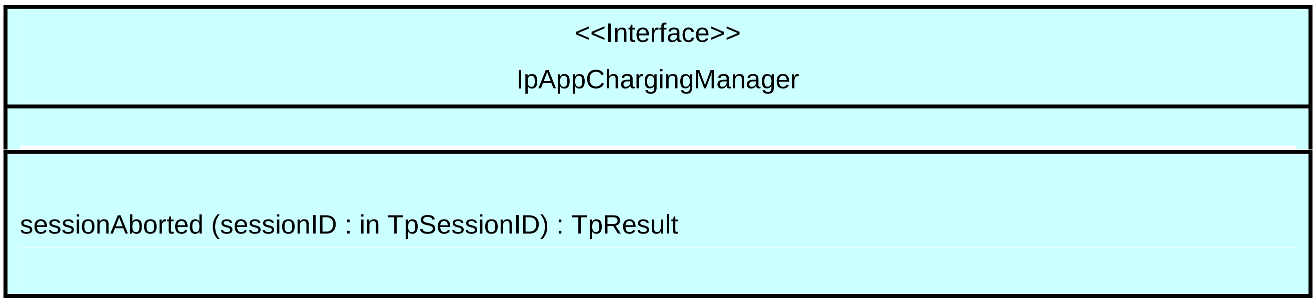
Raises

`TpCommonExceptions,P_INVALID_SESSION_ID,P_INVALID_VOLUME,P_INVALID_REQUEST_NUMBER`

8.4 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.



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.

