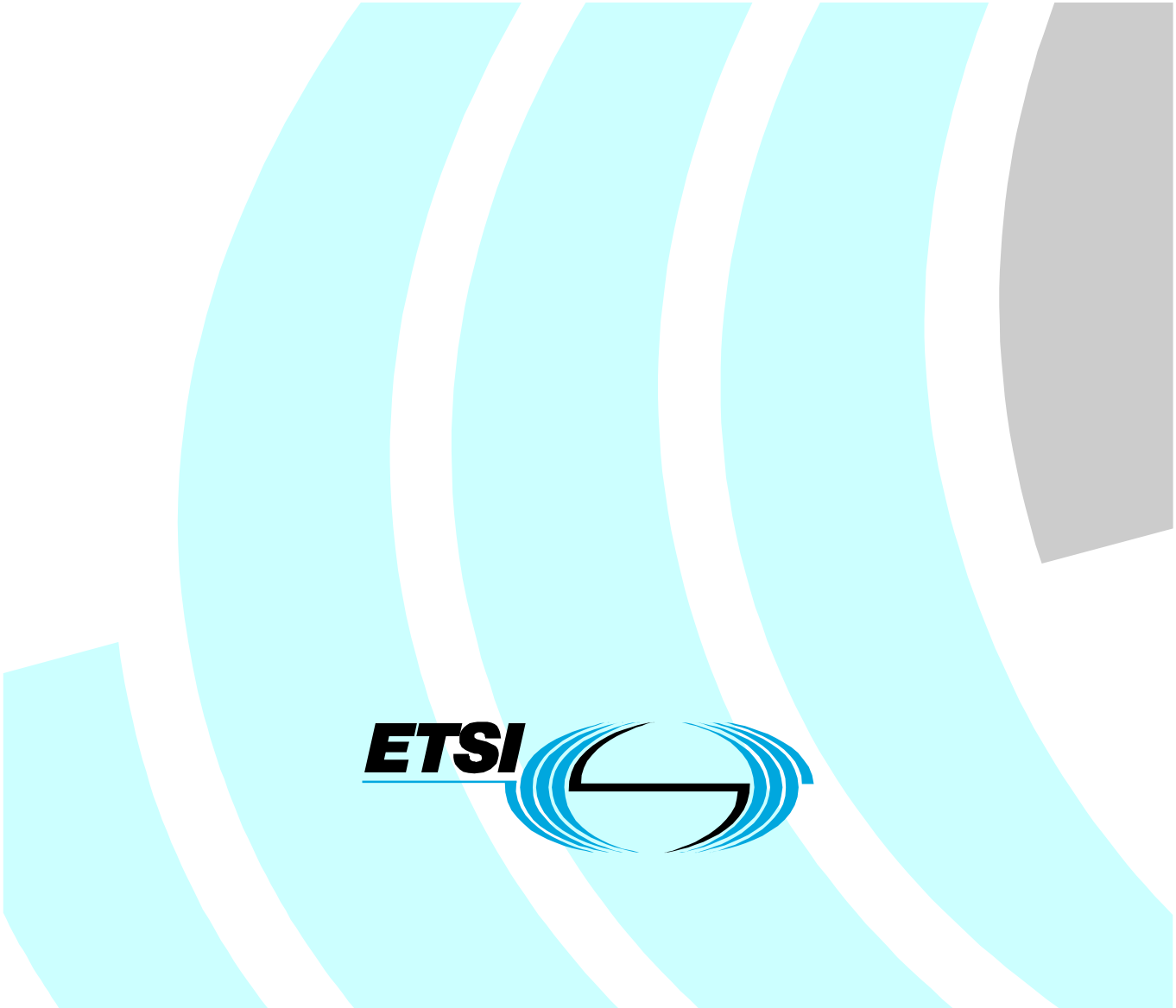
ETSI ES 202 388-11 V1.1.1 (2005-03)

ETSI Standard

Open Service Access (OSA); Application Programming Interface (API); Test Suite Structure and Test Purposes (TSS&TP); Part 11: Account Management SCF (Parlay 4)



Reference
DES/TISPAN-06004-11-OSA

Keywords
API, OSA, TSS&TP

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, please send your comment to one of the following services:
http://portal.etsi.org/chaircor/ETSI_support.asp

Copyright Notification

Contents

Intellectual Property Rights 4
Foreword 4
1 Scope 5
2 References 5
3 Definitions and abbreviations 5
3.1 Definitions 5
3.2 Abbreviations 6
4 Test Suite Structure (TSS) 6
5 Test Purposes (TP) 6
5.1 Introduction 6
5.1.1 TP naming convention 6
5.1.2 Source of TP definition 6
5.1.3 Test strategy 7
5.2 TPs for the Account Management SCF 7
5.2.1 Account Management 7
5.3 TPs for the application using the Account Management SCF 20
5.3.1 Account Management 20
History 29

Intellectual Property Rights

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

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

Foreword

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

The present document is part 11 of a multi-part deliverable. Full details of the entire series can be found in part 1 [6].

To evaluate conformance of a particular implementation, it is necessary to have a set of test purposes to evaluate the dynamic behaviour of the Implementation Under Test (IUT). The specification containing those test purposes is called a Test Suite Structure and Test Purposes (TSS&TP) specification.

1 Scope

The present document provides the Test Suite Structure and Test Purposes (TSS&TP) specification for the Account Management SCF of the Application Programming Interface (API) for Open Service Access (OSA) defined in ES 202 915-11 [1] in compliance with the relevant requirements, and in accordance with the relevant guidance given in ISO/IEC 9646-2 [4] and ETS 300 406 [5].

2 References

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

- References are either specific (identified by date of publication and/or edition number or version number) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies.

Referenced documents which are not found to be publicly available in the expected location might be found at http://docbox.etsi.org/Reference.

[1] ETSI ES 202 915-11: "Open Service Access (OSA); Application Programming Interface (API); Part 11: Account Management SCF (Parlay 4)".

[2] ETSI ES 202 363: "Open Service Access (OSA); Application Programming Interface (API); Implementation Conformance Statement (ICS) proforma specification; (Parlay 4)".

[3] ISO/IEC 9646-1: "Information technology - Open Systems Interconnection - Conformance testing methodology and framework - Part 1: General concepts".

[4] ISO/IEC 9646-2: "Information technology - Open Systems Interconnection - Conformance testing methodology and framework - Part 2: Abstract Test Suite specification".

[5] ETSI ETS 300 406: "Methods for Testing and Specification (MTS); Protocol and profile conformance testing specifications; Standardization methodology".

[6] ETSI ES 202 388-1: "Open Service Access (OSA); Application Programming Interface (API); Test Suite Structure and Test Purposes (TSS&TP); Part 1: Overview (Parlay 4)".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in ES 202 915-11 [1], ISO/IEC 9646-1 [3], ISO/IEC 9646-2 [4] and the following apply:

abstract test case: Refer to ISO/IEC 9646-1 [3].

Abstract Test Method (ATM): Refer to ISO/IEC 9646-1 [3].

Abstract Test Suite (ATS): Refer to ISO/IEC 9646-1 [3].

Implementation Under Test (IUT): Refer to ISO/IEC 9646-1 [3].

Lower Tester (LT): Refer to ISO/IEC 9646-1 [3].

Implementation Conformance Statement (ICS): Refer to ISO/IEC 9646-1 [3].

ICS proforma: Refer to ISO/IEC 9646-1 [3].

Implementation eXtra Information for Testing (IXIT): Refer to ISO/IEC 9646-1 [3].

IXIT proforma: Refer to ISO/IEC 9646-1 [3].

Test Purpose (TP): Refer to ISO/IEC 9646-1 [3].

3.2 Abbreviations

For the purposes of the present document, the following abbreviations apply:

AM	Account Management
API	Application Programming Interface
ATM	Abstract Test Method
ATS	Abstract Test Suite
ICS	Implementation Conformance Statement
IUT	Implementation Under Test
IXIT	Implementation eXtra Information for Testing
LT	Lower Tester
OSA	Open Service Access
SCF	Service Capability Feature
TP	Test Purpose
TSS	Test Suite Structure

4 Test Suite Structure (TSS)

- Account Management

5 Test Purposes (TP)

5.1 Introduction

For each test requirement a TP is defined.

5.1.1 TP naming convention

TPs are numbered, starting at 01, within each group. Groups are organized according to the TSS. Additional references are added to identify the actual test suite (see table 1).

Table 1: TP identifier naming convention scheme

Identifier: <suite_id>_<group>_<nnn>		
<suite_id>	= SCG name:	"AM" for **A**ccount **M**anagement part of Account Management SCF
<group>	= group number:	two character field representing the group reference according to TSS
<nn>	= sequential number:	(01 to 99)

5.1.2 Source of TP definition

The TPs are based on ES 202 915-11 [1].

5.1.3 Test strategy

As the base standard ES 202 915-11 [1] contains no explicit requirements for testing, the TPs were generated as a result of an analysis of the base standard and the ICS specification ES 202 363 [2].

The TPs are only based on conformance requirements related to the externally observable behaviour of the IUT and are limited to conceivable situations to which a real implementation is likely to be faced (see ETS 300 406 [5]).

5.2 TPs for the Account Management SCF

All ICS items referred to in this clause are as specified in ES 202 363 [2] unless indicated otherwise by another numbered reference.

All parameters specified in method calls are valid unless specified.

The procedures to trigger the SCF to call methods in the application are dependant on the underlying network architecture and are out of the scope of the present document. Those method calls are preceded by the words "Triggered action".

5.2.1 Account Management

Test AM_01

Summary: **IpAccountManager** all methods, successful.

Reference: ES 202 915-11 [1], clause 8.1.

Precondition: createNotification and destroyNotification supported.

Preamble: Registration of the IUT (Account Manager SCF) and the tester (application) to the framework. The tester must have obtained a reference to an instance of the IpAccountManager interface through selecting that service and signing the required service agreement.

