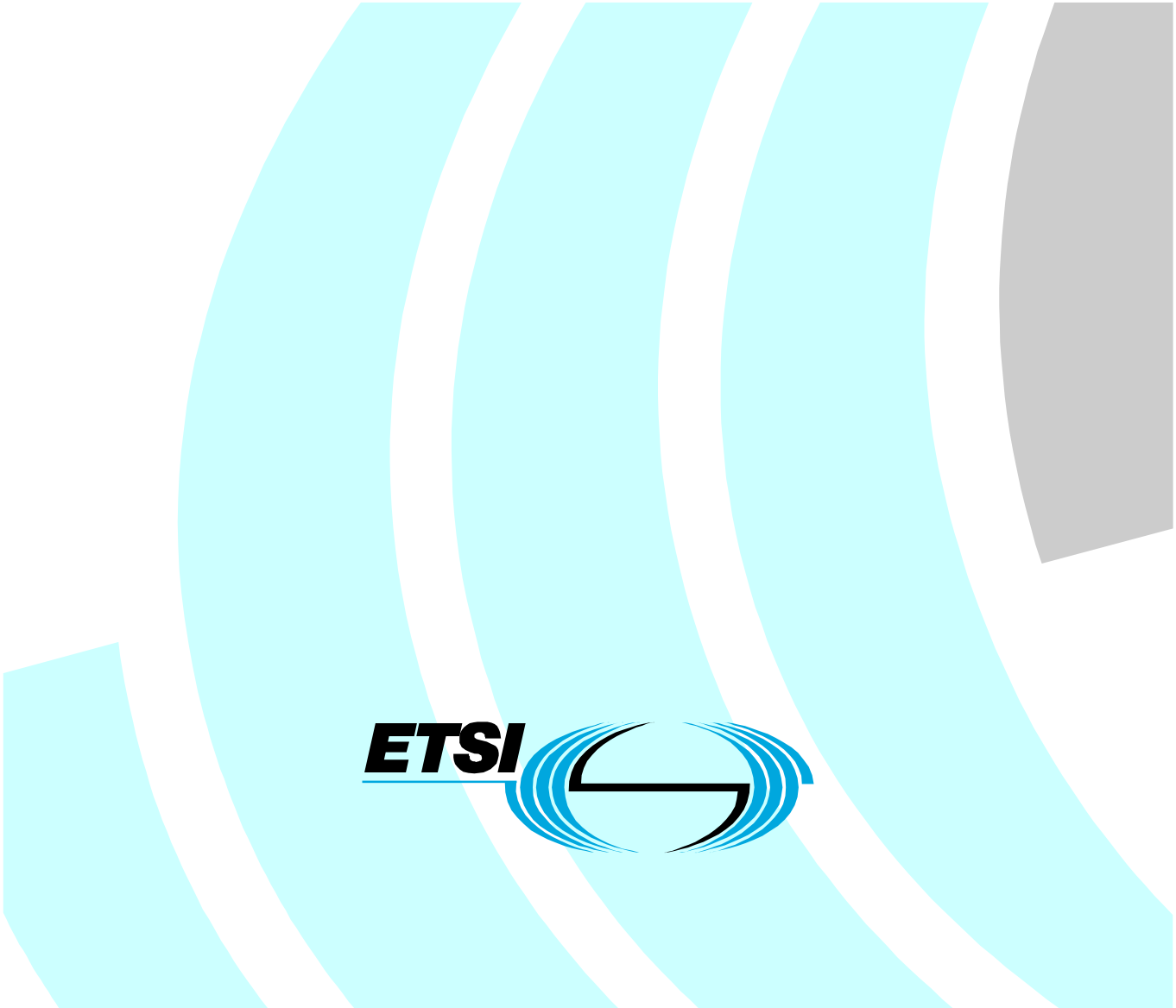
Final draft ETSI ES 202 196-4 V1.1.1 (2003-06)

ETSI Standard

Open Service Access (OSA); Application Programming Interface (API); Test Suite Structure and Test Purposes (TSS&TP); Part 4: Call control SCF



Reference
DES/SPAN-120088-4

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, send your comment to:
editor@etsi.org

Copyright Notification

No part may be reproduced except as authorized by written permission.
The copyright and the foregoing restriction extend to reproduction in all media.

© European Telecommunications Standards Institute 2003.
All rights reserved.

DECT™, **PLUGTESTS**™ and **UMTS**™ are Trade Marks of ETSI registered for the benefit of its Members.
TIPHON™ and the **TIPHON logo** are Trade Marks currently being registered by ETSI for the benefit of its Members.
3GPP™ is a Trade Mark of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners.

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 4 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 Call Control SCF of the Application Programming Interface (API) for Open Service Access (OSA) defined in ES 201 915-4 [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 201 915-4: "Open Service Access (OSA); Application Programming Interface (API); Part 4: Call Control SCF".

[2] ETSI ES 202 170: "Open Service Access (OSA); Application Programming Interface (API); Implementation Conformance Statement (ICS) proforma specification for Framework and SCFs".

[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 202 196-1: "Open Service Access (OSA); Application Programming Interface (API); Test Suite Structure and Test Purposes (TSS&TP); Part 1: Overview".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in ES 201 915-4 [1], ISO/IEC 9646-1 [3] and 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:

API	Application Programming Interface
ATM	Abstract Test Method
ATS	Abstract Test Suite
CC	Call Control
CCC	Conference Call Control service
GCC	Generic Call Control service
GCC	Generic Call Control Service
ICS	Implementation Conformance Statement
IUT	Implementation Under Test
IXIT	Implementation eXtra Information for Testing
LT	Lower Tester
MMCC	MultiMedia Call Control service
MPCC	MultiParty Call Control service
OSA	Open Service Access
SUT	System Under Test
TP	Test Purpose
TSS	Test Suite Structure

4 Test Suite Structure (TSS)

Call Control (CC):

- Generic Call Control Service (GCC)
- MultiParty Call Control Service (MPCC)
- MultiMedia Call Control Service (MMCC)
- Conference Call Control Service (CCC)

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:	"CC" for **C**all **C**ontrol part of Call Control SCF
<group>	= group number:	two character field representing the group reference according to TSS
<nn>	= sequential number:	(01-99)

5.1.2 Source of TP definition

The TPs are based on ES 201 915-4 [1].

5.1.3 Test strategy

As the base standard ES 201 915-4 [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 202 170 [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 Call Control SCF

All ICS items referred to in this clause are as specified in ES 202 170 [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 this test specification. Those method calls are preceded by the words "Triggered action".

5.2.1 Generic Call Control

5.2.1.1 IpCallControlManager

5.2.1.1.1 Mandatory, valid behaviour

Test GCC_IPCALLCONTROLMANAGER _01

Summary: IpCallControlManager, all mandatory methods, successful

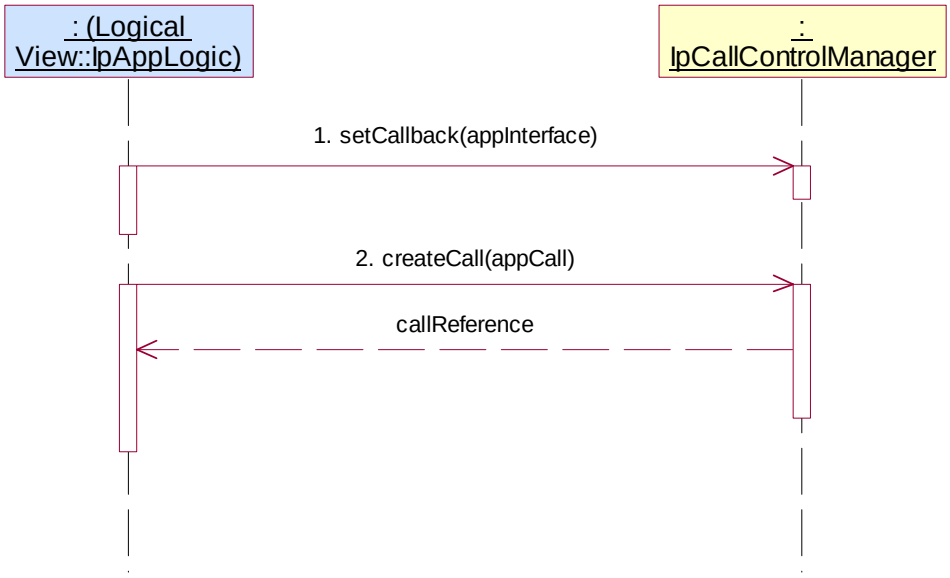
Reference: ES 201 915-4 [1], clause 6.3.1

Condition: createCall method is supported.

