Final draft ETSI ES 201 915-11 V1.1.1 (2001-12)

Open Service Access;
Application Programming Interface;
Part 11: Account Management SCF

Reference
DES/SPAN-120070-11

Keywords
API, OSA, IDL, UML

ETSI

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

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

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

Important notice

Individual copies of the present document can be downloaded from:
http://www.etsi.org

The present document may be made available in more than one electronic version or in print. In any case of existing or
perceived difference in contents between such versions, the reference version is the Portable Document Format (PDF).
In case of dispute, the reference shall be the printing on ETSI printers of the PDF version kept on a specific network drive
within ETSI Secretariat.

Users of the present document should be aware that the document may be subject to revision or change of status.
Information on the current status of this and other ETSI documents is available at
http://portal.etsi.org/tb/status/status.asp

If you find errors in the present document, send your comment to:
editor@etsi.fr

Copyright Notification

ETSI

Contents

Intellectual Property Rights

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

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

Foreword

This ETSI Standard (ES) has been produced by ETSI Technical Committee Services and Protocols for Advanced Networks (SPAN), and is now submitted for the ETSI standards Membership Approval Procedure.

The present document is part 11 of a multi-part deliverable covering Open Service Access (OSA); Application Programming Interface (API), as identified below. The API specification (ES 201 915) is structured in the following parts:

Part 1:   "Overview";

Part 2:   "Common Data Definitions";

Part 3:   "Framework";

Part 4:   "Call Control SCF";

Part 5:   "User Interaction SCF";

Part 6:   "Mobility SCF";

Part 7:   "Terminal Capabilities SCF";

Part 8:   "Data Session Control SCF";

Part 9:   "Generic Messaging SCF";

Part 10:   "Connectivity Manager SCF";

Part 11:   "Account Management SCF";

Part 12:   "Charging SCF".

The present document has been defined jointly between ETSI, The Parlay Group [24] and the 3GPP , in co-operation with a number of JAIN™ Community [25] member companies.

The present document forms part of the Parlay 3.0 set of specifications.

1 Scope

The present document is part 11 of the Stage 3 specification for an Application Programming Interface (API) for Open Service Access (OSA).

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

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

- Sequence Diagrams

- Class Diagrams

- Interface specification plus detailed method descriptions

- State Transition diagrams

- Data Definitions

- IDL Description of the interfaces

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

2 References

The references listed in clause 2 of ES 201 915-1 contain provisions which, through reference in this text, constitute provisions of the present document.

ETSI ES 201 915-1: "Open Service Access; Application Programming Interface; Part 1: Overview".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in ES 201 915-1 apply.

3.2 Abbreviations

For the purposes of the present document, the abbreviations given in ES 201 915-1 apply.

4 Account Management SCF

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

The order is as follows:

- The Sequence diagrams give the reader a practical idea of how each of the SCF is implemented.

- The Class relationships clause show how each of the interfaces applicable to the SCF, relate to one another.

- The Interface specification clause describes in detail each of the interfaces shown within the Class diagram part.

- The State Transition Diagrams (STD) show the transition between states in the SCF. The states and transitions are well-defined; either methods specified in the Interface specification or events occurring in the underlying networks cause state transitions.

- The Data Definitions clause shows a detailed expansion of each of the data types associated with the methods within the classes. Note that some data types are used in other methods and classes and are therefore defined within the Common Data types part of this specification.

5 Sequence Diagrams

5.1 Standard Transaction History Retrieval
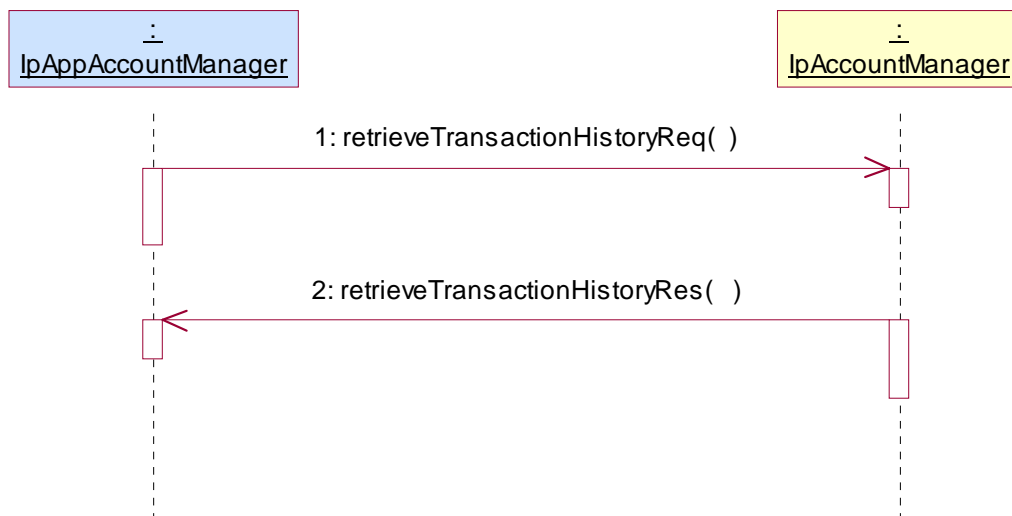
1: This message is used by the application to retrieve a transaction history for a certain subscriber's account.