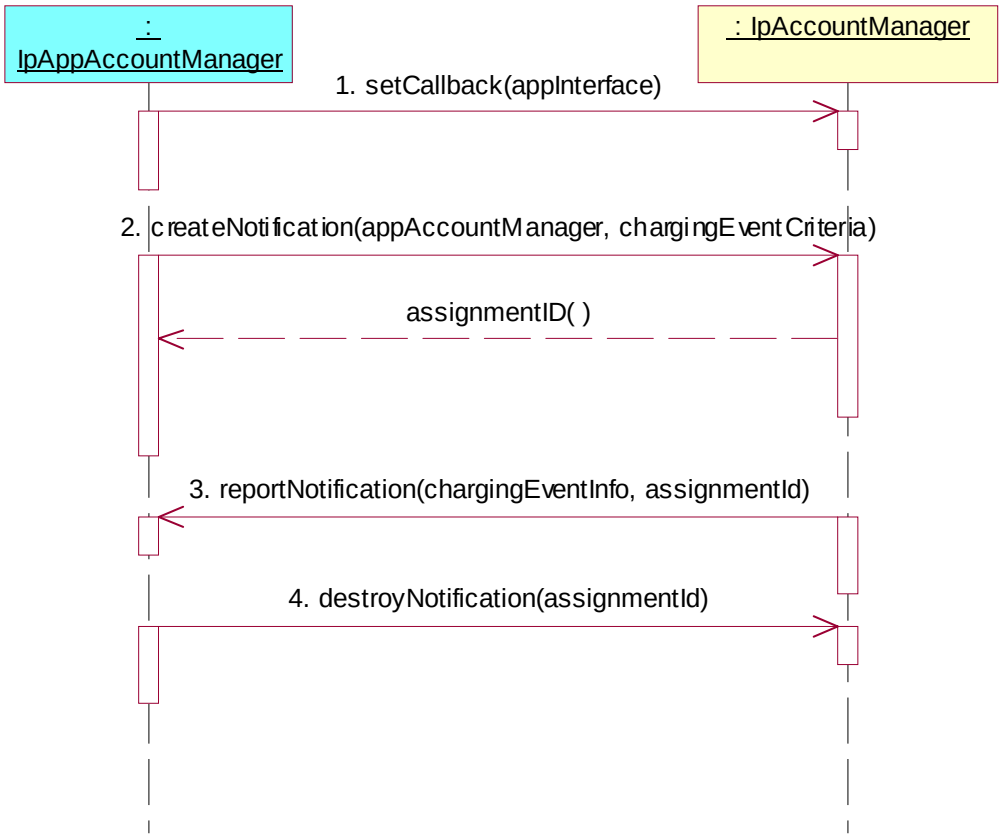
Test Sequence:

1. Method call **setCallback**()
   Parameters: appInterface with value of tester's (Application's) IpAppAccountManager interface
   Check: no exception is returned

2. Method call **createNotification()**
   Parameters: appAccountManager, chargingEventCriteria
   Check: valid value of TpAssignmentID is returned

3. Triggered action: cause IUT to call **reportNotification()** method on the tester's (Application) **IpAppAccountManager** interface.
   Parameters: chargingEventInfo, assignmentId

4. Method call **destroyNotification**()
   Parameters: assignmentID given in 2.
   Check: no exception is returned



Test AM_02

Summary: **IpAccountManager** all methods, successful.

Reference: ES 202 915-11 [1], clause 8.1.

Precondition: createNotification, destroyNotification, getNotification and changeNotification supported.

Preamble: Registration of the IUT (Account Manager SCF) and the tester (application) to the framework. The tester must have obtained a reference to an instance of the IpAccountManager interface through selecting that service and signing the required service agreement.

Test Sequence:

1. Method call **setCallback**()
   Parameters: appInterface with value of tester's (Application's) IpAppAccountManager interface
   Check: no exception is returned

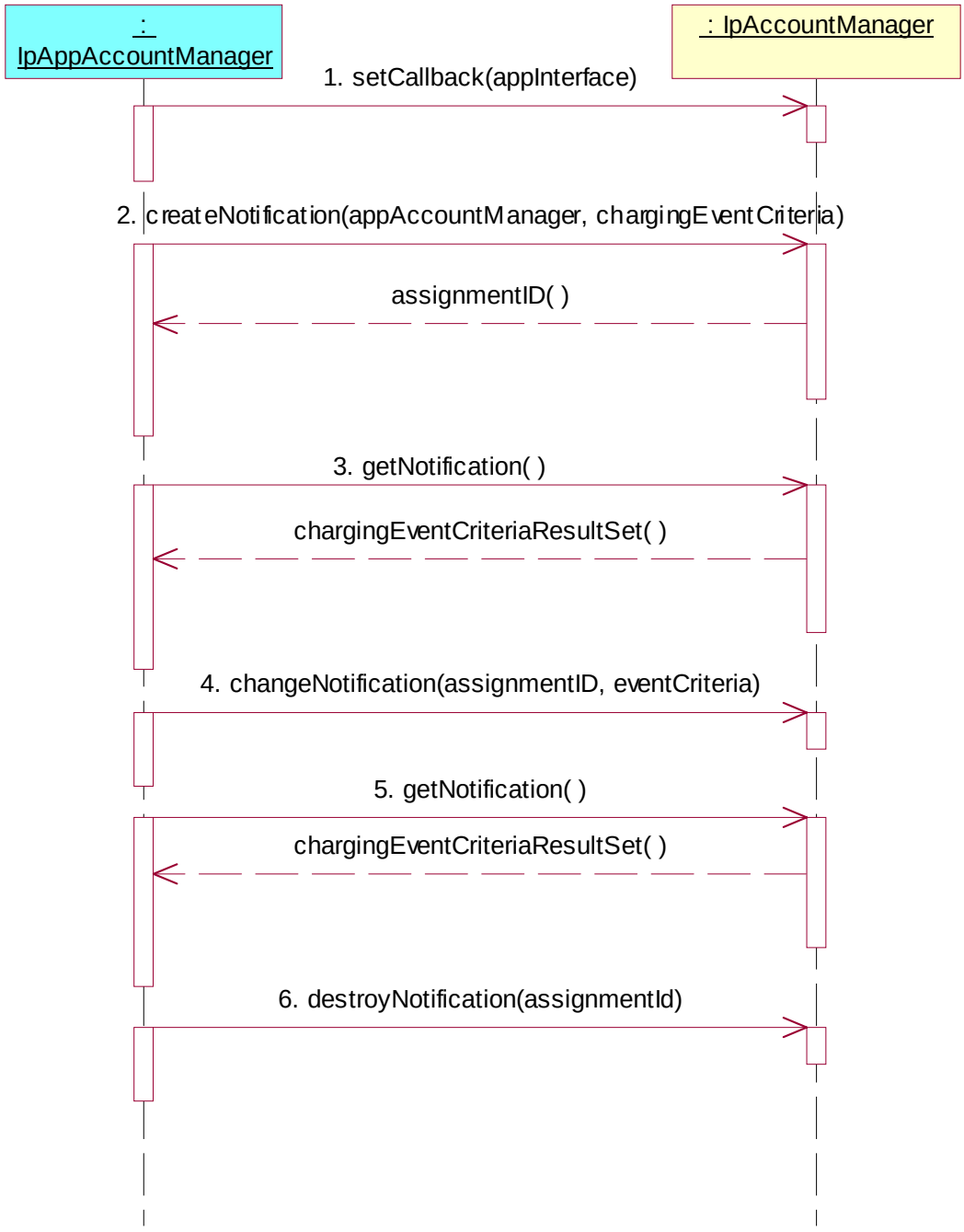
2. Method call **createNotification()**
   Parameters: appAccountManager, chargingEventCriteria
   Check: valid value of TpAssignmentID is returned

3. Method call **getNotification()**
   Parameters: None
   Check: valid value of TpChargingEventCriteriaResultSet is returned, which corresponds to notification created in 1.

4. Method call **changeNotification()**
   Parameters: assignmentID give in 1., eventCriteria
   Check: no exception is returned

5. Method call **getNotification()**
   Parameters: None
   Check: valid value of TpChargingEventCriteriaResultSet is returned, which corresponds to the change made in 3.

6. Method call **destroyNotification**()
   Parameters: assignmentID give in 1.
   Check: no exception is returned



Test AM_03

Summary: **IpAccountManager** createNotification, P_INVALID_CRITERIA.

Reference: ES 202 915-11 [1], clause 8.1.

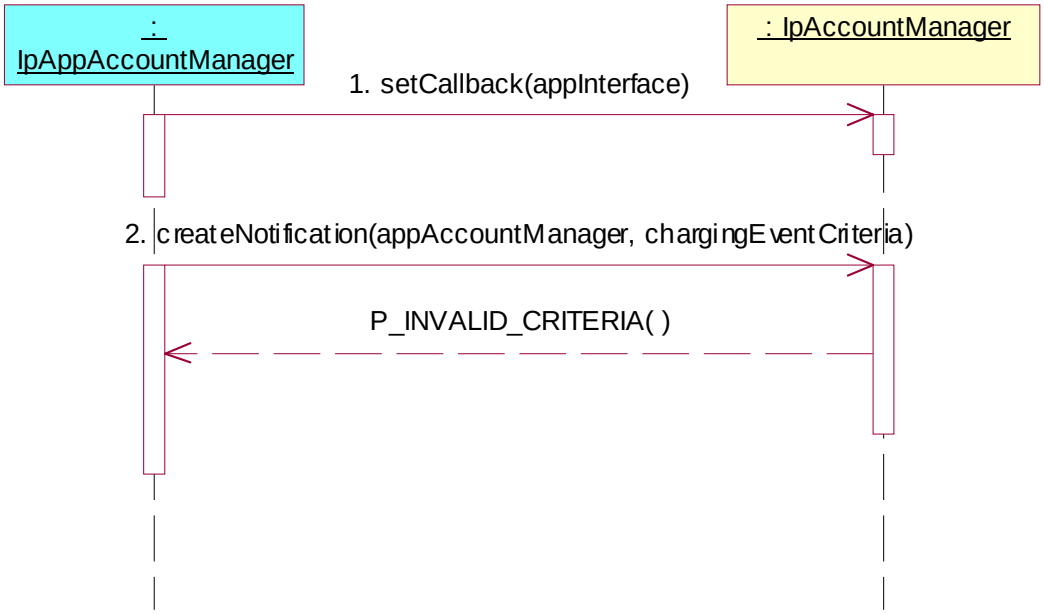
Precondition: createNotification supported.

Preamble: Registration of the IUT (Account Manager SCF) and the tester (application) to the framework. The tester must have obtained a reference to an instance of the IpAccountManager interface through selecting that service and signing the required service agreement.

Test Sequence:

1. Method call **setCallback**()
   Parameters: appInterface with value of tester's (Application's) IpAppAccountManager interface
   Check: no exception is returned

2. Method call **createNotification()**
   Parameters: appAccountManager, invalid chargingEventCriteria
   Check: P_INVALID_CRITERIA is returned



Test AM_04

Summary: **IpAccountManager** changeNotification, P_INVALID_CRITERIA.

Reference: ES 202 915-11 [1], clause 8.1.

Precondition: createNotification and changeNotification supported.

Preamble: Registration of the IUT (Account Manager SCF) and the tester (application) to the framework. The tester must have obtained a reference to an instance of the IpAccountManager interface through selecting that service and signing the required service agreement.