Test Sequence:

1. Method call **setCallback()** on IpCallControlManager
   Parameters: valid, not null, value of appInterface parameter
   Check: no exception is returned

2. Method call **createCall()**
   Parameters: valid appCall
   Check: valid value of TpCallIdentifier is returned



Test GCC_IPCALLCONTROLMANAGER _02

Summary: IpCallControlManager, all mandatory methods, successful

Reference: ES 201 915-4 [1], clause 6.3.1

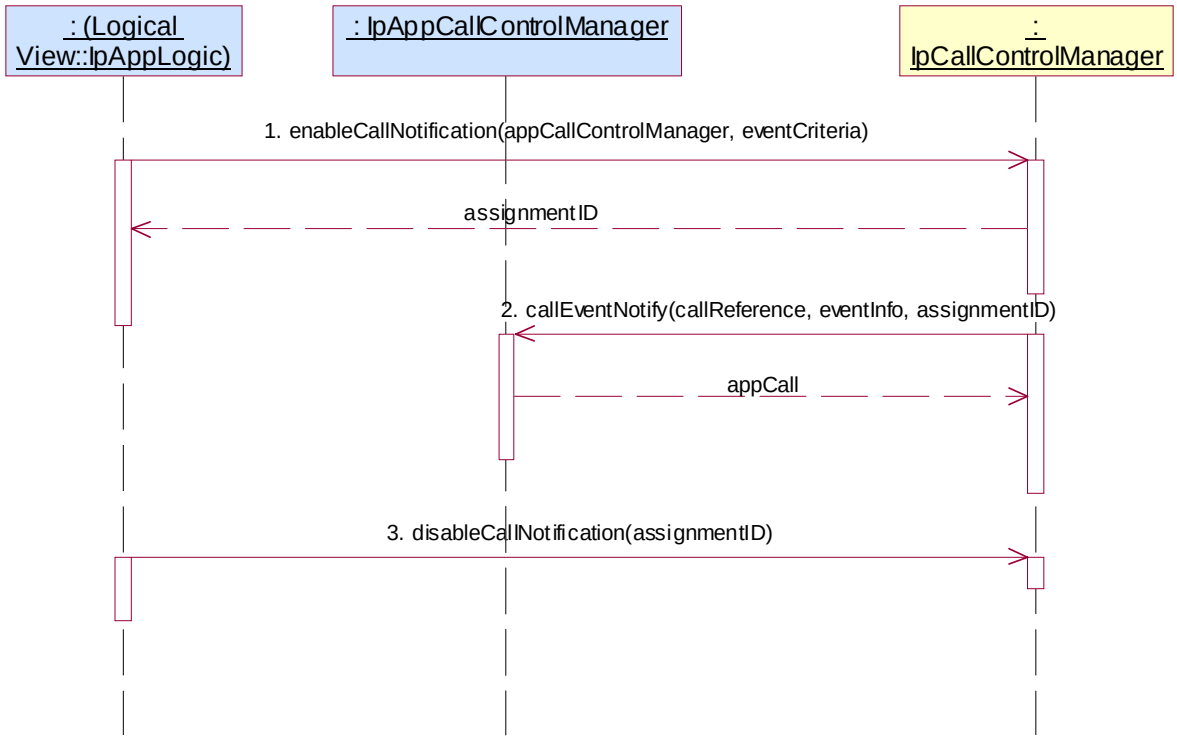
Condition: enableCallNotification method is supported.

Test Sequence:

1. Method call **enableCallNotification()**
   Parameters: appCallControlManager with valid, not null, value, valid eventCriteria
   Check: valid value of TpAssignmentID is returned

2 Triggered action: cause IUT to call **callEventNotify()**method on the tester's (Application) **IpAppCallControlManager** interface
   Parameters: valid callReference, valid eventInfo, assignmentID returned in 1

3. Method call **disableCallNotification**()
   Parameters: assignmentID returned in 1
   Check: no exception is returned



Test GCC_IPCALLCONTROLMANAGER _03

Summary: IpCallControlManager, all mandatory methods, successful

Reference: ES 201 915-4 [1], clause 6.3.1

Condition: enableCallNotification method is supported.

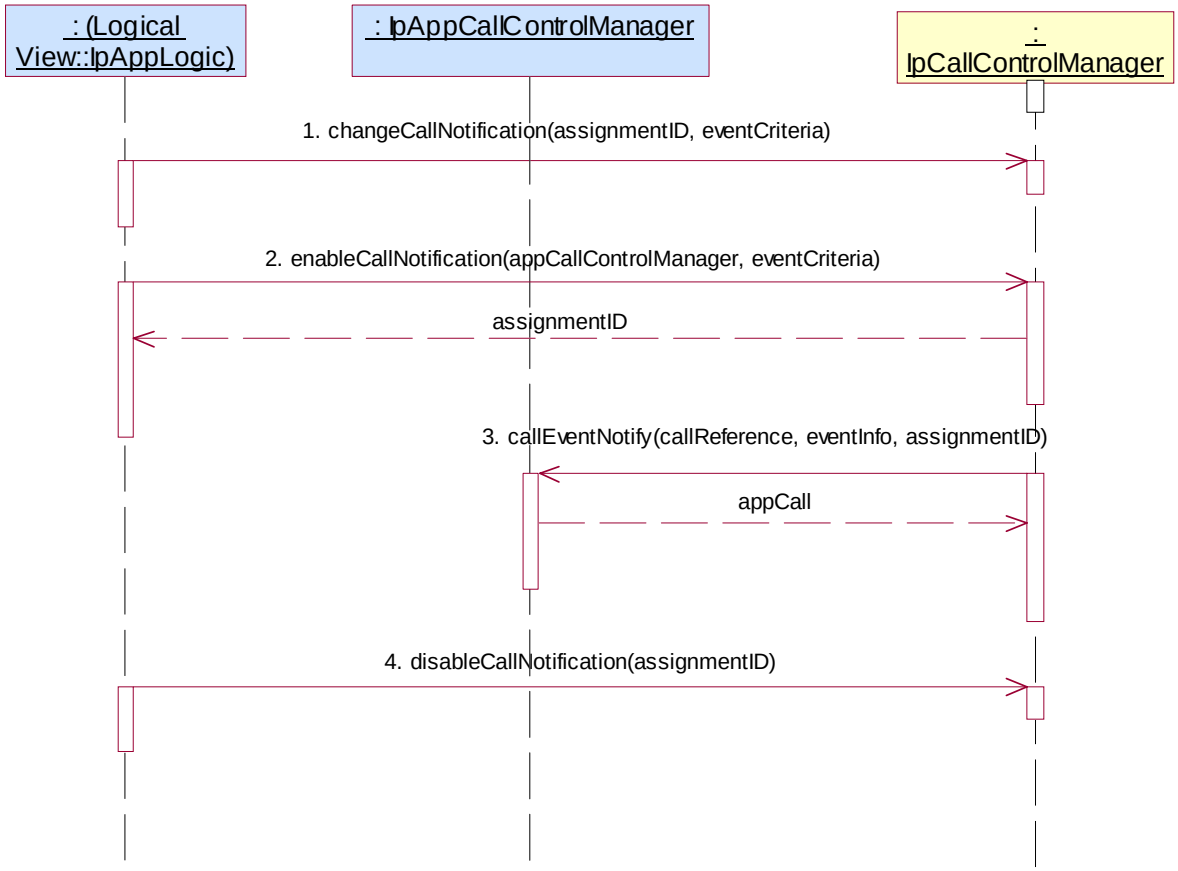
Test Sequence:

1. Method call **setCallback()** on IpCallControlManager
   Parameters: valid, non-null, value of appInterface parameter
   Check: no exception is returned

2. Method call **enableCallNotification()**
   Parameters: appCallControlManager with null, value, valid eventCriteria
   Check: valid value of TpAssignmentID is returned

3. Triggered action: cause IUT to call **callEventNotify()**method on the tester's (Application) **IpAppCallControlManager** interface.
   Parameters: valid callReference, valid eventInfo, assignmentID returned in 2.