2: This method passes the result of the transaction history retrieval request for a specific user to its callback object.

5.2   Standard Query Handling
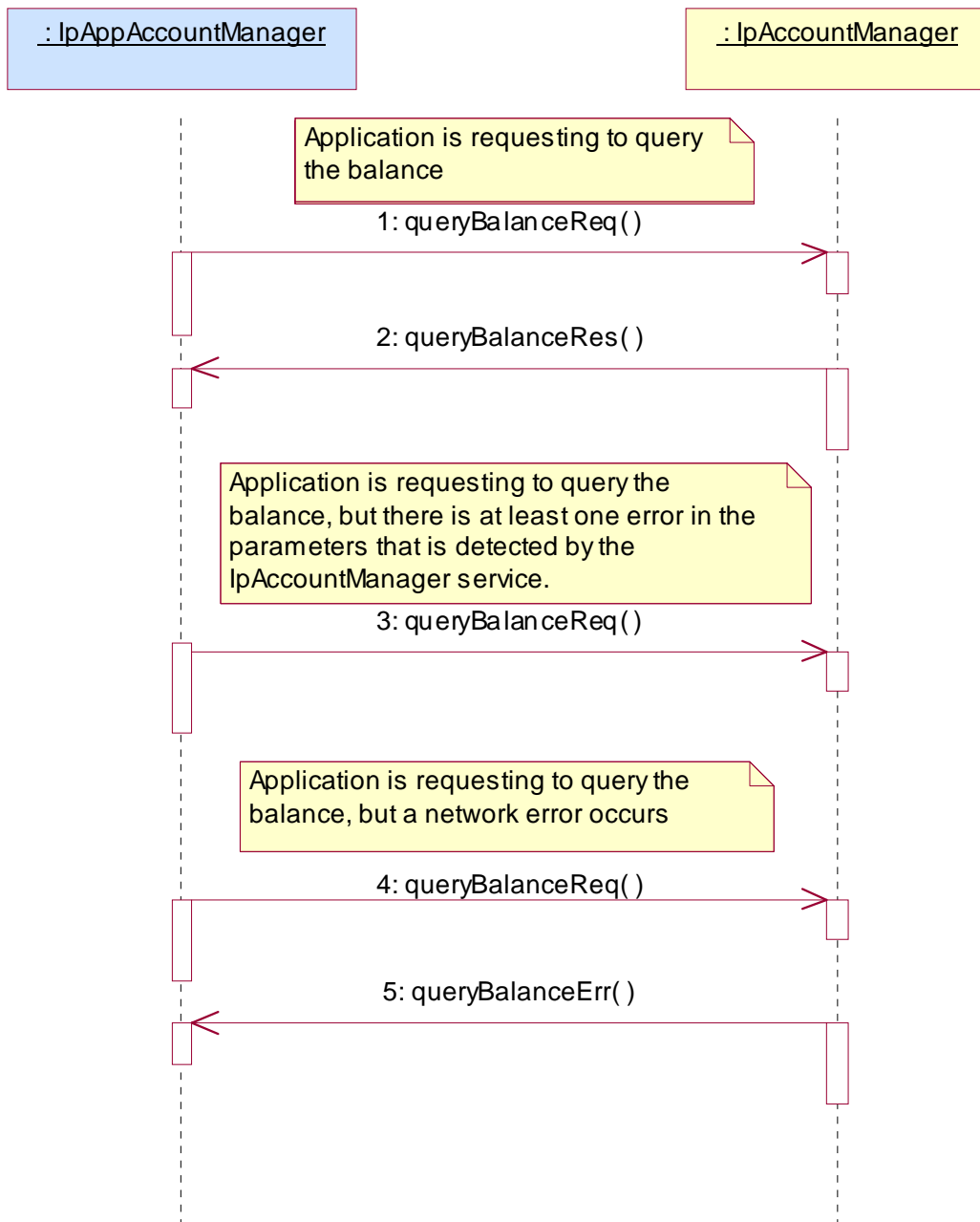
1: This message is used to query the balance of the account of one or several users.

2: This message passes the result of the balance query for one or several users to its callback object.

3: This scenario shows the case where at least one error in the parameters of the message is detected by the IpAccountManager object. An exception will be thrown.