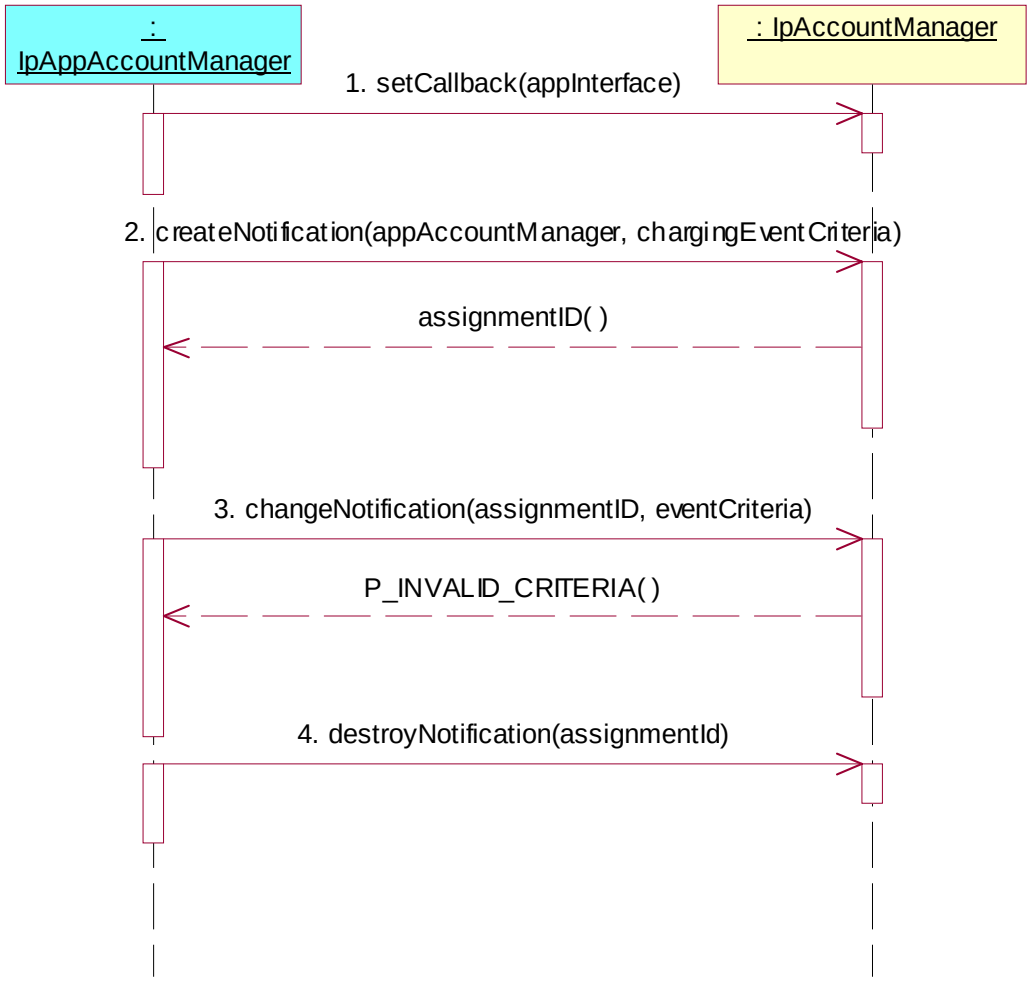
Test Sequence:

1. Method call **setCallback**()
   Parameters: appInterface with value of tester's (Application's) IpAppAccountManager interface
   Check: no exception is returned

2. Method call **createNotification()**
   Parameters: appAccountManager, chargingEventCriteria
   Check: valid value of TpAssignmentID is returned

3. Method call **changeNotification()**
   Parameters: assignmentID give in 1., invalid eventCriteria
   Check: P_INVALID_CRITERIA is returned

4. Method call **destroyNotification**()
   Parameters: assignmentID give in 2.
   Check: no exception is returned



Test AM_05

Summary: **IpAccountManager** changeNotification, P_INVALID_ASSIGNMENT_ID.

Reference: ES 202 915-11 [1], clause 8.1.

Precondition: createNotification and changeNotification supported.

Preamble: Registration of the IUT (Account Manager SCF) and the tester (application) to the framework. The tester must have obtained a reference to an instance of the IpAccountManager interface through selecting that service and signing the required service agreement.

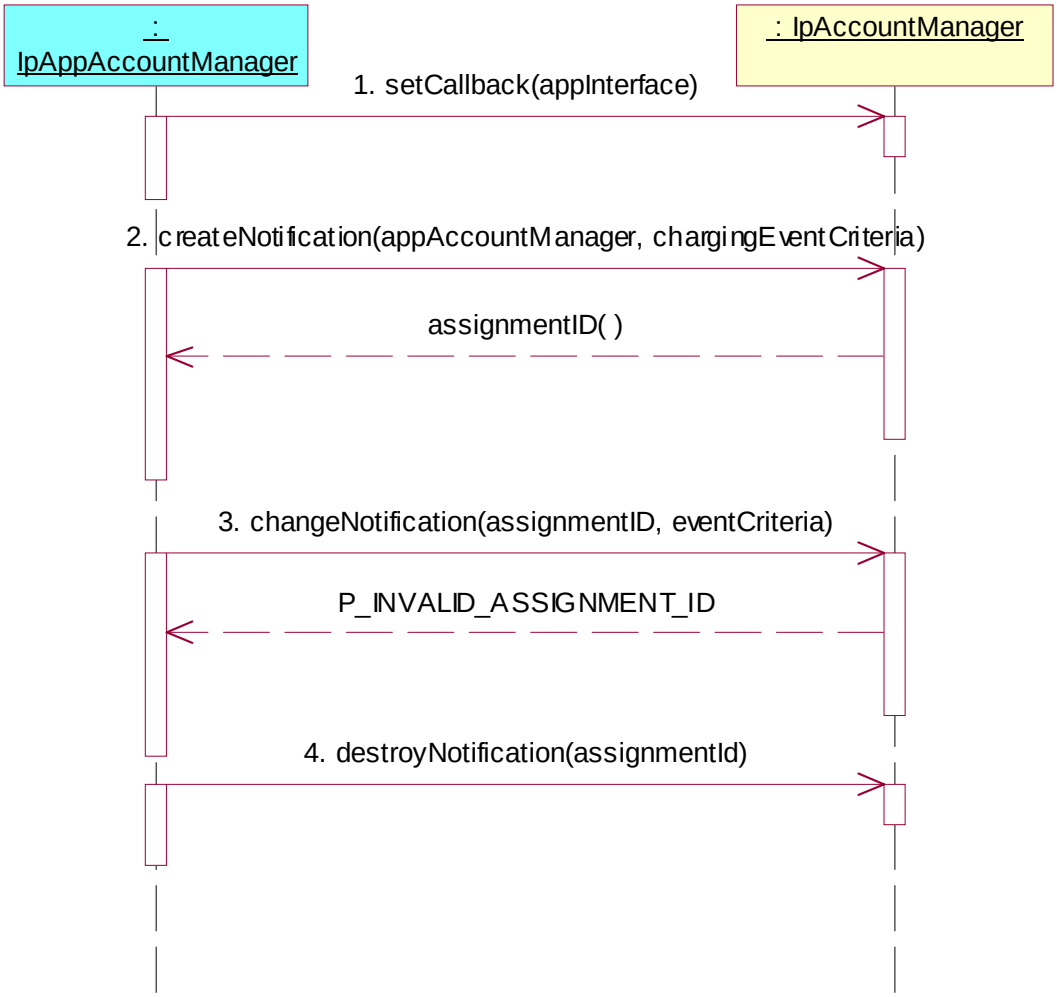
Test Sequence:

1. Method call **setCallback**()
   Parameters: appInterface with value of tester's (Application's) IpAppAccountManager interface
   Check: no exception is returned

2. Method call **createNotification()**
   Parameters: appAccountManager, chargingEventCriteria
   Check: valid value of TpAssignmentID is returned

3. Method call **changeNotification()**
   Parameters: invalid assignmentID, eventCriteria
   Check: P_INVALID_ASSIGNMENT_ID is returned

4. Method call **destroyNotification**()
   Parameters: assignmentID give in 2.
   Check: no exception is returned



Test AM_06

Summary: **IpAccountManager** destroyNotification, P_INVALID_ASSIGNMENT_ID.

Reference: ES 202 915-11 [1], clause 8.1.

Precondition: createNotification and destroyNotification supported.

Preamble: Registration of the IUT (Account Manager SCF) and the tester (application) to the framework. The tester must have obtained a reference to an instance of the IpAccountManager interface through selecting that service and signing the required service agreement.

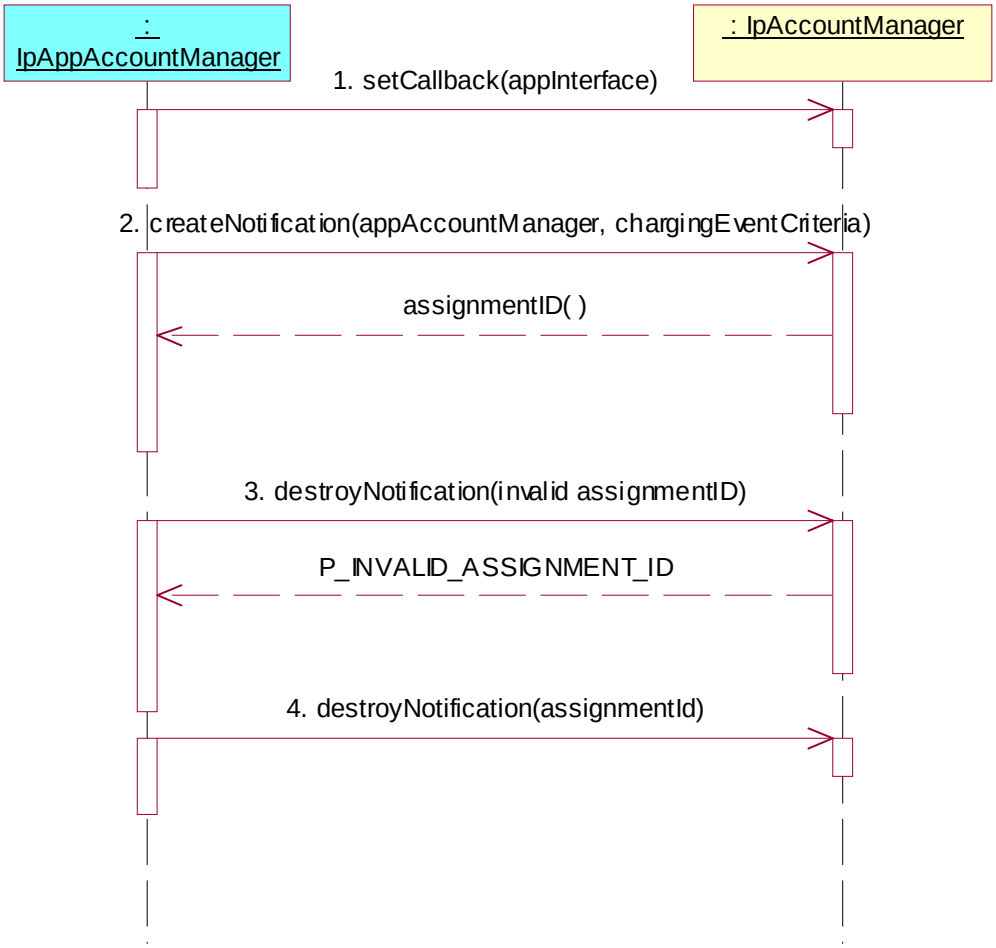
Test Sequence:

1. Method call **setCallback**()
   Parameters: appInterface with value of tester's (Application's) IpAppAccountManager interface
   Check: no exception is returned

2. Method call **createNotification()**
   Parameters: appAccountManager, chargingEventCriteria
   Check: valid value of TpAssignmentID is returned

3. Method call **destroyNotification()**
   Parameters: invalid assignmentID
   Check: P_INVALID_ASSIGNMENT_ID is returned

4. Method call **destroyNotification**()
   Parameters: assignmentID give in 2.
   Check: no exception is returned



Test AM_07

Summary: **IpAccountManager** all methods, successful.

Reference: ES 202 915-11 [1], clause 8.1.

Precondition: retrieveTransactionHistoryReq supported.

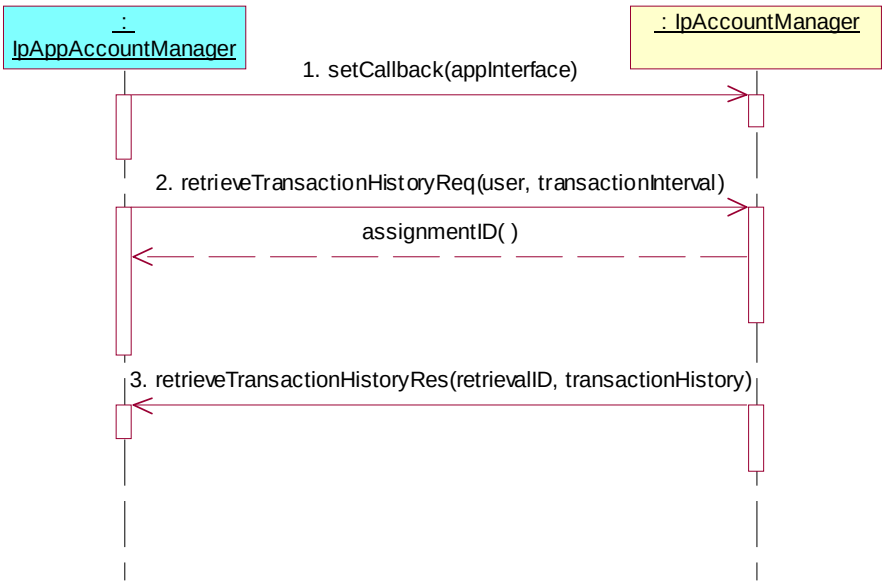
Preamble: Registration of the IUT (Account Manager SCF) and the tester (application) to the framework. The tester must have obtained a reference to an instance of the IpAccountManager interface through selecting that service and signing the required service agreement.

Test Sequence:

1. Method call **setCallback**()
   Parameters: appInterface with value of tester's (Application's) IpAppAccountManager interface
   Check: no exception is returned.

2. Method call **retrieveTransactionHistoryReq()**
   Parameters: user, transactionInterval
   Check: valid value of TpAssignmentID is returned

3. Triggered action: cause IUT to call **retrieveTransactionHistoryRes()** method on the tester's (Application) **IpAppAccountManager** interface.
   Parameters: retrievalID, transactionHistory



Test AM_08

Summary: **IpAccountManager** retrieveTransactionHistoryReq, P_UNKNOWN_SUBSCRIBER.

Reference: ES 202 915-11 [1], clause 8.1.

Precondition: retrieveTransactionHistoryReq supported.

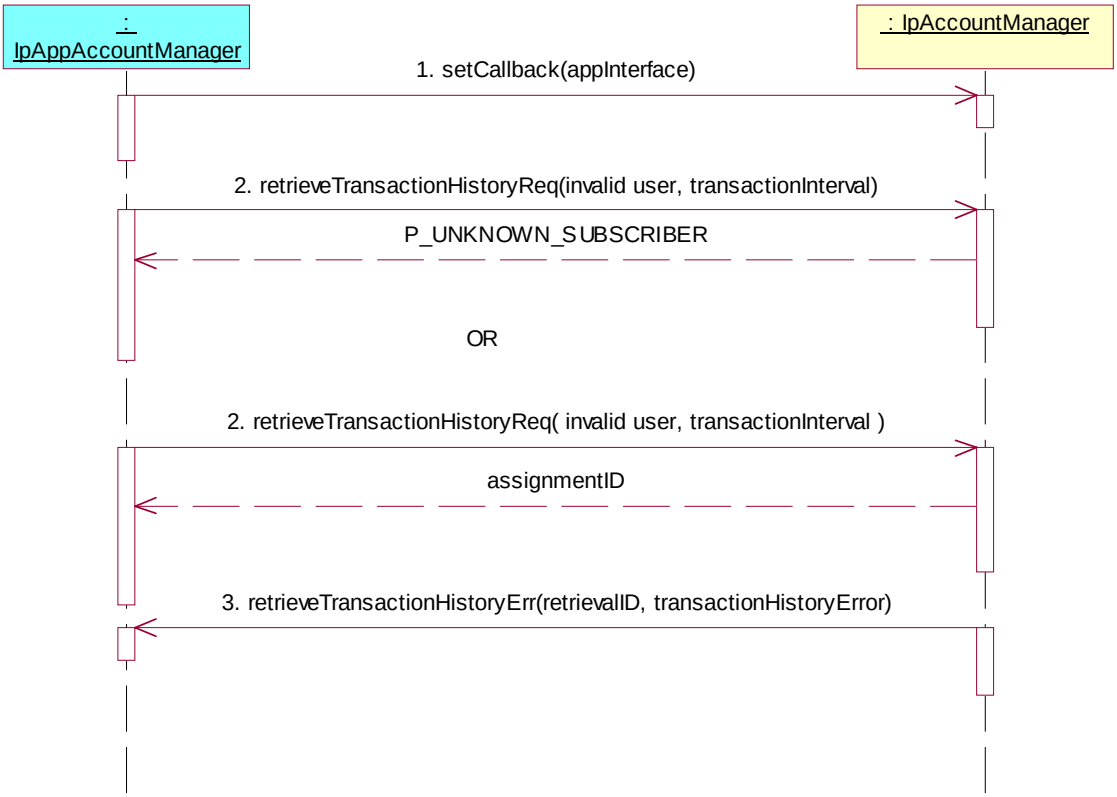
Preamble: Registration of the IUT (Account Manager SCF) and the tester (application) to the framework. The tester must have obtained a reference to an instance of the IpAccountManager interface through selecting that service and signing the required service agreement.

Test Sequence:

1. Method call **setCallback**()
   Parameters: appInterface with value of tester's (Application's) IpAppAccountManager interface
   Check: no exception is returned

2. Method call **retrieveTransactionHistoryReq()**
   Parameters: invalid user, transactionInterval
   Check: P_UNKNOWN_SUBSCRIBER is returned,
   or
   Check: valid value of TpAssignmentID is returned

3. Check: **retrieveTransactionHistoryErr()** method is called on the tester's (Application) **IpAppAccountManager** interface
   Parameters: retrievalID, transactionHistoryError with status code P_AM_TRANSACTION_UNKNOWN_ACCOUNT



Test AM_09

Summary: **IpAccountManager** retrieveTransactionHistoryReq, P_INVALID_TIME_AND_DATE_FORMAT.

Reference: ES 202 915-11 [1], clause 8.1.

Precondition: retrieveTransactionHistoryReq supported.

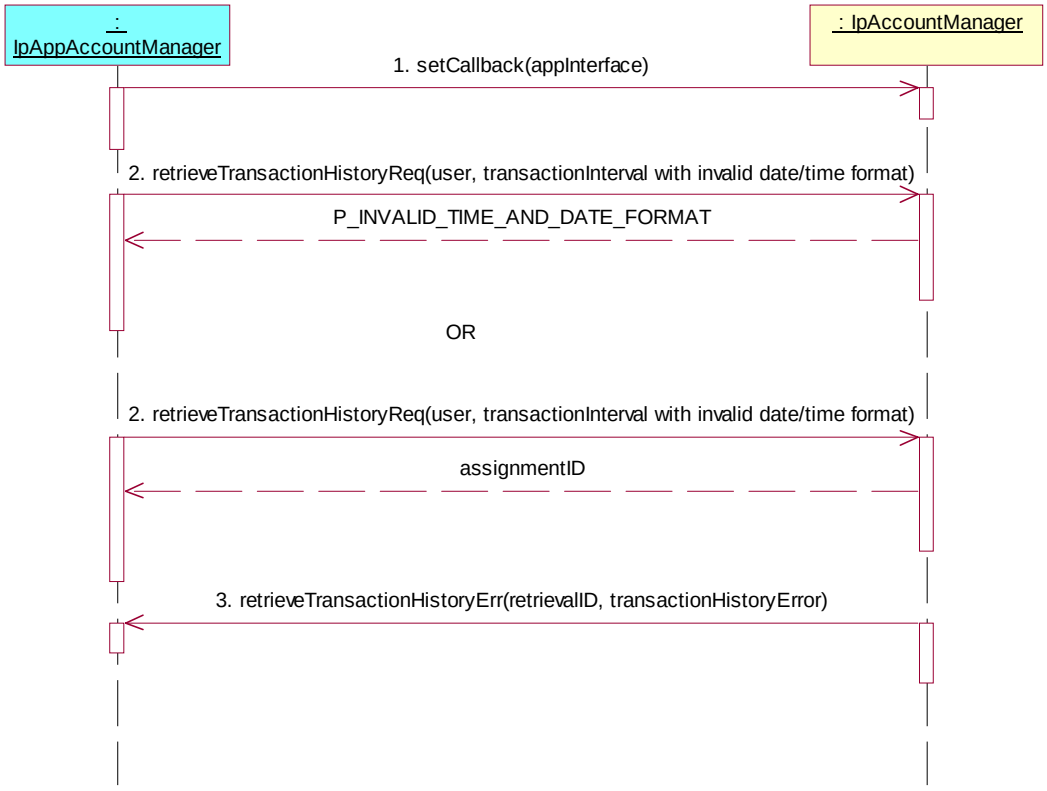
Preamble: Registration of the IUT (Account Manager SCF) and the tester (application) to the framework. The tester must have obtained a reference to an instance of the IpAccountManager interface through selecting that service and signing the required service agreement.