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



5.2.1.1.2 Mandatory, invalid behaviour

Test GCC_IPCALLCONTROLMANAGER _04

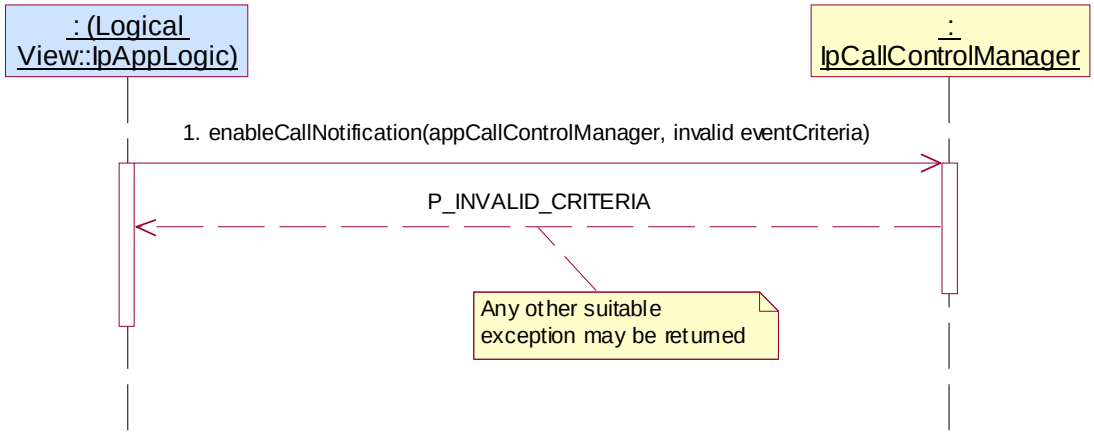
Summary: IpCallControlManager, enableCallNotification, P_INVALID_CRITERIA

Reference: ES 201 915-4 [1], clause 6.3.1

Condition: enableCallNotification method is supported.

Test Sequence:

1. Method call **enableCallNotification()**
   Parameters: valid appCallControlManager, invalid eventCriteria
   Check: P_INVALID_CRITERIA, or another suitable exception, is returned.



Test GCC_IPCALLCONTROLMANAGER _05

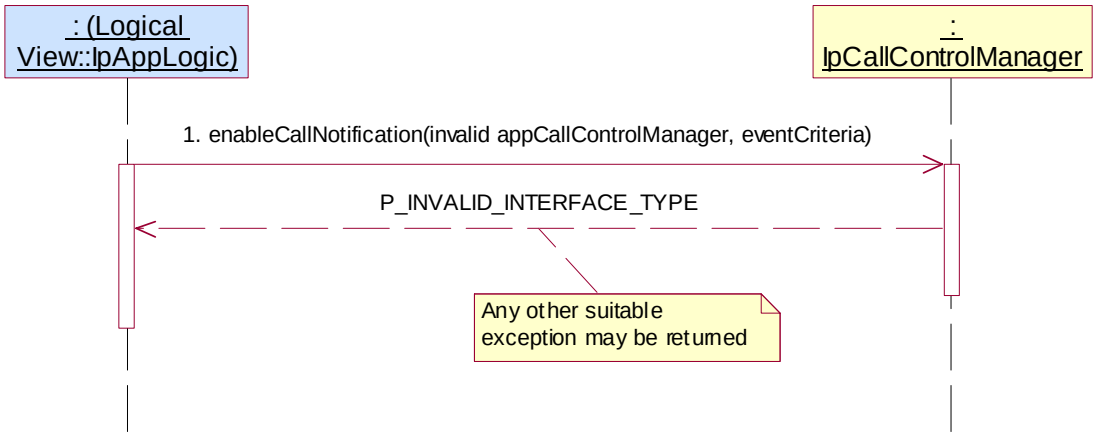
Summary: IpCallControlManager, enableCallNotification, P_INVALID_INTERFACE_TYPE

Reference: ES 201 915-4 [1], clause 6.3.1

Condition: enableCallNotification method is supported.

Test Sequence:

1. Method call **enableCallNotification()**
   Parameters: invalid appCallControlManager, valid eventCriteria
   Check: P_INVALID_INTERFACE_TYPE, or another suitable exception, is returned



Test GCC_IPCALLCONTROLMANAGER _06

Summary: IpCallControlManager, disableCallNotification, P_INVALID_ASSIGNMENT_ID

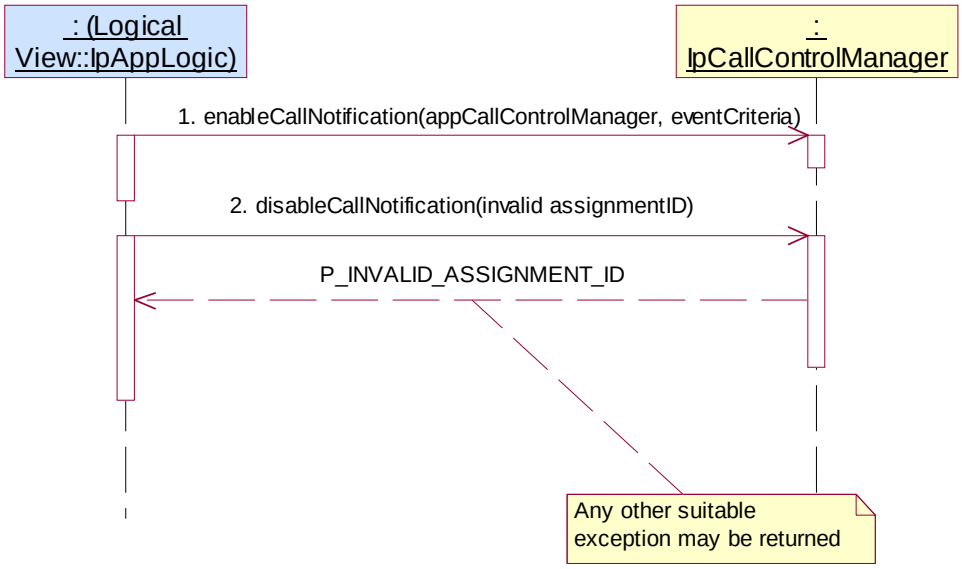
Reference: ES 201 915-4 [1], clause 6.3.1

Condition: disableCallNotification is supported.

Test Sequence:

1. Method call **enableCallNotification()**
   Parameters: appCallControlManager with valid, not null, value, valid eventCriteria
   Check: valid value of TpAssignmentID is returned

2. Method call **disableCallNotification()**
   Parameters: invalid assignmentID
   Check: P_INVALID_ASSIGNMENT_ID, or another suitable exception, is returned



Test GCC_IPCALLCONTROLMANAGER _07

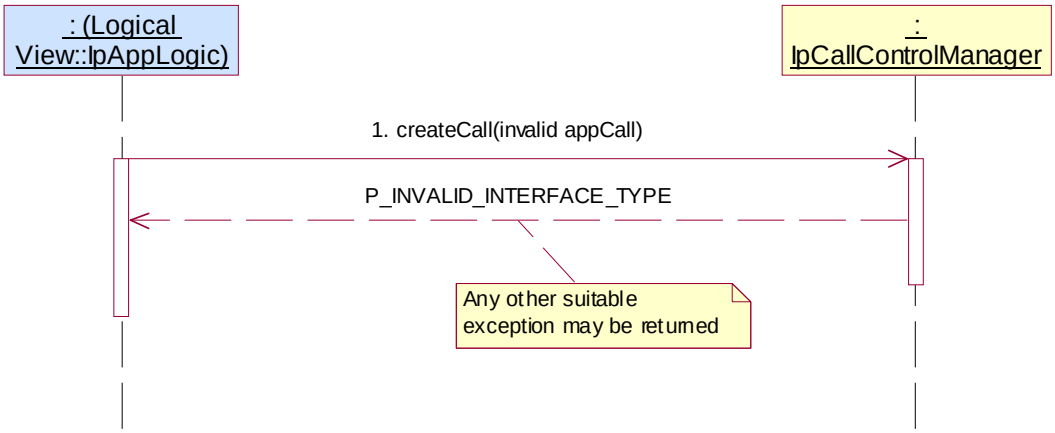
Summary: IpCallControlManager, createCall, P_INVALID_INTERFACE_TYPE

Reference: ES 201 915-4 [1], clause 6.3.1

Condition: createCall method is supported.

