**3GPP TSG CN Plenary Meeting #19** **NP-030026**
**12- 14 March 2003, Birmingham, UK**

**Source:** **CN5 (OSA)**

**Title:** **Rel-4 CRs 29.198-12 OSA API Part 12: Charging**

**Agenda item:** **7.10**

**Document for:** **APPROVAL**

---

| Doc-1st-Level | Spec | CR | Rev | Phase | Subject | Cat | Version-Current | Doc-2nd-Level | Workitem |
|---|---|---|---|---|---|---|---|---|---|
| NP-030026 | 29.198-12 | 020 | - | Rel-4 | Correction of status of methods to Charging interfaces | F | 4.3.0 | N5-021137 | OSA1 |
| NP-030026 | 29.198-12 | 021 | - | Rel-5 | Addition of status of methods to Charging interfaces | A | 5.1.0 | N5-021138 | OSA2 |

CR-Form-v7

# CHANGE REQUEST

⌘ 29.198-12 CR 020 ⌘rev - ⌘ Current version: 4.3.0 ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

**Proposed change affects:** UICC apps⌘ [ ] ME [ ] Radio Access Network [ ] Core Network [X]

| | | |
|---|---|---|
| **Title:** | ⌘ | Correction of status of methods to Charging interfaces |
| **Source:** | ⌘ | N5 |
| **Work item code:** | ⌘ | OSA1 **Date:** ⌘ 31/10/2002 |
| **Category:** | ⌘ | **F** **Release:** ⌘ *REL-4* |

*Use one of the following categories:*
- ***F*** *(correction)*
- ***A*** *(corresponds to a correction in an earlier release)*
- ***B*** *(addition of feature),*
- ***C*** *(functional modification of feature)*
- ***D*** *(editorial modification)*

Detailed explanations of the above categories can be found in 3GPP TR 21.900.

*Use one of the following releases:*

| | |
|---|---|
| *2* | *(GSM Phase 2)* |
| *R96* | *(Release 1996)* |
| *R97* | *(Release 1997)* |
| *R98* | *(Release 1998)* |
| *R99* | *(Release 1999)* |
| *Rel-4* | *(Release 4)* |
| *Rel-5* | *(Release 5)* |
| *Rel-6* | *(Release 6)* |

| | | |
|---|---|---|
| **Reason for change:** | ⌘ | There is no requirement in the standard about the necessity to implement all or only some of the methods defined for an interface. |
| **Summary of change:** | ⌘ | Add a statement that clarifies which methods are mandatory and which are optional. |
| **Consequences if not approved:** | ⌘ | Application developers will not know which methods will actually be available. |

| | | |
|---|---|---|
| **Clauses affected:** | ⌘ | 4, 8 |

**Other specs affected:** ⌘

| Y | N | |
|---|---|---|
| | X | Other core specifications |
| | X | Test specifications |
| | X | O&M Specifications |

⌘

**Other comments:** ⌘

## How to create CRs using this form:

Comprehensive information and tips about how to create CRs can be found at http://www.3gpp.org/specs/CR.htm. Below is a brief summary:

1) Fill out the above form. The symbols above marked ⌘ contain pop-up help information about the field that they are closest to.

2) Obtain the latest version for the release of the specification to which the change is proposed. Use the MS Word "revision marks" feature (also known as "track changes") when making the changes. All 3GPP specifications can be downloaded from the 3GPP server under ftp://ftp.3gpp.org/specs/ For the latest version, look for the directory name with the latest date e.g. 2001-03 contains the specifications resulting from the March 2001 TSG meetings.

3) With "track changes" disabled, paste the entire CR form (use CTRL-A to select it) into the specification just in front of the clause containing the first piece of changed text. Delete those parts of the specification which are not relevant to the change request.