Test Sequence:

1. Method call **setCallback**()
   Parameters: appInterface with value of tester's (Application's) IpAppAccountManager interface
   Check: no exception is returned

2. Method call **retrieveTransactionHistoryReq()**
   Parameters: user, transactionInterval with invalid date/time format
   Check: P_INVALID_TIME_AND_DATE_FORMAT is returned,
   or
   Check: valid value of TpAssignmentID is returned

3. Check: **retrieveTransactionHistoryErr()** method is called on the tester's (Application) **IpAppAccountManager** interface
   Parameters: retrievalID, transactionHistoryError with status code P_AM_TRANSACTION_INVALID_INTERVAL



Test AM_10

Summary: **IpAccountManager** all methods, successful.

Reference: ES 202 915-11 [1], clause 8.1.

Precondition: queryBalanceReq supported.

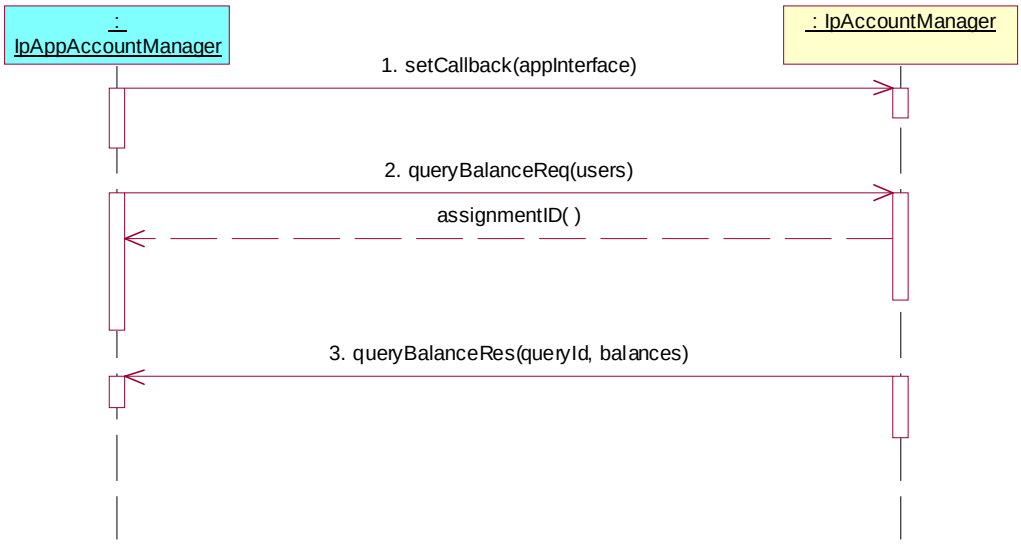
Preamble: Registration of the IUT (Account Manager SCF) and the tester (application) to the framework. The tester must have obtained a reference to an instance of the IpAccountManager interface through selecting that service and signing the required service agreement.

Test Sequence:

1. Method call **setCallback**()
   Parameters: appInterface with value of tester's (Application's) IpAppAccountManager interface
   Check: no exception is returned

2. Method call **queryBalanceReq()**
   Parameters: users
   Check: valid value of TpAssignmentID is returned

3. Triggered action: cause IUT to call **queryBalanceRes()** method on the tester's (Application) **IpAppAccountManager** interface.
   Parameters: queryId, balances



Test AM_11

Summary: **IpAccountManager** queryBalanceReq, P_UNKNOWN_SUBSCRIBER.

Reference: ES 202 915-11 [1], clause 8.1.

Precondition: queryBalanceReq supported.

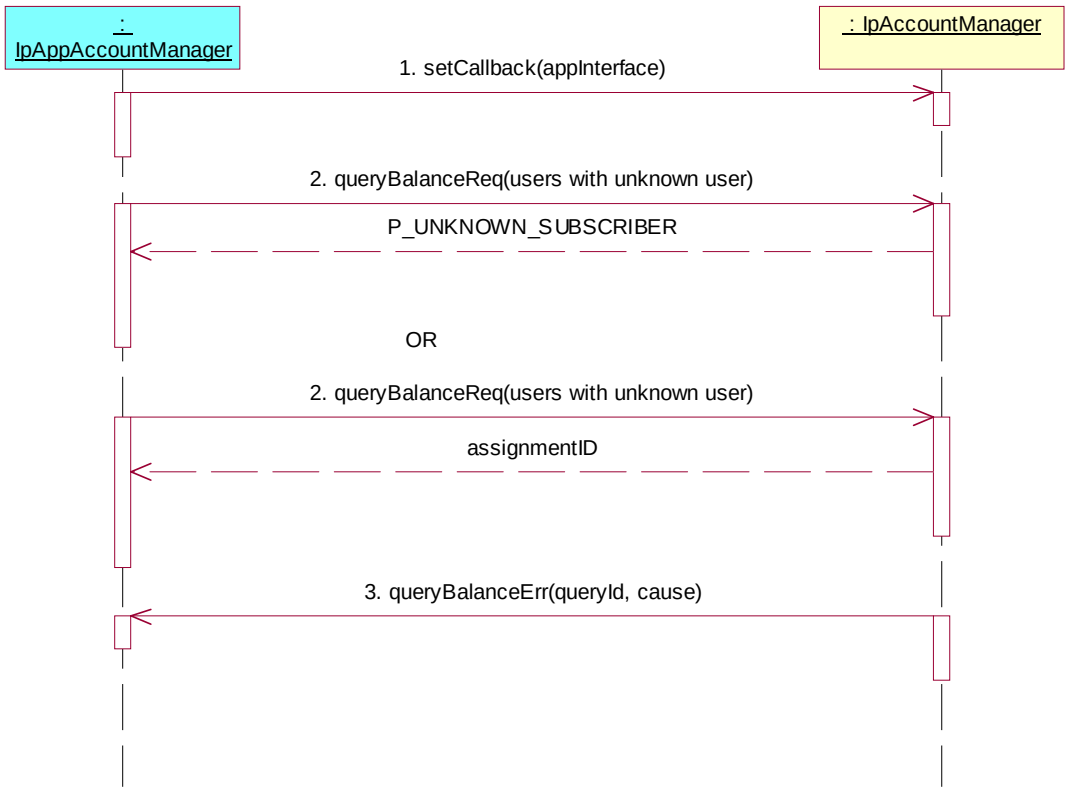
Preamble: Registration of the IUT (Account Manager SCF) and the tester (application) to the framework. The tester must have obtained a reference to an instance of the IpAccountManager interface through selecting that service and signing the required service agreement.

Test Sequence:

1. Method call **setCallback()**
   Parameters: appInterface with value of tester's (Application's) IpAppAccountManager interface
   Check: no exception is returned

2. Method call **queryBalanceReq()**
   Parameters: users, with unknown user
   Check: P_UNKNOWN_SUBSCRIBER is returned,
   or
   Check: valid value of TpAssignmentID is returned

3. Check: **queryBalanceErr()** method is called on the tester's (Application) **IpAppAccountManager** interface
   Parameters: queryId, cause with error code P_BALANCE_QUERY_UNKNOWN_SUBSCRIBER.



Test AM_12

Summary: **IpAccountManager** all methods, successful.

Reference: ES 202 915-11 [1], clause 8.1.

Precondition: enableNotifications supported.

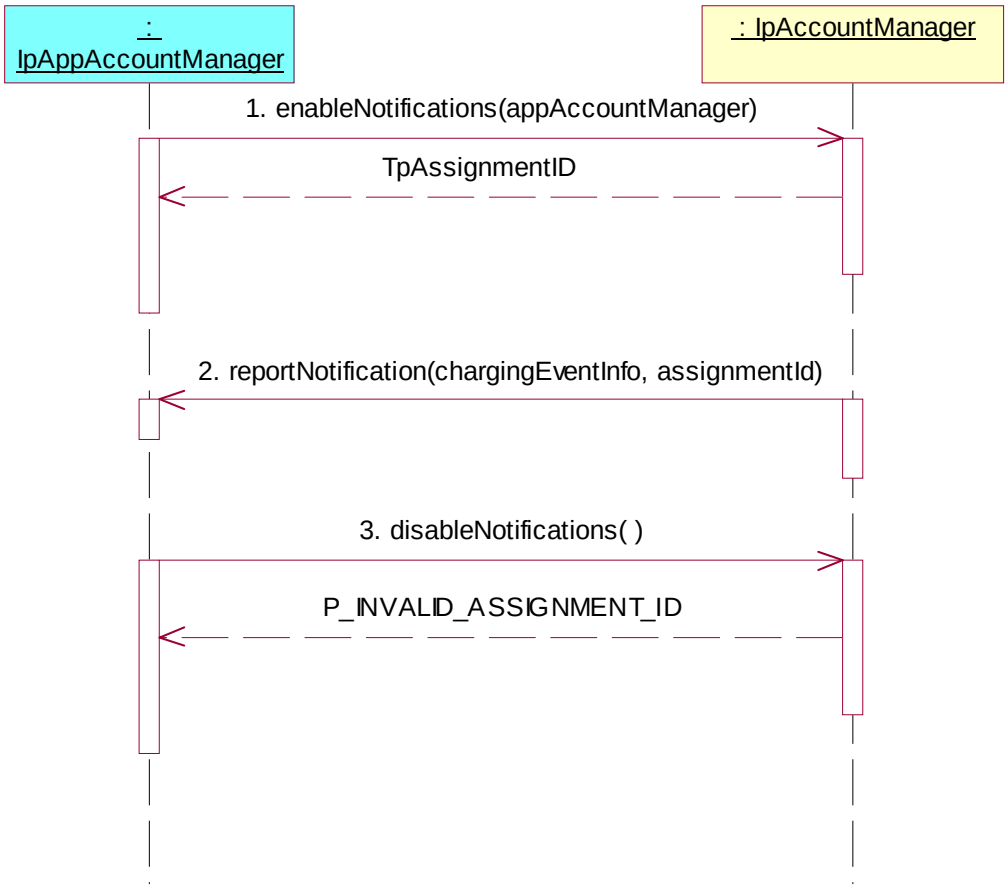
Preamble: Registration of the IUT (Account Manager SCF) and the tester (application) to the framework. The tester must have obtained a reference to an instance of the IpAccountManager interface through selecting that service and signing the required service agreement.