9 State Transition Diagrams

9.1 State Transition Diagrams for IpChargingSession

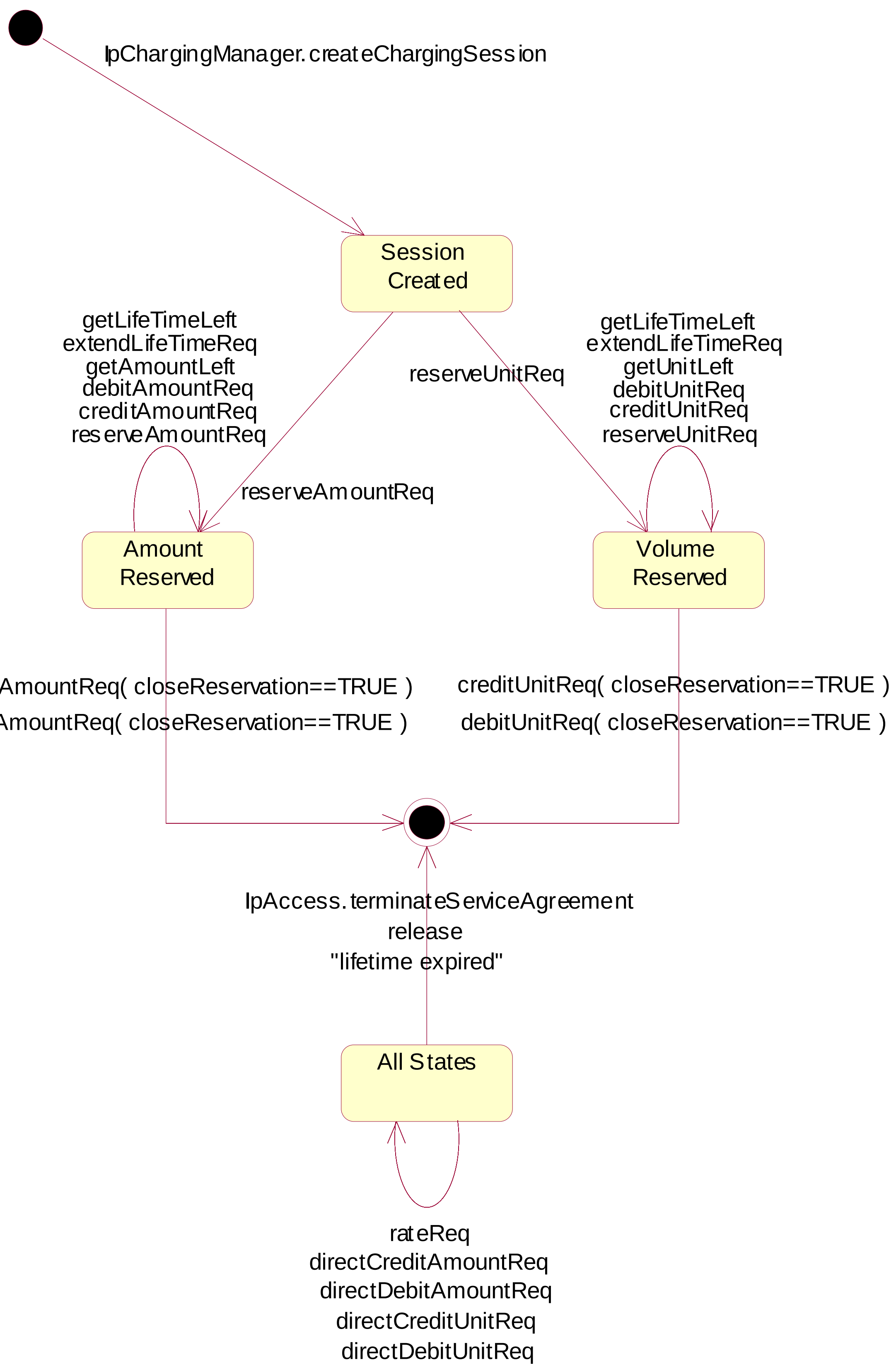

IpChargingManager.createChargingSession
Session Created
getLifeTimeLeft
extendLifeTimeReq
getAmountLeft
debitAmountReq
creditAmountReq
reserveAmountReq
reserveUnitReq
reserveAmountReq
getLifeTimeLeft
extendLifeTimeReq
getUnitLeft
debitUnitReq
creditUnitReq
reserveUnitReq
Amount Reserved
Volume Reserved
creditAmountReq(closeReservation==TRUE)
debitAmountReq(closeReservation==TRUE)
creditUnitReq(closeReservation==TRUE)
debitUnitReq(closeReservation==TRUE)
IpAccess.terminateServiceAgreement
release
"lifetime expired"
All States
rateReq
directCreditAmountReq
directDebitAmountReq
directCreditUnitReq
directDebitUnitReq

Figure : 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 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.

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 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.

10 Data Definitions

10.1 Charging Data Definitions

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

The present document is written using Hypertext link, to aid navigation through the data structures. Underlined text represents Hypertext links.

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.

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

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 comprise an individual application information.

	Tag Element Type	
	TpAppInformationType	

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

TpAppInformationType

Defines the possible information items.

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

TpSessionEndedCause

Defines the reason for which a charging session is released.

Name	Value	Description
P_CHS_CAUSE_UNDEFINED	0	The reason of release isn't known, because no info was received from the network.
P_CHS_CAUSE_TIMER_EXPIRED	1	The session life time has expired.

TpMerchantAccountID

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

Sequence Element Name	Sequence Element Type
MerchantID	TpString
AccountID	TpInt32

TpCorrelationID

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

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.

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

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 4217:1995 [6].	

TpAmount

Defines the `Sequence of Data Elements` that define an amount in integers as “Number * 10 ^ Exponent” (i.e. if Number = 6543 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

TpChargingParameterSet

Defines a `Numbered Set of Data Elements of` TpChargingParameter

TpChargingParameter

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

Sequence Element Name	Sequence Element Type
ParameterID	TpChargingParameterID
ParameterValue	TpChargingParameterValue

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_RESULT	3	Parameter represents the result of the service

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

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

TpVolumeSet

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

TpVolume

Defines a volume.

Sequence Element Name	Sequence Element Type
Amount	TpAmount
Unit	TpUnitID

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

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.

TpPriceVolumeSet

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

TpPriceVolume

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

Sequence Element Name	Sequence Element Type
Price	TpChargingPrice
Volume	TpVolume

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 liftetime of a reservation is rejected.
P_CHS_ERR_RESERVATION_LIMIT	9	This amount or volume violates the bounds of the reservation

11 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 2919812IDL.ZIP) which accompanies the present document.