# 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 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.

## 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.

# 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.

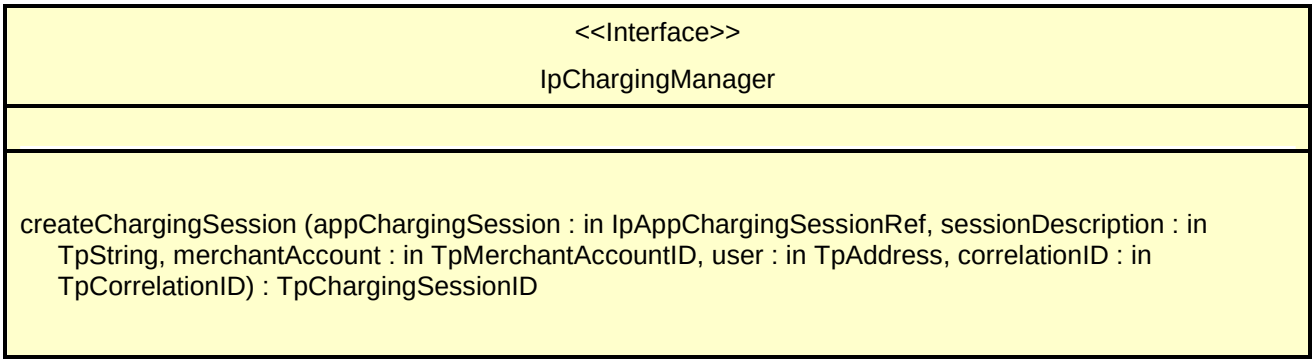
The units (of different types) that are used in a TpVolumeSet are NOT consolidated by the charging SCF. The application must use the same units when making the reservation and when debiting the amount. For example, when after a reservation of 10 minutes a debit request for 5 seconds is done, an error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

## 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 and the createChargingSession() method shall be implemented by a Charging SCF.




*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.

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**

## 8.2 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.

| <<Interface>><br>IpAppChargingManager |
|---|
| |
| sessionAborted (sessionID : in TpSessionID) : void |

*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.

## 8.3 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 creditAmountReq() methods shall also be implemented. If the reserveUnitReq() method is implemented, at least one of the debitUnitReq() or 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

*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**

*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**

*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**

*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**

*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*

## 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*

## 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*

## 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**

*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*

## 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**

*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**

*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**

*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**

*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*

## 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**

*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>><br>IpAppChargingSession |
|---|
| |
| creditAmountErr (sessionID : in TpSessionID, requestNumber : in TpInt32, error : in TpChargingError, requestNumberNextRequest : in TpInt32) : void<br>creditAmountRes (sessionID : in TpSessionID, requestNumber : in TpInt32, creditedAmount : in TpChargingPrice, reservedAmountLeft : in TpChargingPrice, requestNumberNextRequest : in TpInt32) : void<br>creditUnitErr (sessionID : in TpSessionID, requestNumber : in TpInt32, error : in TpChargingError, requestNumberNextRequest : in TpInt32) : void<br>creditUnitRes (sessionID : in TpSessionID, requestNumber : in TpInt32, creditedVolumes : in TpVolumeSet, reservedUnitsLeft : in TpVolumeSet, requestNumberNextRequest : in TpInt32) : void<br>debitAmountErr (sessionID : in TpSessionID, requestNumber : in TpInt32, error : in TpChargingError, requestNumberNextRequest : in TpInt32) : void<br>debitAmountRes (sessionID : in TpSessionID, requestNumber : in TpInt32, debitedAmount : in TpChargingPrice, reservedAmountLeft : in TpChargingPrice, requestNumberNextRequest : in TpInt32) : void<br>debitUnitErr (sessionID : in TpSessionID, requestNumber : in TpInt32, error : in TpChargingError, requestNumberNextRequest : in TpInt32) : void<br>debitUnitRes (sessionID : in TpSessionID, requestNumber : in TpInt32, debitedVolumes : in TpVolumeSet, reservedUnitsLeft : in TpVolumeSet, requestNumberNextRequest : in TpInt32) : void<br>directCreditAmountErr (sessionID : in TpSessionID, requestNumber : in TpInt32, error : in TpChargingError, requestNumberNextRequest : in TpInt32) : void<br>directCreditAmountRes (sessionID : in TpSessionID, requestNumber : in TpInt32, creditedAmount : in TpChargingPrice, requestNumberNextRequest : in TpInt32) : void<br>directCreditUnitErr (sessionID : in TpSessionID, requestNumber : in TpInt32, error : in TpChargingError, requestNumberNextRequest : in TpInt32) : void<br>directCreditUnitRes (sessionID : in TpSessionID, requestNumber : in TpInt32, creditedVolumes : in TpVolumeSet, requestNumberNextRequest : in TpInt32) : void<br>directDebitAmountErr (sessionID : in TpSessionID, requestNumber : in TpInt32, error : in TpChargingError, requestNumberNextRequest : in TpInt32) : void<br>directDebitAmountRes (sessionID : in TpSessionID, requestNumber : in TpInt32, debitedAmount : in TpChargingPrice, requestNumberNextRequest : in TpInt32) : void<br>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