Test Sequence:

1. Method call **enableNotifications()**
   Parameters: appAccountManager
   Check: valid value of TpAssignmentID is returned

2. Triggered action: cause IUT to call **reportNotification()** method on the tester's (Application) **IpAppAccountManager** interface.
   Parameters: chargingEventInfo, assignmentId

3. Method call **disableNotifications()**
   Parameters: none
   Check: no exception is returned



5.3 TPs for the application using the Account Management SCF

All ICS items referred to in this clause are as specified in ES 202 363 [2] unless indicated otherwise by another numbered reference.

All parameters specified in method calls are valid unless specified.

The procedures to trigger the application to call methods in the SCF are dependant on the underlying network architecture and are out of the scope of the present document. Those method calls are preceded by the words "Triggered action".

5.3.1 Account Management

Test AM_APP_01

Summary: **IpAppAccountManager**, create and accept charging event notification.

Reference: ES 202 915-11 [1], clause 8.1.

Precondition: IUT capable of invoking **createNotification().**

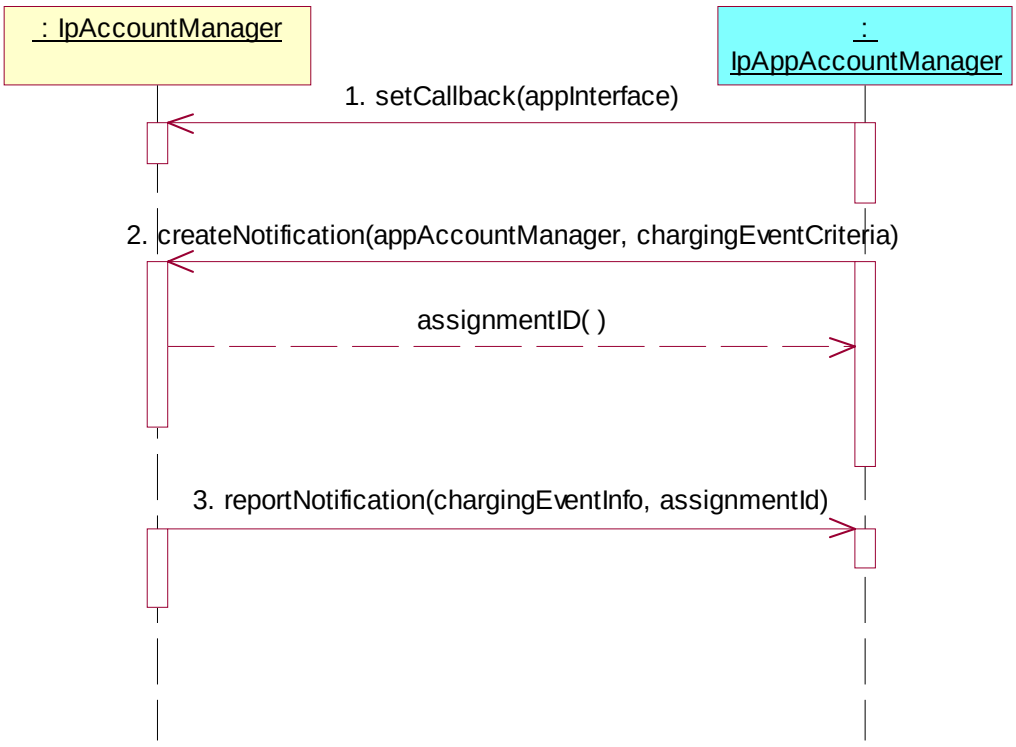
Preamble: Registration of the IUT (application) and the tester (Account Manager SCF) to the framework. The tester must have obtained a reference to an instance of the IpAppAccountManager interface through selecting that service and signing the required service agreement.

Test Sequence:

1. Triggered action: cause IUT to call **setCallback()** method on the tester's (SCF) **IpAccountManager** interface.
   Parameters: appInterface with value of IUT's (application's) IpAppAccountManager interface

2. Triggered action: cause IUT to call **createNotification()** method on the tester's (SCF) **IpAccountManager** interface.
   Parameters: appAccountManager, chargingEventCriteria

3. Method call **reportNotification()**
   Parameters: chargingEventInfo, assignmentId
   Check: no exception is returned



Test AM_APP_02

Summary: **IpAppAccountManager**, create and destroy charging event notification.

Reference: ES 202 915-11 [1], clause 8.1.

Precondition: IUT capable of invoking **createNotification()** and **destroyNotification().**

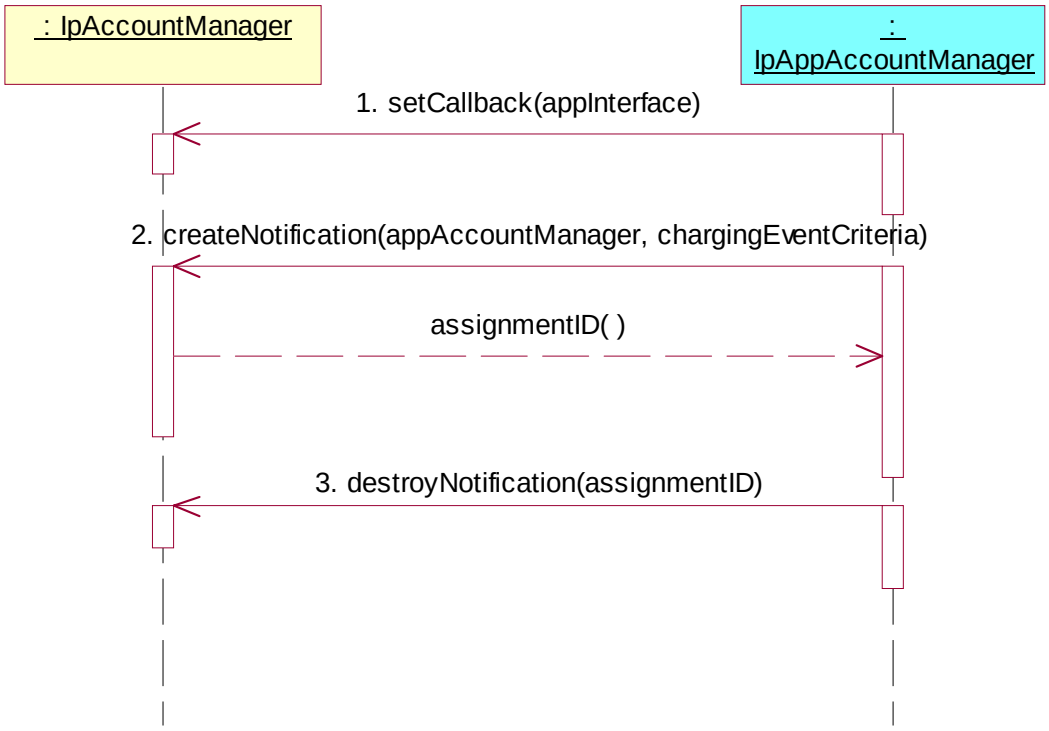
Preamble: Registration of the IUT (application) and the tester (Account Manager SCF) to the framework. The tester must have obtained a reference to an instance of the IpAppAccountManager interface through selecting that service and signing the required service agreement.

Test Sequence:

1. Triggered action: cause IUT to call **setCallback()** method on the tester's (SCF) **IpAccountManager** interface.
   Parameters: appInterface with value of IUT's (application's) IpAppAccountManager interface

2. Triggered action: cause IUT to call **createNotification()** method on the tester's (SCF) **IpAccountManager** interface.
   Parameters: appAccountManager, chargingEventCriteria

3. Triggered action: cause IUT to call **destroyNotification**() method on the tester's (SCF) **IpAccountManager** interface.
   Parameters: assignmentID received in response to **createNotification()**.



Test AM_APP_03

Summary: **IpAppAccountManager**, create and change charging event notification.

Reference: ES 202 915-11 [1], clause 8.1.

Precondition: IUT capable of invoking **createNotification()** and **changeNotification().**

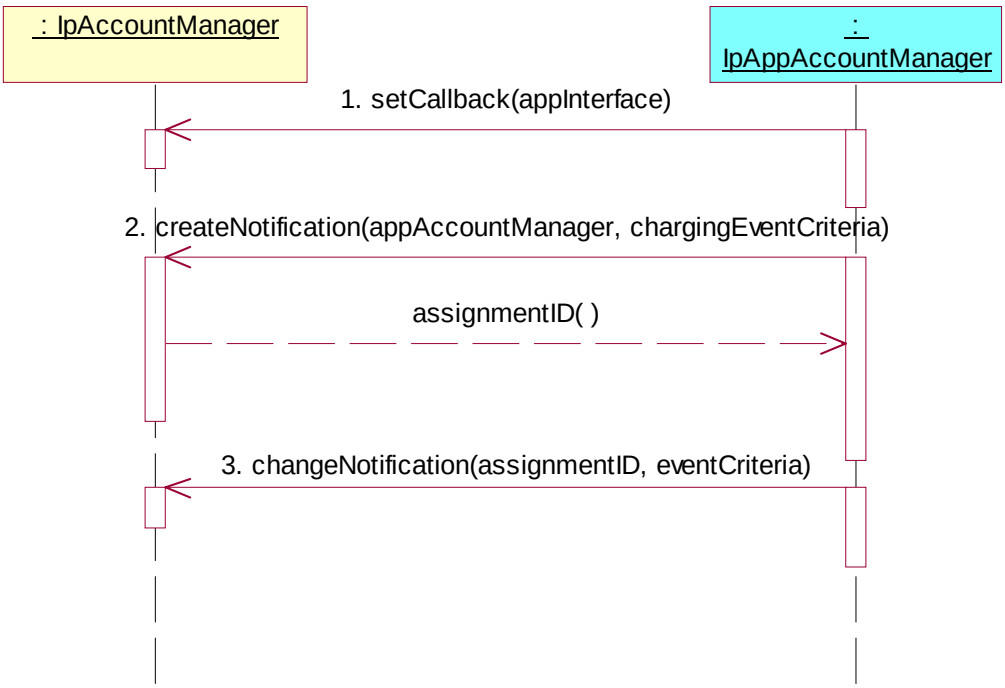
Preamble: Registration of the IUT (application) and the tester (Account Manager SCF) to the framework. The tester must have obtained a reference to an instance of the IpAppAccountManager interface through selecting that service and signing the required service agreement.