Test Sequence:

1. Method call **createCall()**
   Parameters: invalid value of appCall
   Check: P_INVALID_INTERFACE_TYPE, or another suitable exception, is returned



5.2.1.1.3 Optional, valid behaviour

Test GCC_IPCALLCONTROLMANAGER _08

Summary: IpCallControlManager, getCriteria, successful

Reference: ES 201 915-4 [1], clause 6.3.1

Condition: enableCallNotification and getCriteria methods are supported.

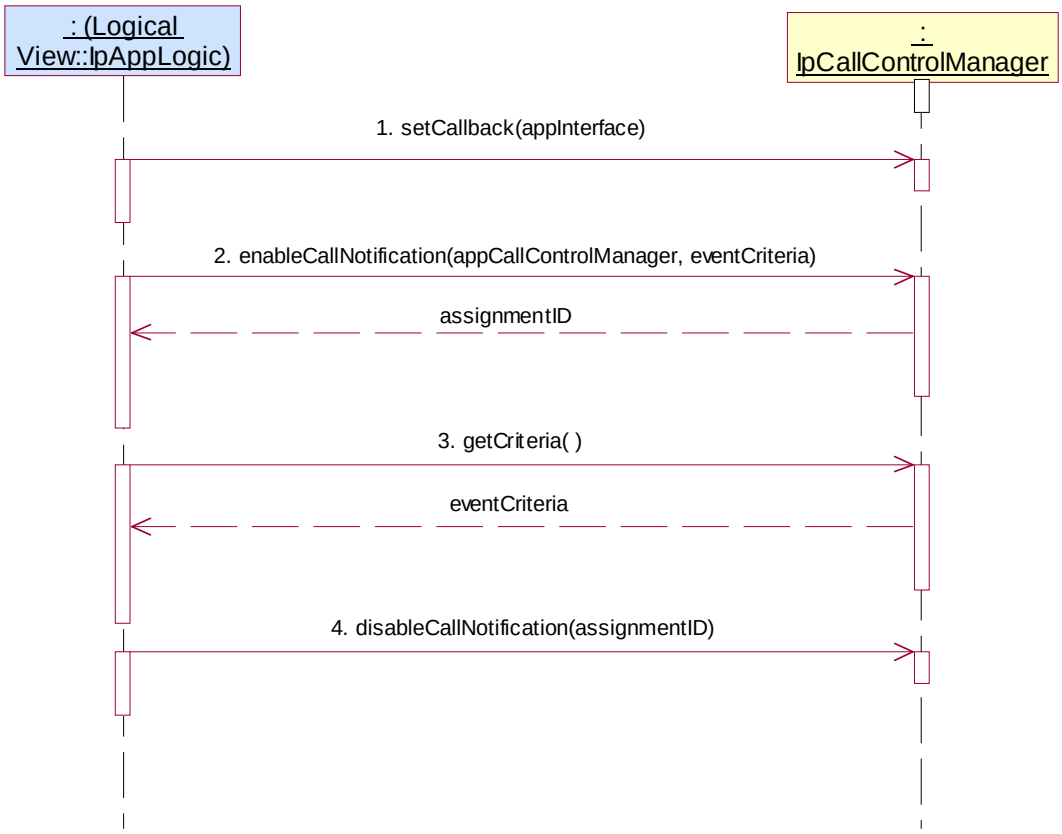
Test Sequence:

1. Method call **setCallback()** on IpCallControlManager
   Parameters: valid, non-null, value of appInterface parameter
   Check: no exception is returned

2. Method call **enableCallNotification()**
   Parameters: appCallControlManager with null value, valid eventCriteria
   Check: valid value of TpAssignmentID is returned

3. Method call **getCriteria()**
   Parameters: None
   Check: valid value of TpCallEventCriteriaResultSet is returned where eventCriteria given in 1. is included as a value of this TpCallEventCriteriaResultSet

4. Method call **disableCallNotification**()
   Parameters: assignmentID returned in 1.
   Check: no exception is returned



Test GCC_IPCALLCONTROLMANAGER _09

Summary: IpCallControlManager, changeCallNotification, successful

Reference: ES 201 915-4 [1], clause 6.3.1

Condition: enableCallNotification, getCriteria and changeCallNotification methods are supported.

Preamble: Application has a reference interface used for callbacks.

Test Sequence:

1. Method call **enableCallNotification()**
   Parameters: appCallControlManager with valid, not null, value, valid eventCriteria
   Check: valid value of TpAssignmentID is returned

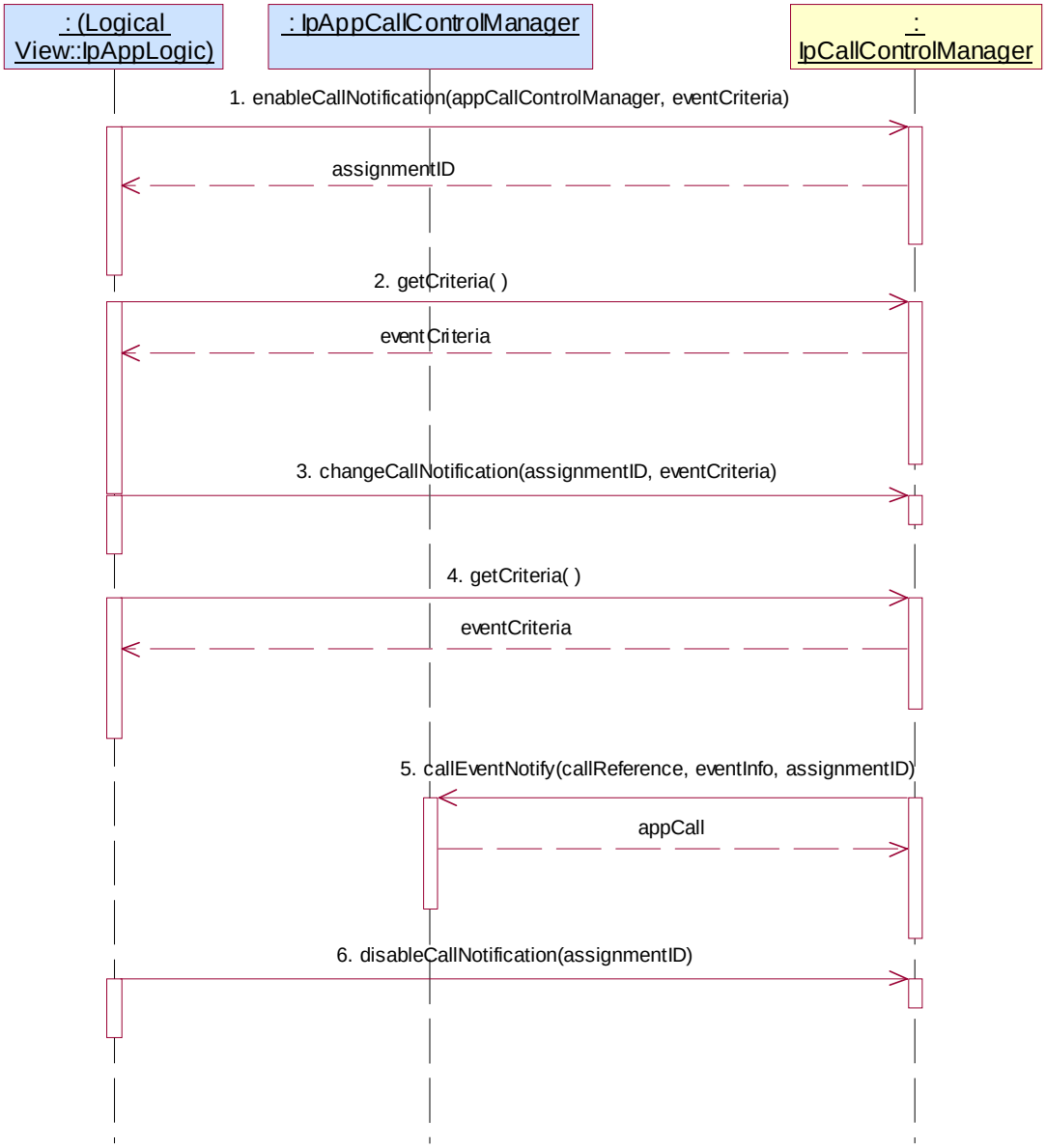
2. Method call **getCriteria()**
   Parameters: None
   Check: valid value of TpCallEventCriteriaResultSet is returned where eventCriteria given in 1. is included as a value of this TpCallEventCriteriaResultSet

3. Method call **changeCallNotification**()
   Parameters: assignmentID returned in 1., valid eventCriteria different from this given in 1.
   Check: no exception is returned

4. Method call **getCriteria()**
   Parameters: None
   Check: valid value of TpCallEventCriteriaResultSet is returned where eventCriteria given in 3. is included as a value of this TpCallEventCriteriaResultSet

5 Triggered action: cause IUT to call **callEventNotify()** method on the tester's (Application) **IpAppCallControlManager** interface.
   Parameters: valid callReference, valid eventInfo, assignmentID returned in 1.