*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.

*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.

*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.

*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.

*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.

*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.

*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.

*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.

*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.

*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.

*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.

*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.

*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

**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.

**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.

*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

**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.

**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.

*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*

## 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*

## 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*

## 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*

## 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

**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.

**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 remains valid.

**requestNumberNextRequest : in TpInt32**

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

*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

**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.

**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 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 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.

CR-Form-v7

# CHANGE REQUEST

⌘ **29.198-12** **CR** **021** ⌘ **rev** **-** ⌘ Current version: **5.1.0** ⌘

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

***Proposed change affects:*** UICC apps⌘ ☐ ME ☐ Radio Access Network ☐ Core Network **X**

| | | |
|---|---|---|
| ***Title:*** | ⌘ | Addition of status of methods to Charging interfaces |
| ***Source:*** | ⌘ | N5 |
| ***Work item code:*** | ⌘ | OSA2 |
| ***Date:*** | ⌘ | 31/10/2002 |
| ***Category:*** | ⌘ | **A** |
| ***Release:*** | ⌘ | *REL-5* |

*Use one of the following categories:*
***F*** *(correction)*
***A*** *(corresponds to a correction in an earlier release)*
***B*** *(addition of feature),*
***C*** *(functional modification of feature)*
***D*** *(editorial modification)*
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

*Use one of the following releases:*

| | |
|---|---|
| *2* | *(GSM Phase 2)* |
| *R96* | *(Release 1996)* |
| *R97* | *(Release 1997)* |
| *R98* | *(Release 1998)* |
| *R99* | *(Release 1999)* |
| *Rel-4* | *(Release 4)* |
| *Rel-5* | *(Release 5)* |
| *Rel-6* | *(Release 6)* |

| | | |
|---|---|---|
| ***Reason for change:*** | ⌘ | There is no requirement in the standard about the necessity to implement all or only some of the methods defined for an interface. |
| ***Summary of change:*** | ⌘ | Add a statement that clarifies which methods are mandatory and which are optional. |
| ***Consequences if not approved:*** | ⌘ | Application developers will not know which methods will actually be available. |

| | | | | | |
|---|---|---|---|---|---|
| ***Clauses affected:*** | ⌘ | 4, 8 | | | |
| ***Other specs affected:*** | ⌘ | **Y** | **N** | | ⌘ |
| | | | **X** | Other core specifications | |
| | | | **X** | Test specifications | |
| | | | **X** | O&M Specifications | |
| ***Other comments:*** | ⌘ | | | | |

**How to create CRs using this form:**
Comprehensive information and tips about how to create CRs can be found at http://www.3gpp.org/specs/CR.htm. Below is a brief summary:

1) Fill out the above form. The symbols above marked ⌘ contain pop-up help information about the field that they are closest to.

2) Obtain the latest version for the release of the specification to which the change is proposed. Use the MS Word "revision marks" feature (also known as "track changes") when making the changes. All 3GPP specifications can be downloaded from the 3GPP server under ftp://ftp.3gpp.org/specs/ For the latest version, look for the directory name with the latest date e.g. 2001-03 contains the specifications resulting from the March 2001 TSG meetings.