Annex B (informative): Change history

Change history							
Date	**TSG #**	**TSG Doc.**	**CR**	**Rev**	**Subject/Comment**	**Old**	**New**
Mar 2001	CN_11	NP-010134	047	--	CR 29.198: for moving TS 29.198 from R99 to Rel 4 (N5-010158)	3.2.0	1.0.0
Jun 2001	CN_12	NP-010328	--	--		2.0.0	4.0.0

```
#ifndef __CS_DEFINED
#define __CS_DEFINED



#include "osa.idl"

module org {

      module csapi {



            module cs {
                  interface IpChargingSession;

                  struct TpMerchantAccountID {
                        TpString MerchantID;
                        TpInt32 AccountID;
                  };



                  typedef TpInt32 TpCorrelationType;



                  struct TpCorrelationID {
                        TpSessionID CorrelationID;
                        TpCorrelationType CorrelationType;
                  };


                  struct TpAmount {
                        TpInt32 Number;
                        TpInt32 Exponent;
                  };



                  struct TpChargingPrice {
                        TpString Currency;
                        TpAmount Amount;
                  };



                  typedef TpInt32 TpChargingParameterID;



                  enum TpChargingParameterValueType {
                        P_CHS_PARAMETER_INT32,
                        P_CHS_PARAMETER_FLOAT,
                        P_CHS_PARAMETER_STRING,
                        P_CHS_PARAMETER_BOOLEAN
                  };
                  union TpChargingParameterValue
switch(TpChargingParameterValueType) {
                        case P_CHS_PARAMETER_INT32: TpInt32 IntValue;
                        case P_CHS_PARAMETER_FLOAT: TpFloat FloatValue;
                        case P_CHS_PARAMETER_STRING: TpString StringValue;
                        case P_CHS_PARAMETER_BOOLEAN: TpBoolean BooleanValue;
                  };



                  typedef TpInt32 TpUnitID;



                  struct TpVolume {
                        TpUnitID Unit;
                        TpAmount Amount;
                  };



                  typedef sequence <TpVolume> TpVolumeSet;



                  struct TpChargingSessionID {
                        IpChargingSession ChargingSessionReference;
                        TpSessionID ChargingSessionID;
                        TpInt32 RequestNumberFirstRequest;
                  };



                  struct TpPriceVolume {
                        TpChargingPrice Price;
                        TpVolume Volume;
                  };



                  typedef sequence <TpPriceVolume> TpPriceVolumeSet;

                  enum TpChargingError {
                        P_CHS_ERR_UNDEFINED,
                        P_CHS_ERR_ACCOUNT,
                        P_CHS_ERR_USER,
                        P_CHS_ERR_PARAMETER,
                        P_CHS_ERR_NO_DEBIT,
                        P_CHS_ERR_NO_CREDIT,
                        P_CHS_ERR_VOLUMES,
                        P_CHS_ERR_CURRENCY,
                        P_CHS_ERR_NO_EXTEND,
                        P_CHS_ERR_RESERVATION_LIMIT
                  };



                  enum TpSessionEndedCause {
                        P_CHS_CAUSE_UNDEFINED,
                        P_CHS_CAUSE_TIMER_EXPIRED
                  };


                  enum TpAppInformationType {
                        P_APP_INF_TIMESTAMP
                  };


                  union TpAppInformation switch(TpAppInformationType) {
                        case P_APP_INF_TIMESTAMP: TpDateAndTime Timestamp;
                  };


                  typedef sequence <TpAppInformation> TpAppInformationSet;


                  struct TpApplicationDescription {
                        TpString Text;
                        TpAppInformationSet AppInformation;
                  };


                  struct TpChargingParameter {
                        TpChargingParameterID ParameterID;
                        TpChargingParameterValue ParameterValue;
                  };


                  typedef sequence <TpChargingParameter> TpChargingParameterSet;


                  const TpChargingParameterID P_CHS_PARAM_UNDEFINED = 0;

                  const TpChargingParameterID P_CHS_PARAM_ITEM = 1;

                  const TpChargingParameterID P_CHS_PARAM_SUBTYPE = 2;

                  const TpChargingParameterID P_CHS_PARAM_RESULT = 3;

                  const TpCorrelationType P_CHS_CORRELATION_UNDEFINED = 0;

                  const TpCorrelationType P_CHS_CORRELATION_VOICE = 1;

                  const TpCorrelationType P_CHS_CORRELATION_DATA = 2;

                  const TpCorrelationType P_CHS_CORRELATION_MM = 3;
                  const TpUnitID P_CHS_UNIT_UNDEFINED = 0;


                  const TpUnitID P_CHS_UNIT_NUMBER = 1;


                  const TpUnitID P_CHS_UNIT_OCTETS = 2;


                  const TpUnitID P_CHS_UNIT_SECONDS = 3;


                  const TpUnitID P_CHS_UNIT_MINUTES = 4;


                  const TpUnitID P_CHS_UNIT_HOURS = 5;


                  const TpUnitID P_CHS_UNIT_DAYS = 6;


                  exception P_INVALID_USER {
                        TpString extraInformation;
                  };



                  exception P_INVALID_ACCOUNT {
                        TpString extraInformation;
                  };



                  exception P_INVALID_REQUEST_NUMBER {
                        TpString extraInformation;
                  };



                  exception P_INVALID_VOLUME {
                        TpString extraInformation;
                  };




                  interface IpAppChargingSession : IpInterface {

                        void creditAmountErr (
                              in TpSessionID sessionID,
                              in TpChargingError error,
                              in TpInt32 requestNumberNextRequest
                              );


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


                        void creditUnitErr (
                              in TpSessionID sessionID,
                              in TpChargingError error,
                              in TpInt32 requestNumberNextRequest
                              );


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


                        void debitAmountErr (
                              in TpSessionID sessionID,
                              in TpChargingError error,
                              in TpInt32 requestNumberNextRequest
                              );


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


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


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


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


                        void directCreditAmountRes (
                              in TpSessionID sessionID,
                              in TpChargingPrice creditedAmount,
                              in TpInt32 requestNumberNextRequest
                              );
```

```
void directCreditUnitErr (
      in TpSessionID sessionID,
      in TpChargingError error,
      in TpInt32 requestNumberNextRequest
      );


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


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


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


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


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


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


void extendLifeTimeRes (
      in TpSessionID sessionID,
      in TpInt32 SessionTimeLeft
      );


void rateErr (
      in TpSessionID sessionID,
      in TpChargingError error
      );
```