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


: (Logical View::IpAppLogic)
: IpAppCallControlManager
: IpCallControlManager
1. enableCallNotification(appCallControlManager, eventCriteria)
assignmentID
2. getCriteria( )
eventCriteria
3. changeCallNotification(assignmentID, eventCriteria)
4. getCriteria( )
eventCriteria
5. callEventNotify(callReference, eventInfo, assignmentID)
appCall
6. disableCallNotification(assignmentID)

Test GCC_IPCALLCONTROLMANAGER _10

Summary: IpCallControlManager, all methods, successful

Reference: ES 201 915-4 [1], clause 6.3.1

Condition: enableCallNotification, setCallLoadControl, callOverLoadEncountered and callOverLoadCeased methods are supported.

Test Sequence:

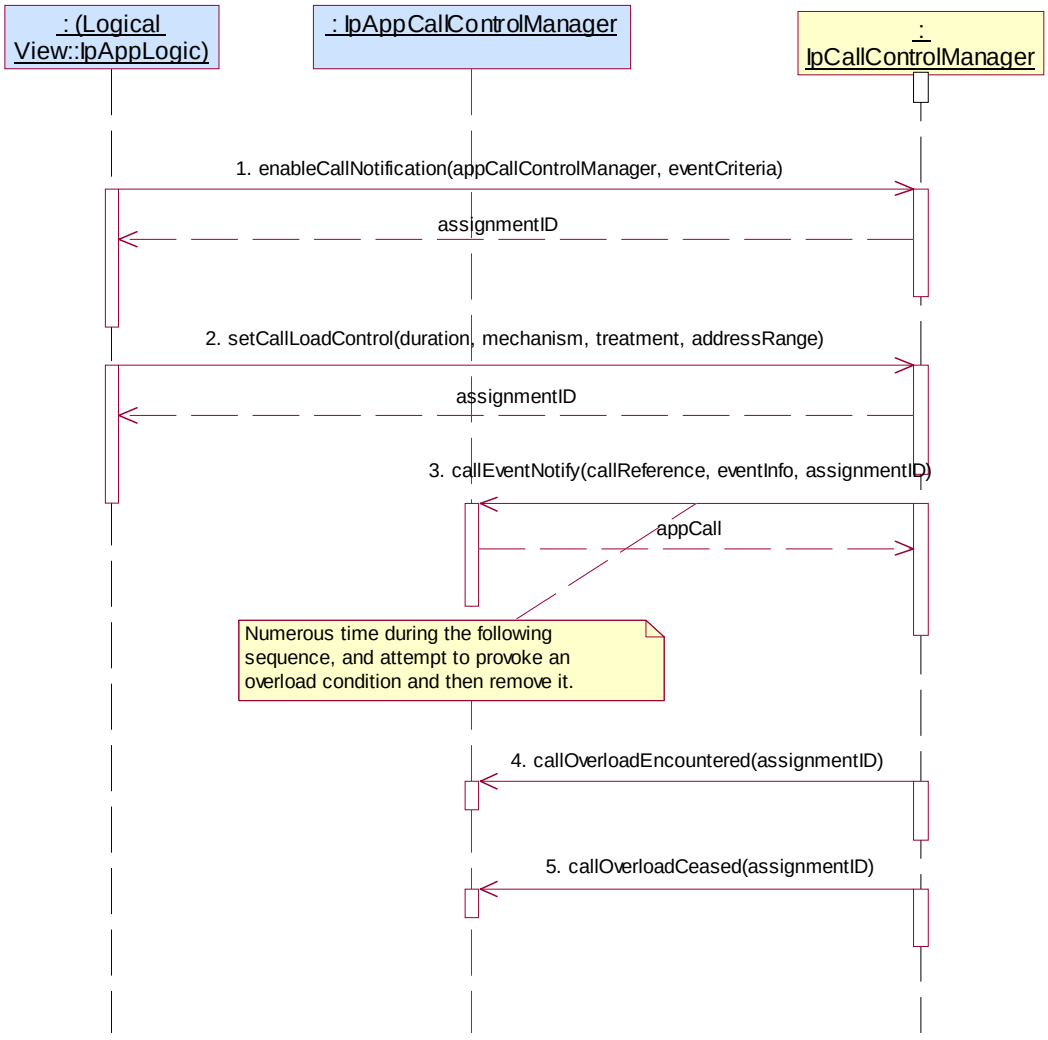
1. Method call **enableCallNotification()**
   Parameters: appCallControlManager with valid, not null, value, valid eventCriteria
   Check: valid value of TpAssignmentID is returned

2. Method call **setCallLoadControl()**
   Parameters: valid duration, valid mechanism, valid treatment, valid addressRange
   Check: valid value of TpAssignmentID is returned

3. Triggered action: cause IUT to call callEventNotify() numerous times during the following sequence, and attempt to provoke an overload condition and then remove it.

4. Triggered action: cause IUT to call **callOverLoadEncountered()** method on the tester's (Application) **IpAppCallControlManager** interface.
   Parameters: valid assignmentID

5. Triggered action: cause IUT to call **callOverLoadCeased()** method on the tester's (Application) **IpAppCallControlManager** interface.
   Parameters: valid assignmentID



Test GCC_IPCALLCONTROLMANAGER _11

Summary: IpCallControlManager, all methods, successful

Reference: ES 201 915-4 [1], clause 6.3.1

Condition: enableCallNotification method is supported.