3) With "track changes" disabled, paste the entire CR form (use CTRL-A to select it) into the specification just in front of the clause containing the first piece of changed text. Delete those parts of the specification which are not relevant to the change request.

# 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 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.

## 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.

# 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 amont 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.

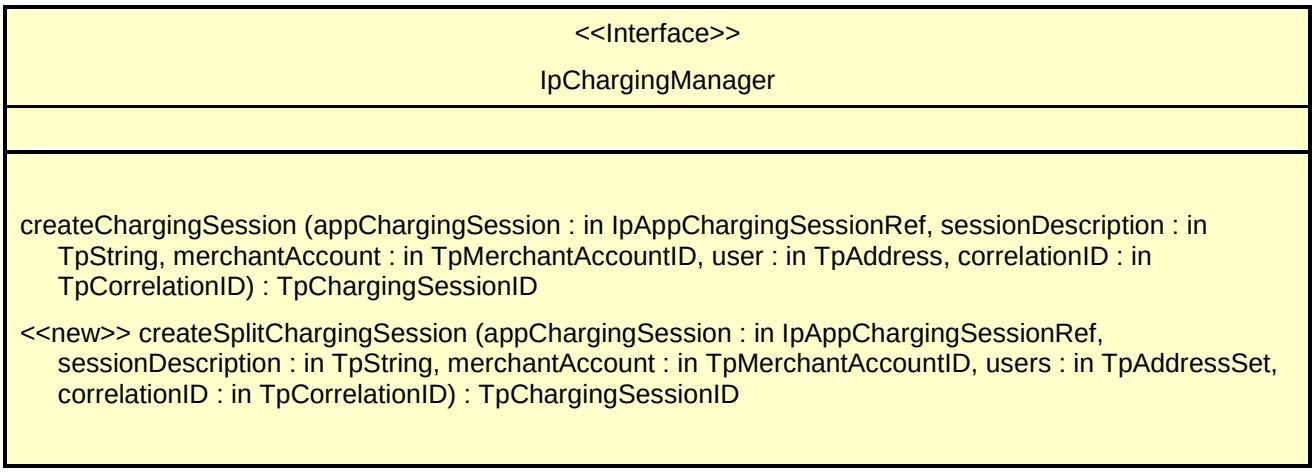
It is subject to service level agreements that are negotiated between the OSA client provider and the network operator how the charge is split between the users.

## 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.




## 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 <<new>> 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**

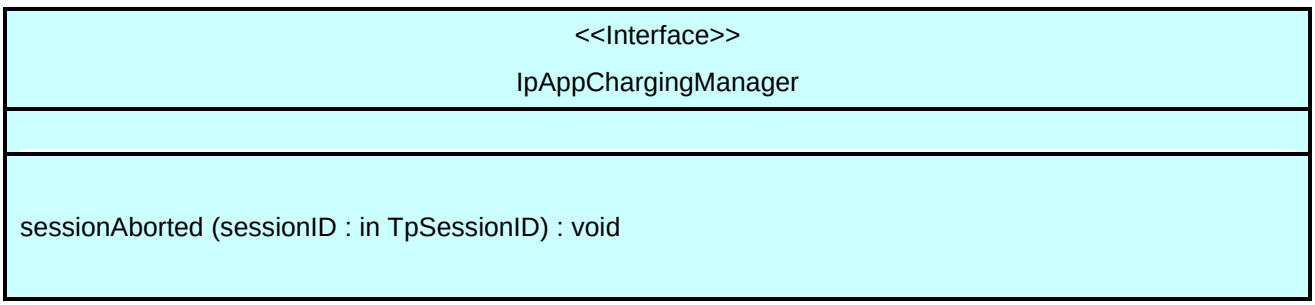
*Raises*

**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.

## 8.3 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 milli-seconds 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 remains 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 remains 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.