4: This scenario shows the case where a network error occurs.

5: This message passes the error of the balance query. No exception is thrown.

5.3   Standard Notification handling
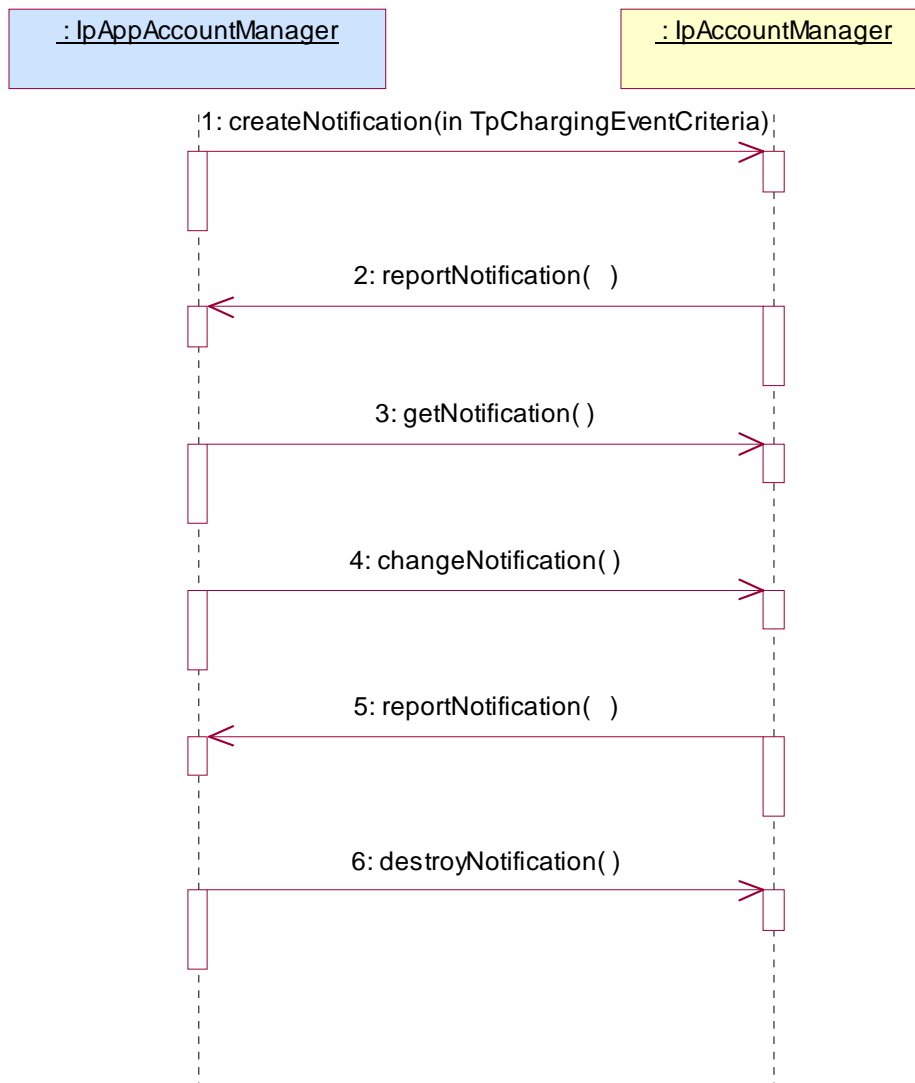
1: This message is used by the application to request notifications from the IpAccountManager service on certain criteria for one or several users.

2: This message is used by the IpAccountManager service to report a charging event that meets the criteria set in the createNotification message.

3: The application can request the current criteria set in the IpAccountManager service by invoking the getNotification method.

4: This message is used by the application to change the criteria initially created by createNotification, and previously obtained by getNotification.

5: This message is used by the IpAccountManager service to report a charging event that meets the new criteria.

6: This method is used by the application to disable the charging notifications.