Test Sequence:

1. Triggered action: cause IUT to call **setCallback()** method on the tester's (SCF) **IpAccountManager** interface.
   Parameters: appInterface with value of IUT's (application's) IpAppAccountManager interface

2. Triggered action: cause IUT to call **createNotification()** method on the tester's (SCF) **IpAccountManager** interface.
   Parameters: appAccountManager, chargingEventCriteria

3. Triggered action: cause IUT to call **changeNotification**() method on the tester's (SCF) **IpAccountManager** interface.
   Parameters: assignmentID received in response to **createNotification()**, eventCriteria



Test AM_APP_04

Summary: **IpAppAccountManager**, create and query charging event notification.

Reference: ES 202 915-11 [1], clause 8.1.

Precondition: IUT capable of invoking **createNotification()** and **getNotification().**

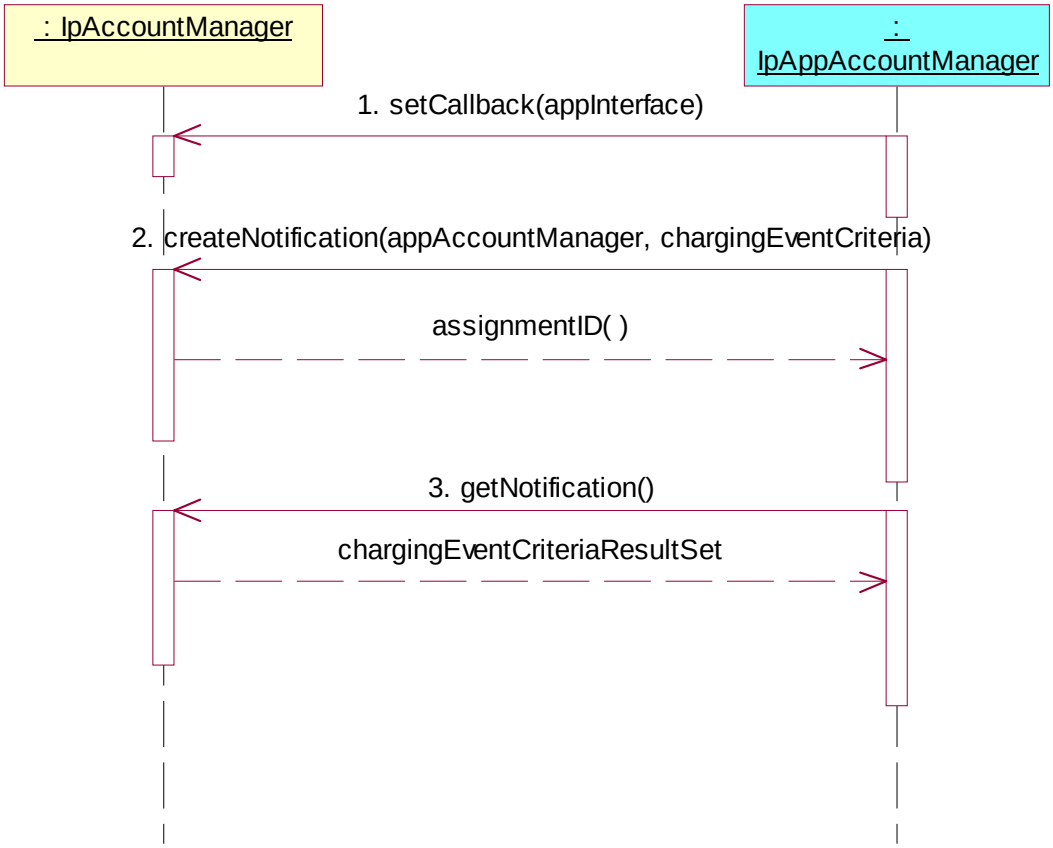
Preamble: Registration of the IUT (application) and the tester (Account Manager SCF) to the framework. The tester must have obtained a reference to an instance of the IpAppAccountManager interface through selecting that service and signing the required service agreement.

Test Sequence:

1. Triggered action: cause IUT to call **setCallback()** method on the tester's (SCF) **IpAccountManager** interface.
   Parameters: appInterface with value of IUT's (application's) IpAppAccountManager interface

2. Triggered action: cause IUT to call **createNotification()** method on the tester's (SCF) **IpAccountManager** interface.
   Parameters: appAccountManager, chargingEventCriteria

3. Triggered action: cause IUT to call **getNotification**() method on the tester's (SCF) **IpAccountManager** interface.
   Parameters: none



Test AM_APP_05

Summary: **IpAppAccountManager**, retrieve transaction history, successful.

Reference: ES 202 915-11 [1], clause 8.1.

Precondition: IUT capable of invoking **retrieveTransactionHistoryReq().**

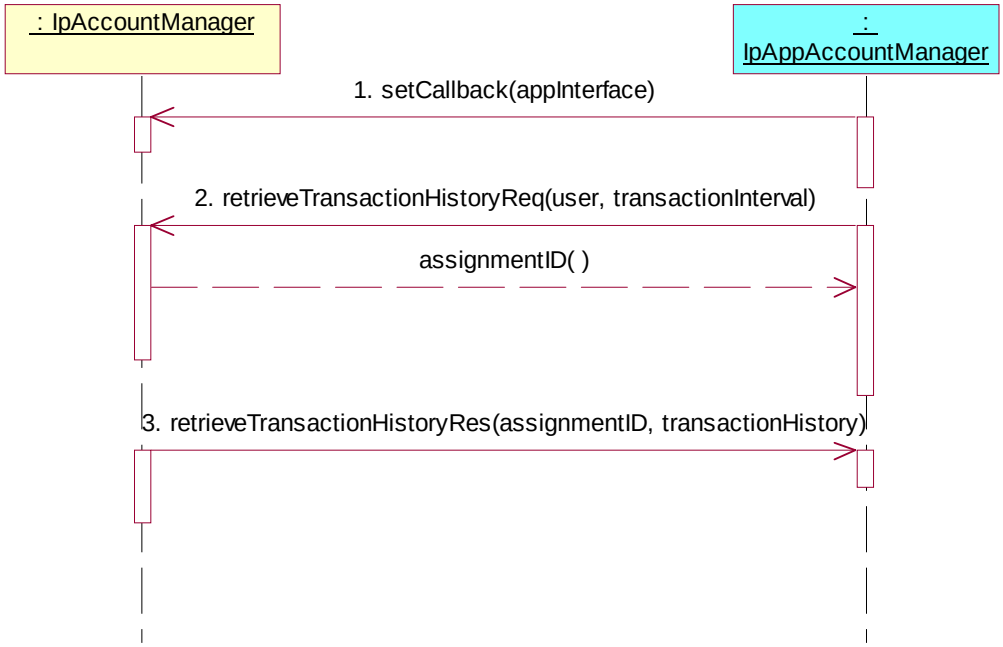
Preamble: Registration of the IUT (application) and the tester (Account Manager SCF) to the framework. The tester must have obtained a reference to an instance of the IpAppAccountManager interface through selecting that service and signing the required service agreement.

Test Sequence:

1. Triggered action: cause IUT to call **setCallback()** method on the tester's (SCF) **IpAccountManager** interface.
   Parameters: appInterface with value of IUT's (application's) IpAppAccountManager interface

2. Triggered action: cause IUT to call **retrieveTransactionHistoryReq()** method on the tester's (SCF) **IpAccountManager** interface.
   Parameters: user, transactionInterval

3. Method call **retrieveTransactionHistoryRes()**
   Parameters: assignmentID, transactionHistory
   Check: no exception is returned



Test AM_APP_06

Summary: **IpAppAccountManager**, retrieve transaction history, unsuccessful.

Reference: ES 202 915-11 [1], clause 8.1.

Precondition: IUT capable of invoking **retrieveTransactionHistoryReq().**

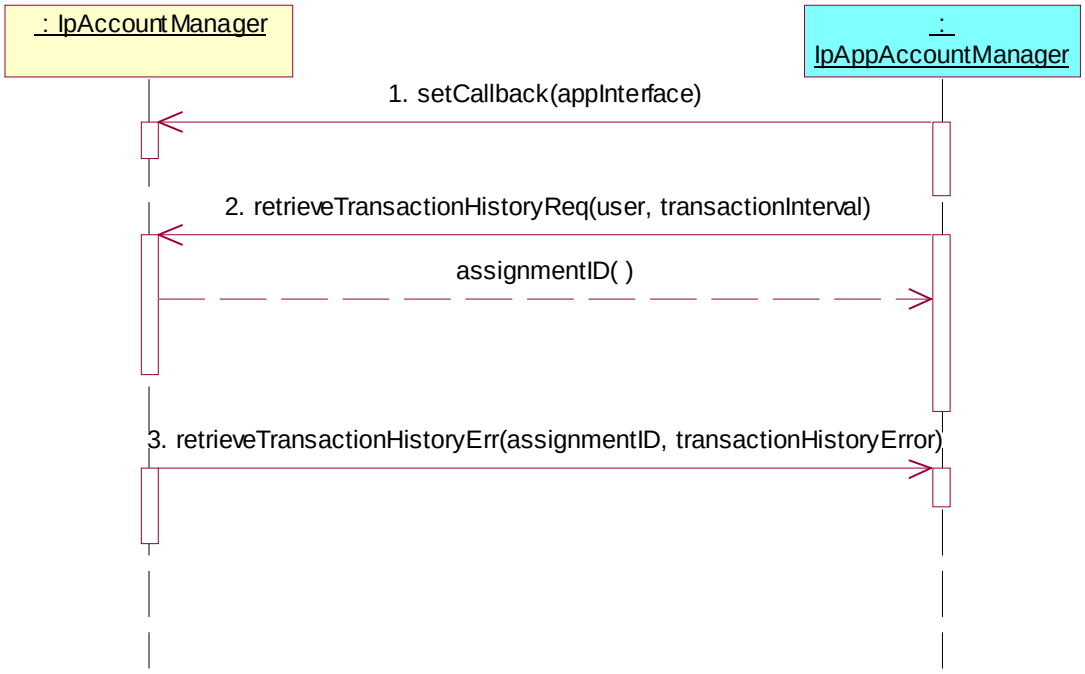
Preamble: Registration of the IUT (application) and the tester (Account Manager SCF) to the framework. The tester must have obtained a reference to an instance of the IpAppAccountManager interface through selecting that service and signing the required service agreement.

Test Sequence:

1. Triggered action: cause IUT to call **setCallback()** method on the tester's (SCF) **IpAccountManager** interface.
   Parameters: appInterface with value of IUT's (application's) IpAppAccountManager interface

2. Triggered action: cause IUT to call **retrieveTransactionHistoryReq()** method on the tester's (SCF) **IpAccountManager** interface.
   Parameters: user, transactionInterval

3. Method call **retrieveTransactionHistoryErr()**
   Parameters: assignmentID, transactionHistoryError
   Check: no exception is returned



Test AM_APP_07

Summary: **IpAppAccountManager**, query account balance, successful.