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


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


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


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


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


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

                  };



                  interface IpChargingSession : IpService {

                        void creditAmountReq (
                              in TpSessionID sessionID,
                              in TpApplicationDescription
applicationDescription,
                              in TpChargingPrice amount,
                              in TpBoolean closeReservation,
                              in TpInt32 requestNumber
                              )
                              raises
(TpCommonExceptions,P_INVALID_SESSION_ID,P_INVALID_AMOUNT,P_INVALID_CURRENCY,P_I
NVALID_REQUEST_NUMBER);
```

```
                        void creditUnitReq (
                              in TpSessionID sessionID,
                              in TpApplicationDescription
applicationDescription,
                              in TpVolumeSet volumes,
                              in TpBoolean closeReservation,
                              in TpInt32 requestNumber
                              )
                              raises
(TpCommonExceptions,P_INVALID_SESSION_ID,P_INVALID_VOLUME,P_INVALID_REQUEST_NUMB
ER);


                        void debitAmountReq (
                              in TpSessionID sessionID,
                              in TpApplicationDescription
applicationDescription,
                              in TpChargingPrice amount,
                              in TpBoolean closeReservation,
                              in TpInt32 requestNumber
                              )
                              raises
(TpCommonExceptions,P_INVALID_SESSION_ID,P_INVALID_AMOUNT,P_INVALID_CURRENCY,P_I
NVALID_REQUEST_NUMBER);


                        void debitUnitReq (
                              in TpSessionID sessionID,
                              in TpApplicationDescription
applicationDescription,
                              in TpVolumeSet volumes,
                              in TpBoolean closeReservation,
                              in TpInt32 requestNumber
                              )
                              raises
(TpCommonExceptions,P_INVALID_SESSION_ID,P_INVALID_VOLUME,P_INVALID_REQUEST_NUMB
ER);


                        void directCreditAmountReq (
                              in TpSessionID sessionID,
                              in TpApplicationDescription
applicationDescription,
                              in TpChargingParameterSet chargingParameters,
                              in TpChargingPrice amount,
                              in TpInt32 requestNumber
                              )
                              raises
(TpCommonExceptions,P_INVALID_SESSION_ID,P_INVALID_AMOUNT,P_INVALID_CURRENCY,P_I
NVALID_REQUEST_NUMBER);