6   Class Diagrams

Figure 1: Application Interfaces

```
┌─────────────────────────────────────┐
│            IpService                 │
├─────────────────────────────────────┤
├─────────────────────────────────────┤
│  setCallback()                       │
│  setCallbackWithSessionID()          │
└─────────────────────────────────────┘
                   △
                   │
                   │
┌─────────────────────────────────────┐
│            <<Interface>>             │
│          IpAccountManager            │
│            (from am)                  │
├─────────────────────────────────────┤
├─────────────────────────────────────┤
│  ◆createNotification()               │
│  ◆destroyNotification()              │
│  ◆queryBalanceReq()                  │
│  ◆changeNotification()               │
│  ◆getNotification()                  │
│  ◆retrieveTransactionHistoryReq()    │
└─────────────────────────────────────┘
```
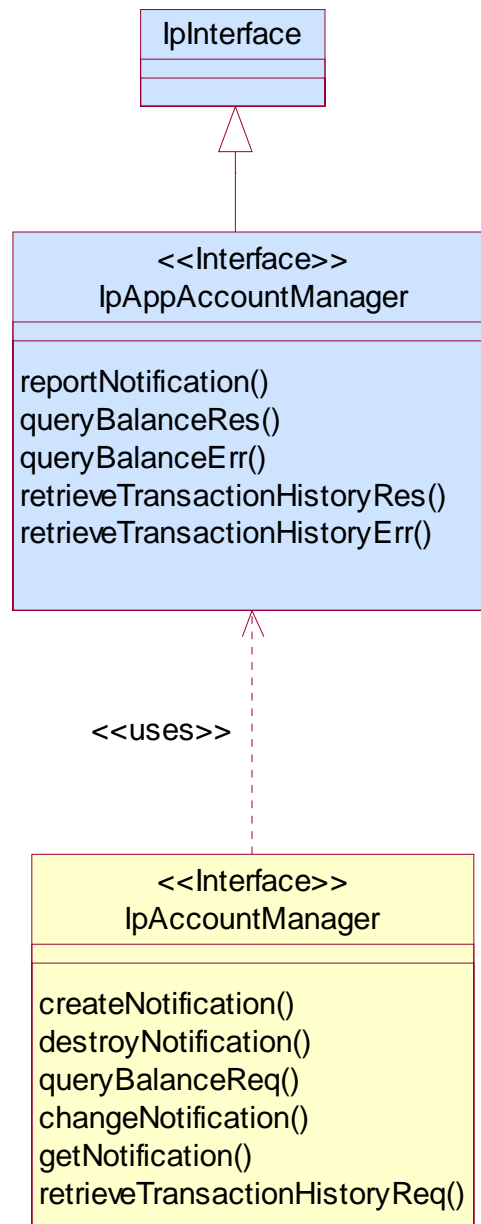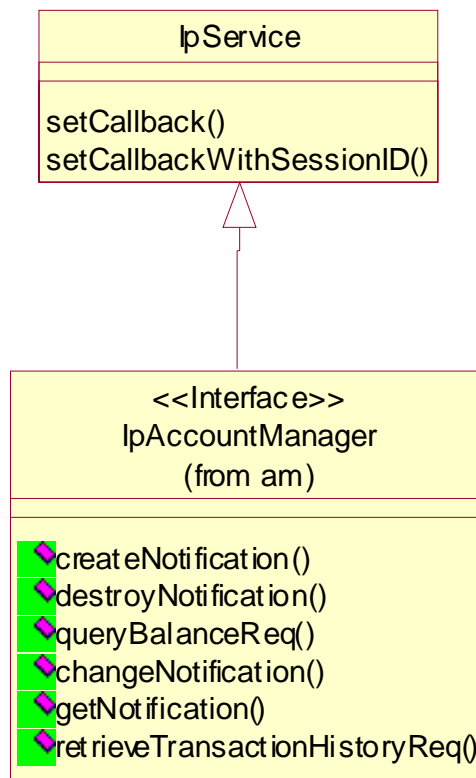
Figure 2: Service Interfaces

7 The Service Interface Specifications

7.1 Interface Specification Format

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

7.1.1 Interface Class

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

7.1.2 Method descriptions

Each method (API method "call") is described. All methods in the API return a value of type TpResult, indicating, amongst other things, if the method invocation was successfully executed or not.

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

7.1.3   Parameter descriptions

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

7.1.4   State Model

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

7.2   Base Interface

7.2.1   Interface Class IpInterface

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

<<Interface>> IpInterface

7.3   Service Interfaces

7.3.1   Overview

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

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

7.4   Generic Service Interface

7.4.1   Interface Class IpService

Inherits from: IpInterface

All service interfaces inherit from the following interface.

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

Method
setCallback()

This method specifies the reference address of the callback interface that a service uses to invoke methods on the application. It is not allowed to invoke this method on an interface that uses SessionIDs.

Parameters

appInterface : in IpInterfaceRef

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

Raises

TpCommonExceptions

Method
setCallbackWithSessionID()

This method specifies the reference address of the application's callback interface that a service uses for interactions associated with a specific session ID: e.g. a specific call, or call leg. It is not allowed to invoke this method on an interface that does not uses SessionIDs.

Parameters

appInterface : in IpInterfaceRef

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

sessionID : in TpSessionID

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

Raises

TpCommonExceptions, P_INVALID_SESSION_ID

8   Account Management Interface Classes

8.1   Interface Class IpAccountManager

Inherits from: IpService.

The account manager interface provides methods for monitoring accounts. Applications can use this interface to enable or disable charging-related event notifications and to query account balances.