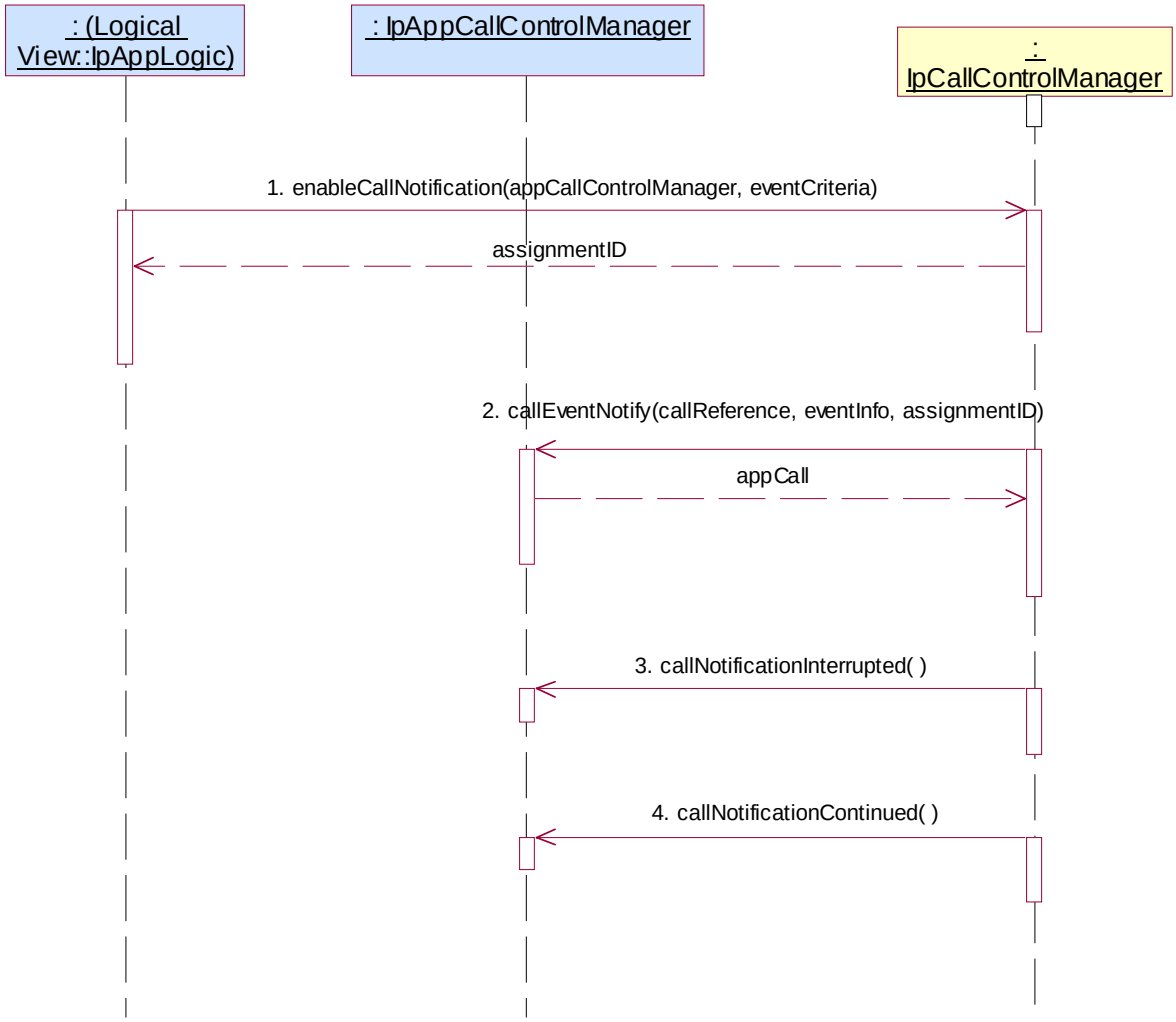
Test Sequence:

1. Method call **enableCallNotification()**
   Parameters: appCallControlManager with valid, not null, value, valid eventCriteria
   Check: valid value of TpAssignmentID is returned

2. Triggered action: cause IUT to call **callEventNotify**()method on the tester's (Application) **IpAppCallControlManager** interface.

3. Triggered action: cause IUT to call **callNotificationInterrupted()** method on the tester's (Application) **IpAppCallControlManager** interface.
   Parameters: None

4. Triggered action: cause IUT to call **callNotificationContinue()** method on the tester's (Application) **IpAppCallControlManager** interface.
   Parameters: None



Test GCC_IPCALLCONTROLMANAGER _12

Summary: IpCallControlManager, all methods, successful

Reference: ES 201 915-4 [1], clause 6.3.1

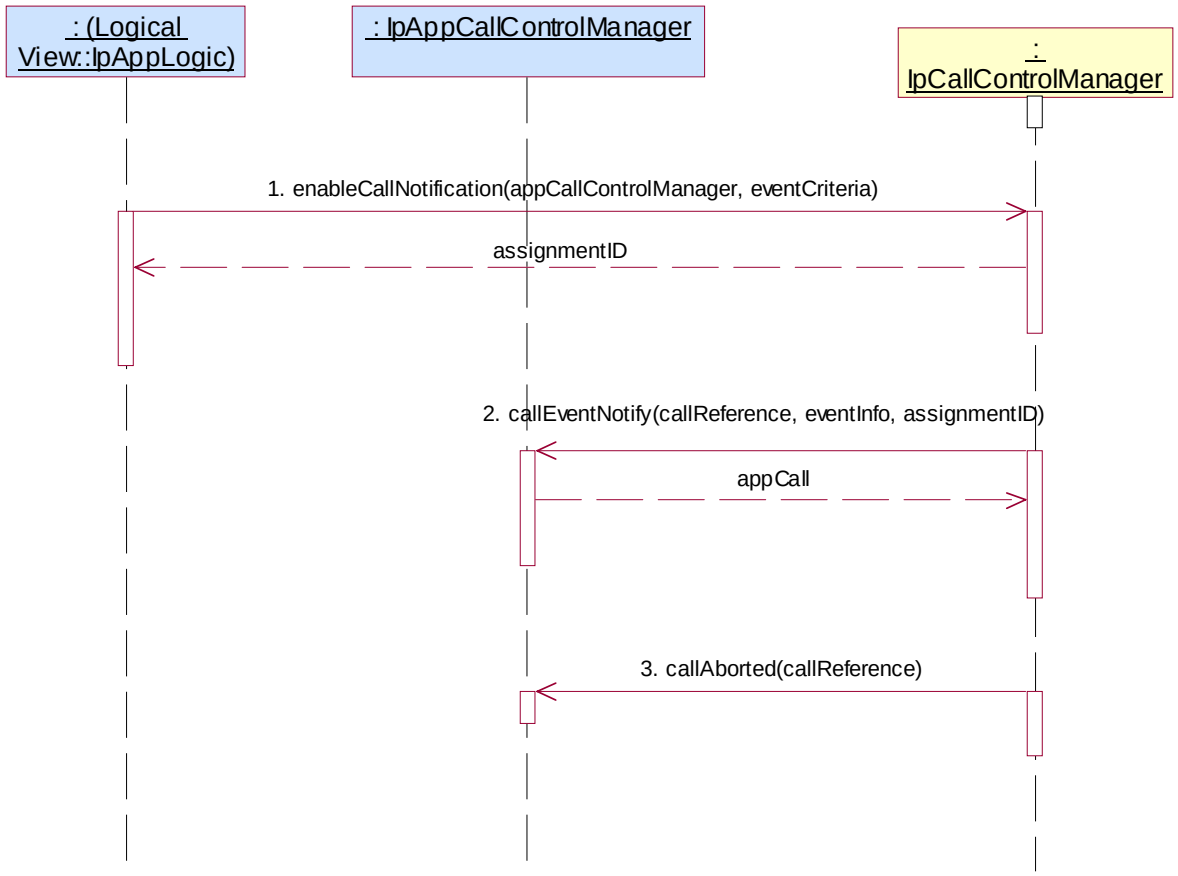
Condition: enableCallNotification and callAborted methods are supported.

Test Sequence:

1. Method call **enableCallNotification()**
   Parameters: appCallControlManager with valid, not null, value, valid eventCriteria
   Check: valid value of TpAssignmentID is returned

2. Triggered action: cause IUT to call **callEventNotify**() method on the tester's (Application) **IpAppCallControlManager** interface.

3 Triggered action: cause IUT to call **callAborted()** method on the tester's (Application) **IpAppCallControlManager** interface.
   Parameters: valid assignmentID as reported in callEventNotify.



5.2.1.1.4 Optional, invalid behaviour

Test GCC_IPCALLCONTROLMANAGER _13

Summary: IpCallControlManager, changeCallNotification, P_INVALID_ASSIGNMENT_ID

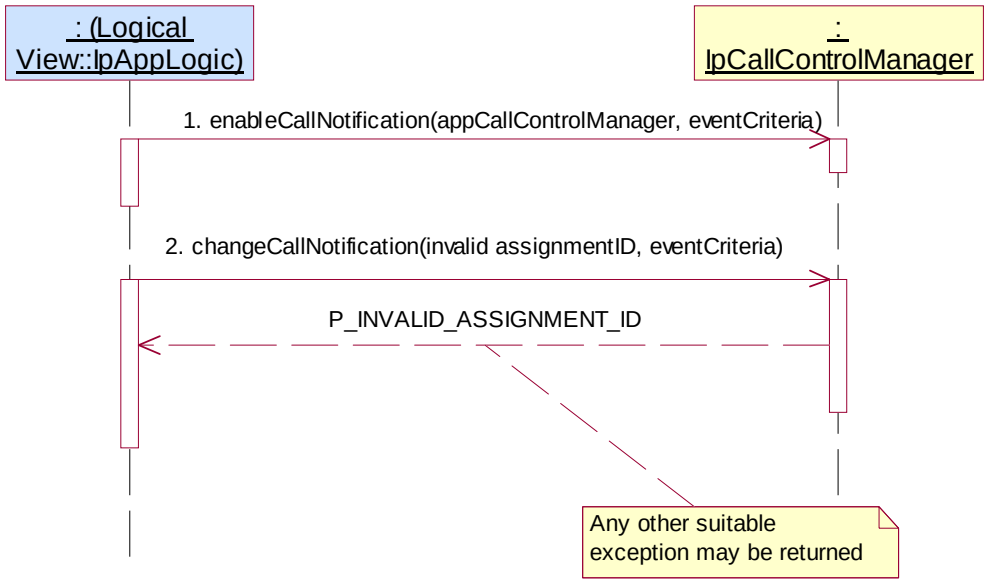
Reference: ES 201 915-4 [1], clause 6.3.1

Condition: changeCallNotification is supported.