Reference: ES 202 915-11 [1], clause 8.1.

Precondition: IUT capable of invoking **queryBalanceReq().**

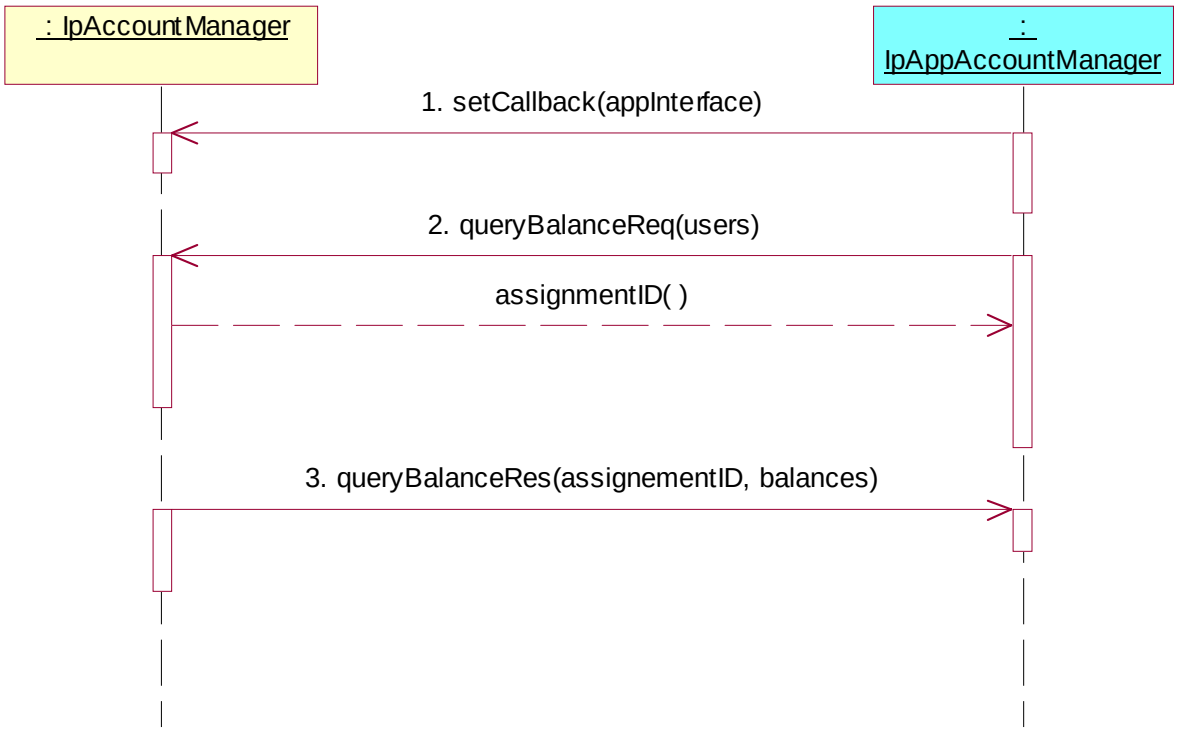
Preamble: Registration of the IUT (application) and the tester (Account Manager SCF) to the framework. The tester must have obtained a reference to an instance of the IpAppAccountManager interface through selecting that service and signing the required service agreement.

Test Sequence:

1. Triggered action: cause IUT to call **setCallback()** method on the tester's (SCF) **IpAccountManager** interface.
   Parameters: appInterface with value of IUT's (application's) IpAppAccountManager interface

2. Triggered action: cause IUT to call **queryBalanceReq()** method on the tester's (SCF) **IpAccountManager** interface.
   Parameters: users

3. Method call **queryBalanceRes()**
   Parameters: assignmentID, balances
   Check: no exception is returned



Test AM_APP_08

Summary: **IpAppAccountManager**, query account balance, unsuccessful.

Reference: ES 202 915-11 [1], clause 8.1.

Precondition: IUT capable of invoking **queryBalanceReq().**

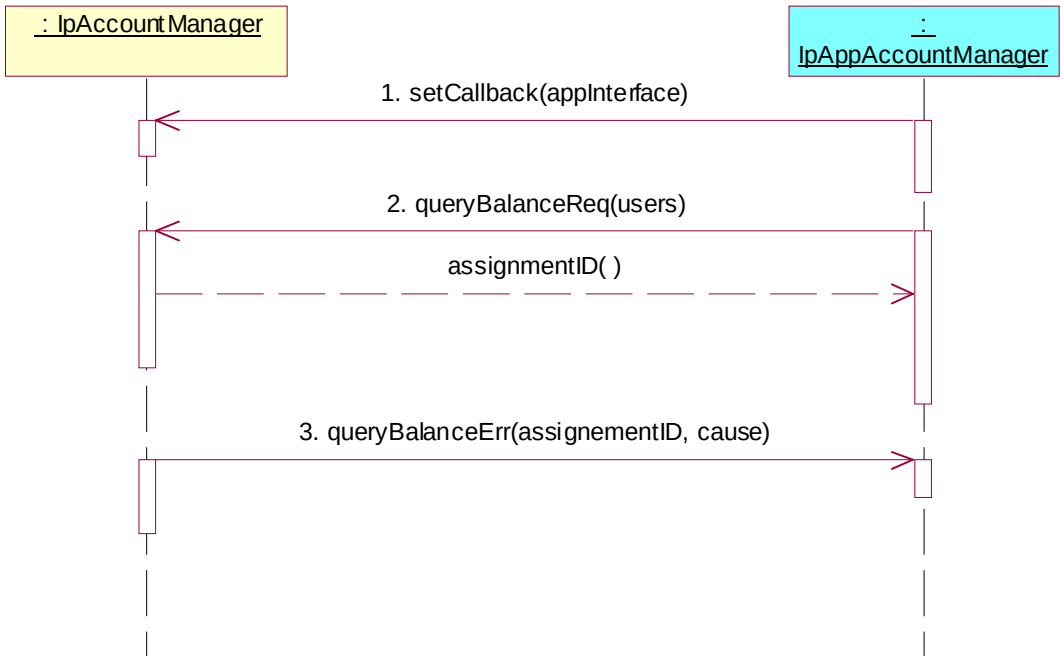
Preamble: Registration of the IUT (application) and the tester (Account Manager SCF) to the framework. The tester must have obtained a reference to an instance of the IpAppAccountManager interface through selecting that service and signing the required service agreement.

Test Sequence:

1. Triggered action: cause IUT to call **setCallback()** method on the tester's (SCF) **IpAccountManager** interface.
   Parameters: appInterface with value of IUT's (application's) IpAppAccountManager interface

2. Triggered action: cause IUT to call **queryBalanceReq()** method on the tester's (SCF) **IpAccountManager** interface.
   Parameters: users

3. Method call **queryBalanceErr()**
   Parameters: assignmentID, cause
   Check: no exception is returned



Test AM_APP_09

Summary: **IpAppAccountManager** all methods, successful.

Reference: ES 202 915-11 [1], clause 8.1.

Precondition: reportNotification supported.

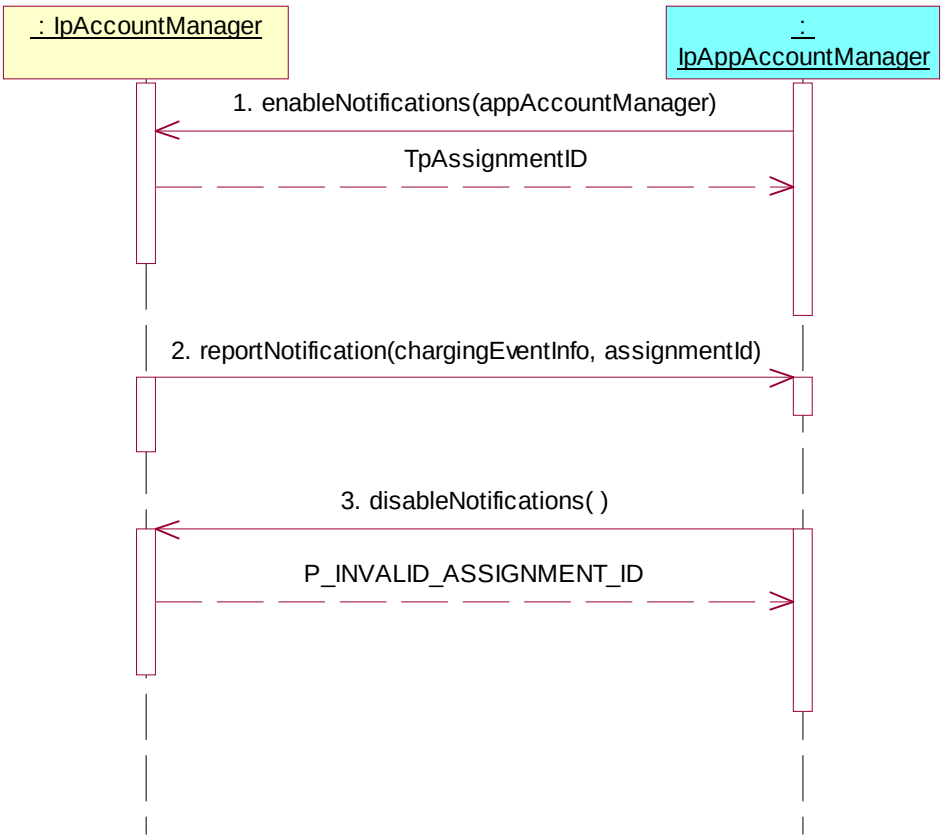
Preamble: Registration of the IUT (application) and the tester (Account Manager SCF) to the framework. The tester must have obtained a reference to an instance of the IpAppAccountManager interface through selecting that service and signing the required service agreement.

Test Sequence:

1. Triggered action: cause IUT to call **enableNotifications ()** method on the tester's (SCF) **IpAccountManager** interface.
   Parameters: appAccountManager

2. Method call **reportNotification ()**
   Parameters: chargingEventInfo, assignmentId
   Check: no exception is returned

3. Triggered action: cause IUT to call **disableNotifications ()** method on the tester's (SCF) **IpAccountManager** interface.
   Parameters: none


: IpAccountManager
: IpAppAccountManager
1. enableNotifications(appAccountManager)
TpAssignmentID
2. reportNotification(chargingEventInfo, assignmentId)
3. disableNotifications( )
P_INVALID_ASSIGNMENT_ID

History

Document history		
V1.1.1	January 2005	Membership Approval Procedure MV 20050311: 2005-01-11 to 2005-03-11
V1.1.1	March 2005	Publication