<<Interface>>
IpAccountManager
createNotification (ChargingEventCriteria : in TpChargingEventCriteria) : TpAssignmentID destroyNotification (assignmentId : in TpAssignmentID) : void queryBalanceReq (users : in TpAddressSet) : TpSessionID changeNotification (assignmentID : in TpAssignmentID, eventCriteria : in TpChargingEventCriteria) : void getNotification () : TpChargingEventCriteriaResult retrieveTransactionHistoryReq (user : in TpAddress, transactionInterval : in TpTimeInterval) : TpSessionID

Method
createNotification()

This method is used by the application to enable charging event notifications to be sent to the application.

Returns assignmentId : Specifies the ID assigned by the account management object for this newly enabled event notification.

Parameters

ChargingEventCriteria : in TpChargingEventCriteria

Specifies the event specific criteria used by the application to define the charging event required. Individual addresses or address ranges may be specified for subscriber accounts. Example of events are "charging" and "recharging".

Returns

TpAssignmentID

Raises

**TpCommonExceptions, P_INVALID_ADDRESS, P_INVALID_CRITERIA,
P_INVALID_EVENT_TYPE, P_UNKNOWN_SUBSCRIBER**

Method
destroyNotification()

This method is used by the application to disable charging notifications.

Parameters

assignmentId : in TpAssignmentID

Specifies the assignment ID that was given by the account management object when the application enabled the charging notification.

Raises

TpCommonExceptions, P_INVALID_ASSIGNMENT_ID

Method
queryBalanceReq()

This method is used by the application to query the balance of an account for one or several users.

Returns queryId : Specifies the ID of the balance query request.

Parameters

users : in TpAddressSet

Specifies the user(s) for which the balance is queried.

Returns

TpSessionID

Raises

TpCommonExceptions, P_UNKNOWN_SUBSCRIBER, P_UNAUTHORIZED_APPLICATION

Method
changeNotification()

This method is used by the application to change the event criteria introduced with createNotification. Any stored criteria associated with the specified assignementID will be replaced with the specified criteria.

Parameters

assignmentID : in TpAssignmentID

Specifies the ID assigned by the manager interface for the event notification.

eventCriteria : in TpChargingEventCriteria

Specifies the new set of event criteria used by the application to define the event required. Only events that meet these criteria are reported.

Raises

TpCommonExceptions, P_INVALID_ASSIGNMENT_ID, P_INVALID_CRITERIA, P_INVALID_EVENT_TYPE, P_UNKNOWN_SUBSCRIBER, P_INVALID_ADDRESS

Method
getNotification()

This method is used by the application to query the event criteria set with createNotification or changeNotification.

Returns eventCriteria : Specifies the event criteria used by the application to define the event required. Only events that meet these criteria are reported.

Parameters
No Parameters were identified for this method.

Returns

TpChargingEventCriteriaResult

Raises

TpCommonExceptions

Method
retrieveTransactionHistoryReq()

This asynchronous method is used by the application to retrieve a transaction history of a subscriber's account. The history is a set of Detailed Records.

Returns retrievalID : Specifies the retrieval ID of the transaction history retrieval request.

Parameters
user : in TpAddress

Specifies the subscriber for whose account the transaction history is to be retrieved.

transactionInterval : in TpTimeInterval

Specifies the time interval for which the application history is to be retrieved.

Returns

TpSessionID

Raises

TpCommonExceptions, P_UNKNOWN_SUBSCRIBER, P_UNAUTHORIZED_APPLICATION, P_INVALID_TIME_AND_DATE_FORMAT

8.2 Interface Class IpAppAccountManager

Inherits from: IpInterface.

The account manager application interface is implemented by the client application developer and is used to handle charging event notifications and query balance responses.

<<Interface>>
IpAppAccountManager
reportNotification (chargingEventInfo : in TpChargingEventInfo, assignmentId : in TpAssignmentID) : void queryBalanceRes (queryId : in TpSessionID, balances : in TpBalanceSet) : void queryBalanceErr (queryId : in TpSessionID, cause : in TpBalanceQueryError) : void retrieveTransactionHistoryRes (retrievalID : in TpSessionID, transactionHistory : in TpTransactionHistorySet) : void retrieveTransactionHistoryErr (retrievalID : in TpSessionID, transactionHistoryError : in TpTransactionHistoryStatus) : void

Method
reportNotification()

This method is used to notify the application of a charging event.

Parameters

chargingEventInfo : in TpChargingEventInfo

Specifies data associated with this charging event. These data include the charging event being notified, the current value of the balance after the notified event occurred, and the time at which the charging event occurred.

assignmentId : in TpAssignmentID

Specifies the assignment ID that was returned by the createNotification() method. The application can use the assignment ID to associate events with event-specific criteria and to act accordingly.