                        void directCreditUnitReq (
                              in TpSessionID sessionID,
                              in TpApplicationDescription
applicationDescription,
                              in TpChargingParameterSet chargingParameters,
                              in TpVolumeSet volumes,
                              in TpInt32 requestNumber
                              )
                              raises
(TpCommonExceptions,P_INVALID_SESSION_ID,P_INVALID_VOLUME,P_INVALID_REQUEST_NUMB
ER);


                        void directDebitAmountReq (
                              in TpSessionID sessionID,
                              in TpApplicationDescription
applicationDescription,
                              in TpChargingParameterSet chargingParameters,
                              in TpChargingPrice amount,
                              in TpInt32 requestNumber
                              )
                              raises
(TpCommonExceptions,P_INVALID_SESSION_ID,P_INVALID_AMOUNT,P_INVALID_CURRENCY,P_I
NVALID_REQUEST_NUMBER);


                        void directDebitUnitReq (
                              in TpSessionID sessionID,
                              in TpApplicationDescription
applicationDescription,
                              in TpChargingParameterSet chargingParameters,
                              in TpVolumeSet volumes,
                              in TpInt32 requestNumber
                              )
                              raises
(TpCommonExceptions,P_INVALID_SESSION_ID,P_INVALID_VOLUME,P_INVALID_REQUEST_NUMB
ER);


                        void extendLifeTimeReq (
                              in TpSessionID sessionID
                              )
                              raises (TpCommonExceptions,P_INVALID_SESSION_ID);


                        void getAmountLeft (
                              in TpSessionID sessionID,
                              out TpChargingPrice amountLeft
                              )
                              raises (TpCommonExceptions,P_INVALID_SESSION_ID);


                        void getLifeTimeLeft (
                              in TpSessionID sessionID,
                              out TpInt32 reservationTimeLeft
                              )
                              raises (TpCommonExceptions,P_INVALID_SESSION_ID);


                        void getUnitLeft (
                              in TpSessionID sessionID,
                              out TpVolumeSet volumesLeft
                              )
                              raises (TpCommonExceptions, P_INVALID_SESSION_ID);


                        void rateReq (
                              in TpSessionID sessionID,
                              in TpChargingParameterSet chargingParameters
```

```
                                )
                                raises (TpCommonExceptions,P_INVALID_SESSION_ID);


                        void release (
                                in TpSessionID sessionID,
                                in TpInt32 requestNumber
                                )
                                raises
(TpCommonExceptions,P_INVALID_SESSION_ID,P_INVALID_REQUEST_NUMBER);


                        void reserveAmountReq (
                                in TpSessionID sessionID,
                                in TpChargingPrice preferredAmount,
                                in TpChargingPrice minimumAmount,
                                in TpApplicationDescription
applicationDescription,
                                in TpInt32 requestNumber,
                                in TpChargingParameterSet chargingParameters
                                )
                                raises
(TpCommonExceptions,P_INVALID_SESSION_ID,P_INVALID_AMOUNT,P_INVALID_CURRENCY,P_I
NVALID_REQUEST_NUMBER);


                        void reserveUnitReq (
                                in TpSessionID sessionID,
                                in TpChargingParameterSet chargingParameters,
                                in TpVolumeSet volumes,
                                in TpApplicationDescription
applicationDescription,
                                in TpInt32 requestNumber
                                )
                                raises
(TpCommonExceptions,P_INVALID_SESSION_ID,P_INVALID_VOLUME,P_INVALID_REQUEST_NUMB
ER);

                  };



                  interface IpAppChargingManager : IpInterface {

                        void sessionAborted (
                                in TpSessionID sessionID
                                );

                  };



                  interface IpChargingManager : IpService {

                        void createChargingSession (
                                in IpAppChargingSession appChargingSession,
                                in TpString applicationDescription,
                                in TpMerchantAccountID merchantAccount,
                                in TpAddress user,
                                in TpCorrelationID correlationID,
                                out TpChargingSessionID chargingSession
```

```
                                    )
                                    raises (TpCommonExceptions, P_INVALID_USER,
P_INVALID_ACCOUNT);

                  };

            };

      };

};

#endif
```