Test Sequence:

1. Method call **enableCallNotification()**
   Parameters: appCallControlManager with valid, not null, value, valid eventCriteria
   Check: valid value of TpAssignmentID is returned

2. Method call **changeCallNotification()**
   Parameters: invalid assignmentID, valid eventCriteria
   Check: P_INVALID_ASSIGNMENT_ID, or another suitable exception, is returned



Test GCC_IPCALLCONTROLMANAGER _14

Summary: IpCallControlManager, changeCallNotification, P_INVALID_CRITERIA

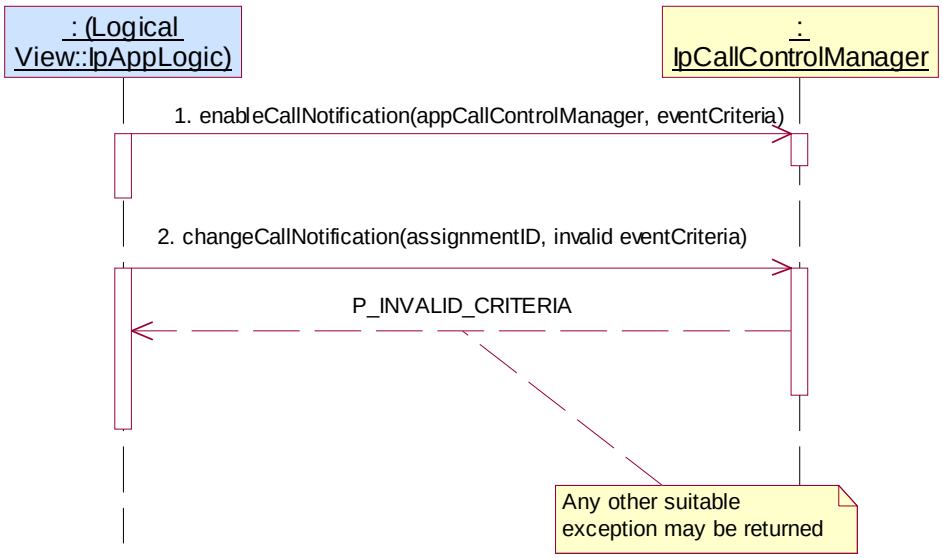
Reference: ES 201 915-4 [1], clause 6.3.1

Condition: enableCallNotification and changeCallNotification methods are supported.

Test Sequence:

1. Method call **enableCallNotification()**
   Parameters: appCallControlManager with null value, valid eventCriteria
   Check: valid value of TpAssignmentID is returned

2. Method call **changeCallNotification()**
   Parameters: assignmentID returned in 1., invalid eventCriteria
   Check: P_INVALID_CRITERIA, or another suitable exception, is returned



Test GCC_IPCALLCONTROLMANAGER _15

Summary: IpCallControlManager, setCallLoadControl, P_INVALID_ADDRESS

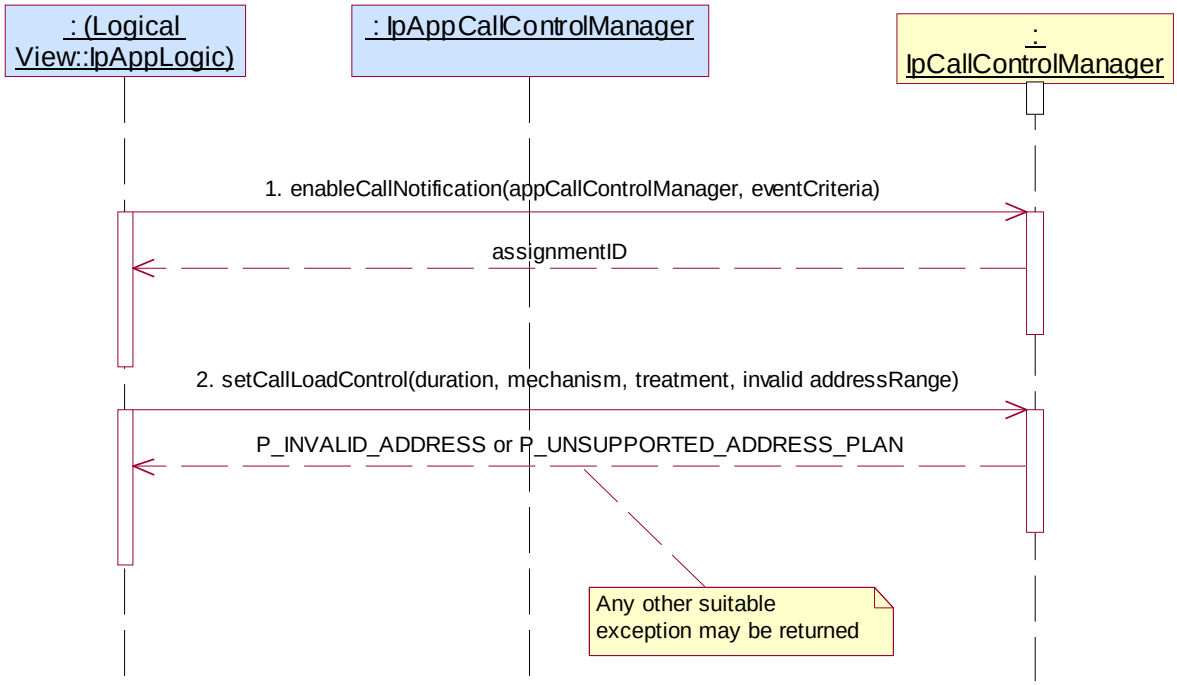
Reference: ES 201 915-4 [1], clause 6.3.1

Condition: enableCallNotification and setCallLoadControl method are supported.

Test Sequence:

1. Method call **enableCallNotification()**
   Parameters: appCallControlManager with valid, not null, value, valid eventCriteria
   Check: valid value of TpAssignmentID is returned

2. Method call **setCallLoadControl()**
   Parameters: valid duration, valid mechanism, valid treatment, invalid addressRange
   Check: P_INVALID_ADDRESS, P_UNSUPPORTED_ADDRESS_PLAN or another suitable exception, is returned



5.2.1.2 IpCall

5.2.1.2.1 Mandatory, valid behaviour

Test GCC_IPCALL _01

Summary: IpCall, all mandatory methods, successful

Reference: ES 201 915-4 [1], clauses 6.3.1 and 6.3.3.