Method
queryBalanceRes()

This method indicates that the request to query the balance was successful and it reports the requested balance of an account to the application.

Parameters

queryId : in TpSessionID

Specifies the ID of the balance query request.

balances : in TpBalanceSet

Specifies the balance for one or more user accounts.

Method
queryBalanceErr()

This method indicates that the request to query the balance failed and it reports the cause of failure to the application.

Parameters

queryId : in TpSessionID

Specifies the ID of the balance query request.

cause : in TpBalanceQueryError

Specifies the error that led to the failure.

Method
retrieveTransactionHistoryRes()

This method indicates that the request to retrieve the transaction history was successful and it returns the requested transaction history.

Parameters

retrievalID : in TpSessionID

Specifies the retrievalID of the transaction history retrieval request.

transactionHistory : in TpTransactionHistorySet

Specifies the requested transaction history.

Method
retrieveTransactionHistoryErr()

This method indicates that the request to retrieve the transaction history failed and it reports the cause of failure to the application.

Parameters

retrievalID : in TpSessionID

Specifies the retrievalID of the transaction history retrieval request.

transactionHistoryError : in TpTransactionHistoryStatus

Specifies the error that occurred while retrieving the transaction history.

9   State Transition Diagrams

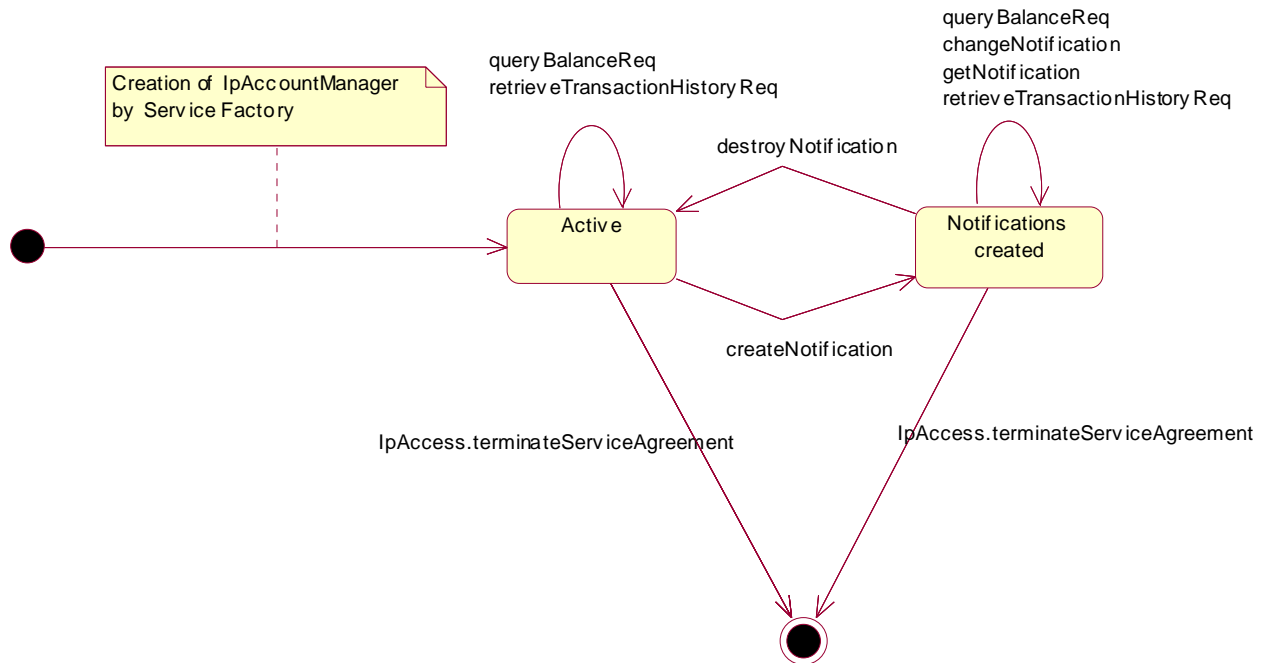## 9.1   State Transition Diagrams for IpAccountManager

Figure 3: Application view on the IpAccountManager

9.1.1   Active State

In this state a relation between the Application and the Account Management has been established. The state allows the application to indicate that it is interested in charging related events, by calling createNotification. In case such an event occurs, Account Manager will inform the application by invoking the operation reportNotification() on the IpAppAccountManager interface. The application can also indicate it is no longer interested in certain charging related events by calling destroyNotification().

9.1.2   Notifications created State

When the Account Manager is in the Notifications created state, events requested with createNotification() will be forwarded to the application. In this state the application can request to change the notifications or query the Account Manager for the notifications currently set.

10   Account Management Service Properties

The following table lists properties relevant for the Account Management API.

Property	Type	Description/Interpretation
P_ EVENT_TYPES	INTEGER_SET	Indicates the event types supported by the SCS. Static events are the events by which applications are initiated.
P_ADDRESSPLAN	INTEGER_SET	Indicates the supported address plan (defined in TpAddressPlan.) E.g. {P_ADDRESS_PLAN_E164, P_ADDRESS_PLAN_IP})

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

Property	Type	Description/Interpretation
P_TRIGGERING_ADDRESSES	ADDRESS_RANGE_SET	Indicates for which numbers the notification may be set. For terminating notifications it applies to the terminating number, for originating notifications it applies only to the originating number.
P_CURRENCY_ALLOWED	STRING_SET	Indicates the currencies that can be returned in the queryBalanceRes. The valid values for the string set are according to ISO-4217:1995. E.g. {"EUR", "NLG"}.
P_HISTORY_ALLOWED	STRING_SET	Indicates the length of the transaction history interval that is allowed to be retrieved by the application. The valid values for the string are according to TpDateAndTime. The string-set will be of format {"lower_start_time", "upper_stop_time"}, e.g. {"1998-12-04 10:30", "1999-12-04 10:30"}
P_BULK_QUERY_ALLOWED	BOOLEAN_SET	Indicates whether the application is allowed to issue a queryBalanceReq for more than one user. Value = TRUE : the users parameter of type TpAddressSet may contain more than one user. Value = FALSE : the users parameter of type TpAddressSet may contain only one user.

11   Data Definitions

11.1   Account Management Data Definitions

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

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

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

- Description, that describes the data type.

- Tabular specification, that specifies the data types and values of the data type.

- Example, if relevant, shown to illustrate the data type.

11.1.1   TpBalanceQueryError

Defines an error that is reported by the Charging service capability feature as a result of a balance query request.

Name	Value	Description
P_BALANCE_QUERY_OK	0	No error occurred while processing the request
P_BALANCE_QUERY_ERROR_UNDEFINED	1	General error, unspecified
P_BALANCE_QUERY_UNKNOWN_SUBSCRIBER	2	Subscriber for which balance is queried is unknown
P_BALANCE_QUERY_UNAUTHORIZED_APPLICATION	3	Application is not authorized to query balance
P_BALANCE_QUERY_SYSTEM_FAILURE	4	System failure. The request could not be handled

11.1.2   TpChargingEventName

Defines the charging event for which notifications can be requested by the application.

Name	Value	Description
P_AM_CHARGING	0	End user's account has been charged by an application
P_AM_RECHARGING	1	End user has recharged the account
P_AM_ACCOUNT_LOW	2	Account balance is below the balance threshold
P_AM_ACCOUNT_ZERO	3	Account balance is at zero
P_AM_ACCOUNT_DISABLED	4	Account has been disabled

11.1.3   TpBalanceInfo

Defines the structure of data elements that specifies detailed balance info.

Structured Member Name	Structured Member Type	Description
Currency	TpString	Currency unit according to ISO-4217:1995
ValuePartA	TpInt32	This data type is identical to a TpInt32 and specifies the most significant part of the composed value. A currency amount is composed as follows: ((ValuePartA*2^{32}+ ValuePartB) $\times$ 0,0001)
ValuePartB	TpInt32	This data type is identical to a TpInt32 and specifies the least significant part of the composed value.
Exponent	TpInt32	Specifies the position of the decimal point in the currency amount made up of the unitPart and the fractionPart, as described above. E.g. an exponent of 4 means a pure integer value, whereas an exponent of 2 means an accuracy of 0.01.
AdditionalInfo	TpString	Descriptive string, containing additional information, which is sent to the application without prior evaluation.

As an example, the currency amount composed of a Currency of EUR, a ValuePartA of 0, a ValuePartB of 10 000, and an exponent of 2 yields a currency amount of € 100,00.

Valid Currencies are:

ADP, AED, AFA, ALL, AMD, ANG, AON, AOR, ARS, ATS, AUD, AWG, AZM, BAM,
BBD, BDT, BEF, BGL, BGN, BHD, BIF, BMD, BND, BOB, BOV, BRL, BSD, BTN,
BWP, BYB, BZD, CAD, CDF, CHF, CLF, CLP, CNY, COP, CRC, CUP, CVE, CYP,
CZK, DEM, DJF, DKK, DOP, DZD, ECS, ECV, EEK, EGP, ERN, ESP, ETB, EUR,
FIM, FJD, FKP, FRF, GBP, GEL, GHC, GIP, GMD, GNF, GRD, GTQ, GWP, GYD,
HKD, HNL, HRK, HTG, HUF, IDR, IEP, ILS, INR, IQD, IRR, ISK, ITL, JMD,
JOD, JPY, KES, KGS, KHR, KMF, KPW, KRW, KWD, KYD, KZT, LAK, LBP, LKR,
LRD, LSL, LTL, LUF, LVL, LYD, MAD, MDL, MGF, MKD, MMK, MNT, MOP, MRO,
MTL, MUR, MVR, MWK, MXN, MXV, MYR, MZM, NAD, NGN, NIO, NLG, NOK, NPR,
NZD, OMR, PAB, PEN, PGK, PHP, PKR, PLN, PTE, PYG, QAR, ROL, RUB, RUR,
RWF, SAR, SBD, SCR, SDD, SEK, SGD, SHP, SIT, SKK, SLL, SOS, SRG, STD,
SVC, SYP, SZL, THB, TJR, TMM, TND, TOP, TPE, TRL, TTD, TWD, TZS, UAH,
UGX, USD, USN, USS, UYU, UZS, VEB, VND, VUV, WST, XAF, XAG, XAU, XBA,
XBB, XBC, XBD, XCD, XDR, XFO, XFU, XOF, XPD, XPF, XPT, XTS, XXX, YER,
YUM, ZAL, ZAR, ZMK, ZRN, ZWD.

XXX is used for transactions where no currency is involved.

11.1.4 TpChargingEventInfo

Defines the structure of data elements that specifies charging event information.

Structured Member Name	Structured Member Type	Description
ChargingEventName	TpChargingEventName	The charging event for which notifications can be requested by the application
CurrentBalanceInfo	TpBalanceInfo	The current balance of the user's account
ChargingEventTime	TpTime	The time at which the charging event occurred.

11.1.5 TpChargingEventCriteria

Defines the structure of data elements that specifies charging event criteria.

Structured Member Name	Structured Member Type	Description
Users	TpAddressSet	Specifies the user(s) for which the charging events are requested to be reported.
ChargingEvents	TpChargingEventNameSet	Specifies the specific charging event criteria used by the application to define the event required.

11.1.6 TpChargingEventNameSet

Defines a collection of TpChargingEventName elements.

11.1.7 TpChargingEventCriteriaResult

Defines the Sequence of Data Elements that specify the criteria relating to event requests.

Sequence Element Name	Sequence Element Type
ChargingEventCriteria	TpChargingEventCriteria
AssignmentID	TpAssignmentID

11.1.8 TpBalance

Defines the structure of data elements that specifies a balance.

Structured Member Name	Structured Member Type	Description
UserID	TpAddress	Specifies the user to whom the account belongs to.
StatusCode	TpBalanceQueryError	Specifies the status code for the balance query request.
BalanceInfo	TpBalanceInfo	Specifies the balance information for the user.

11.1.9 TpBalanceSet

Defines a collection of TpBalance elements.

11.1.10 TpTransactionHistory

This data type is a sequence of data elements that describes the transaction history.

Sequence Element Name	Sequence Element Type	Description
TransactionID	TpAssignmentID	Specifies the ID of the specific transaction
TimeStamp	TpDateAndTime	Specifies the date and time when the specific transaction was processed.
AdditionalInfo	TpString	Specifies a free format string providing additional information on the specific transaction. This could be the applicationDescription provided with the actual transaction.

11.1.11 TpTransactionHistorySet

Defines a collection of TpTransactionHistory elements.

11.1.12 TpTransactionHistoryStatus

Defines a status code that is reported by the Account Manager service capability feature as a result of a transaction history retrieval request.

Name	Value	Description
P_AM_TRANSACTION_ERROR_UNSPECIFIED	0	General error, unspecified
P_AM_TRANSACTION_INVALID_INTERVAL	1	An invalid interval for the transaction history was specified.
P_AM_TRANSACTION_UNKNOWN_ACCOUNT	2	No account for the specified user is known.
P_AM_TRANSACTION_UNAUTHORIZED_APPLICATION	3	Application is not authorized to query balance.
P_AM_TRANSACTION_PROCESSING_ERROR	4	A processing error occurred while compiling the transaction history.
P_AM_TRANSACTION_SYSTEM_FAILURE	5	System failure. The request could not be handled

12   Exception Classes

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

Name	Description
P_UNAUTHORIZED_APPLICATION	Application is not authorized to perform charging operations

Each exception class contains the following structure:

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

Annex A (normative):
OMG IDL Description of Account Management SCF

The OMG IDL representation of this interface specification is contained in a text file (am.idl contained in archive es_20191511v010101m0.ZIP) which accompanies the present document.

Annex B (informative):
Contents of 3GPP OSA R4 Account Management

All of the present document is relevant for TS 129 198-11 V4 (Release 4).

History

Document history		
V1.1.1	December 2001	Membership Approval Procedure MV 20020215: 2001-12-18 to 2002-02-15