Preamble: Application has a valid callSessionID returned by one of the two following sequence:

1. Method call **setCallback()** on IpCallControlManager
   Parameters: valid, non-null, value of appInterface parameter
   Check: no exception is returned

2. Method call **createCall()**
   Parameters: valid appCall
   Check: valid value of TpCallIdentifier is returned

or

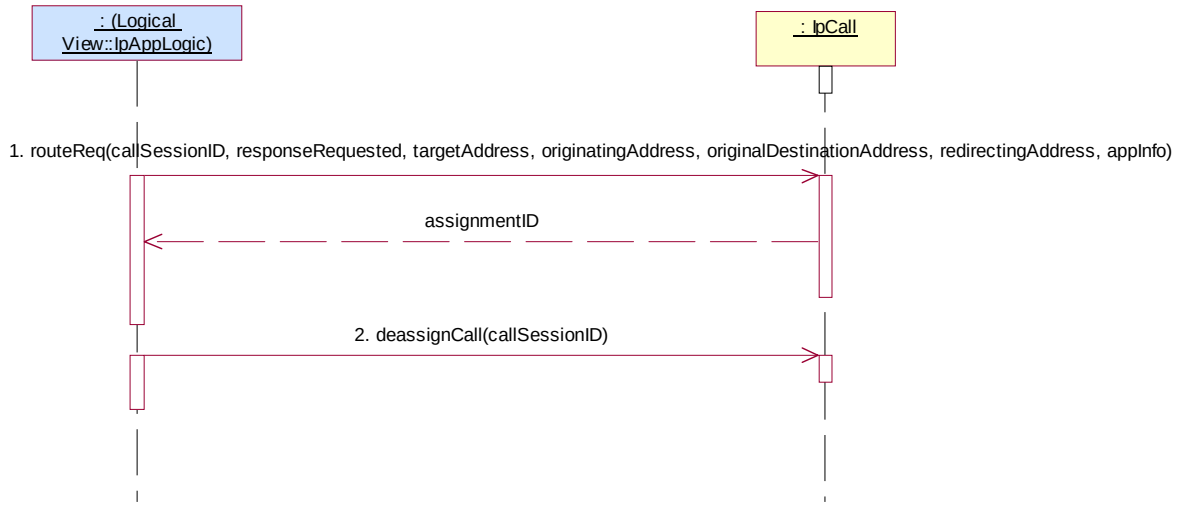
1. Method call **enableCallNotification()**
   Parameters: appCallControlManager with valid, not null, value, valid eventCriteria
   Check: valid value of TpAssignmentID is returned

2 Triggered action: cause IUT to call **callEventNotify()** method on the tester's (Application) **IpAppCallControlManager** interface.
   Parameters: valid callReference, valid eventInfo, assignmentID returned in 1.

Test Sequence:

1. Method call **routeReq()**
   Parameters: valid callSessionID reported in preamble, valid responseRequested, valid targetAddress, valid originatingAddress, valid originalDestinationAddress, valid redirectingAddress, valid appInfo
   Check: Valid value of TpSessionID is returned

2. Method call **deassignCall()**
   Parameters: valid callSessionID reported in preamble.
   Check: no exception is returned



Test GCC_IPCALL _02

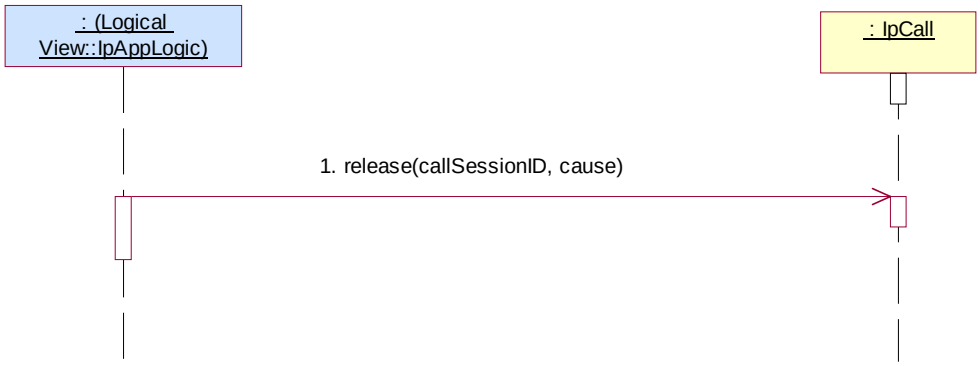
Summary: IpCall, all mandatory methods, successful

Reference: ES 201 915-4 [1], clauses 6.3.1 and 6.3.3.

Preamble: Same as GCC_IPCALL_01

Test Sequence:

1. Method call **release()**
   Parameters: valid callSessionID reported in preamble.
   Check: no exception is returned



5.2.1.2.2 Mandatory, invalid behaviour

Test GCC_IPCALL _03

Summary: IpCall, routeReq, P_INVALID_SESSION_ID

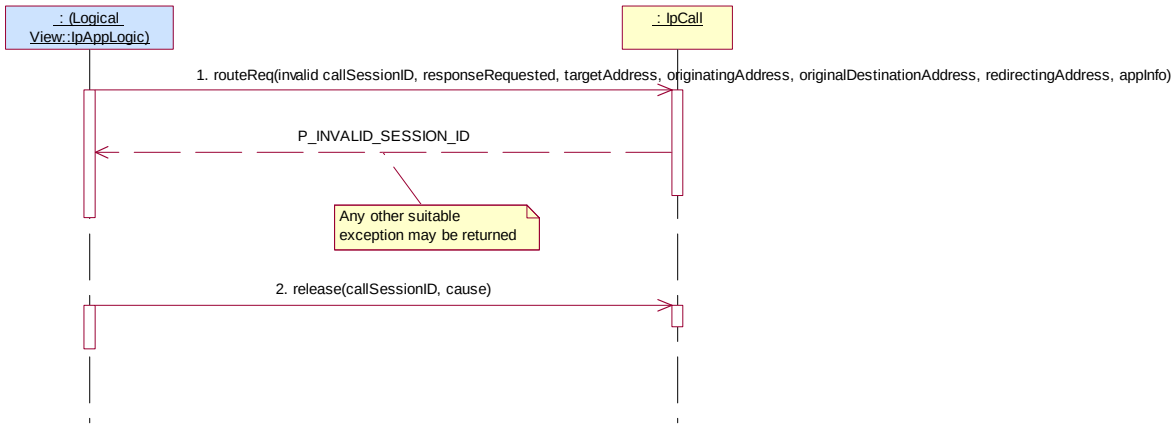
Reference: ES 201 915-4 [1], clause 6.3.3

Preamble: Same as GCC_IPCALL_01

Test Sequence:

1. Method call **routeReq()**
   Parameters: invalid callSessionID, valid responseRequested, valid targetAddress, valid originatingAddress, valid originalDestinationAddress, valid redirectingAddress, valid appInfo
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned

2. Method call **release()**
   Parameters: valid callSessionID reported in preamble.
   Check: no exception is returned



Test GCC_IPCALL _04

Summary: IpCall, routeReq, P_INVALID_ADDRESS

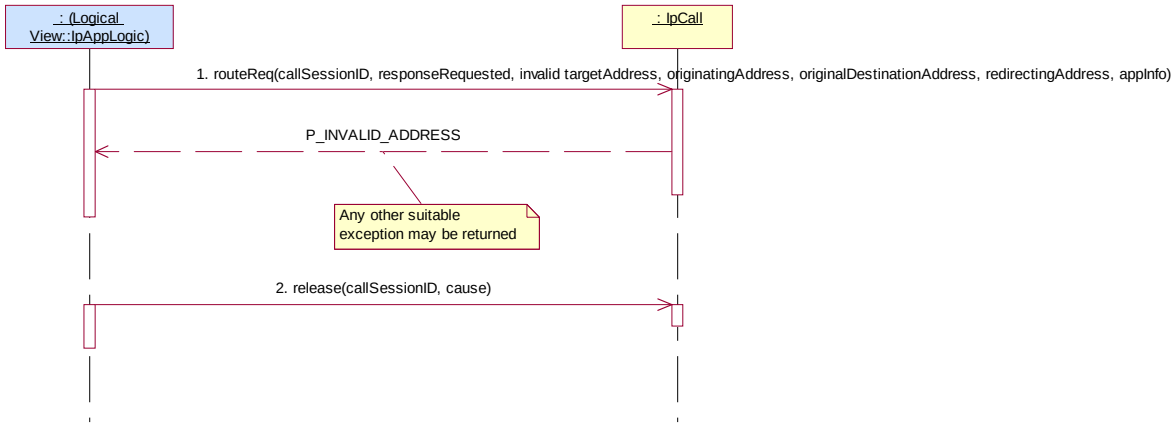
Reference: ES 201 915-4 [1], clauses 6.3.1 and 6.3.3.

Preamble: Same as GCC_IPCALL_01

Test Sequence:

1. Method call **routeReq()**
   Parameters: valid callSessionID reported in preamble, valid responseRequested, invalid targetAddress, valid originatingAddress, valid originalDestinationAddress or null value, valid redirectingAddress or null value, valid appInfo
   Check: P_INVALID_ADDRESS, or another suitable exception, is returned

2. Method call **release()**
   Parameters: valid callSessionID reported in preamble.
   Check: no exception is returned



Test GCC_IPCALL _05

Summary: IpCall, routeReq, P_INVALID_ADDRESS

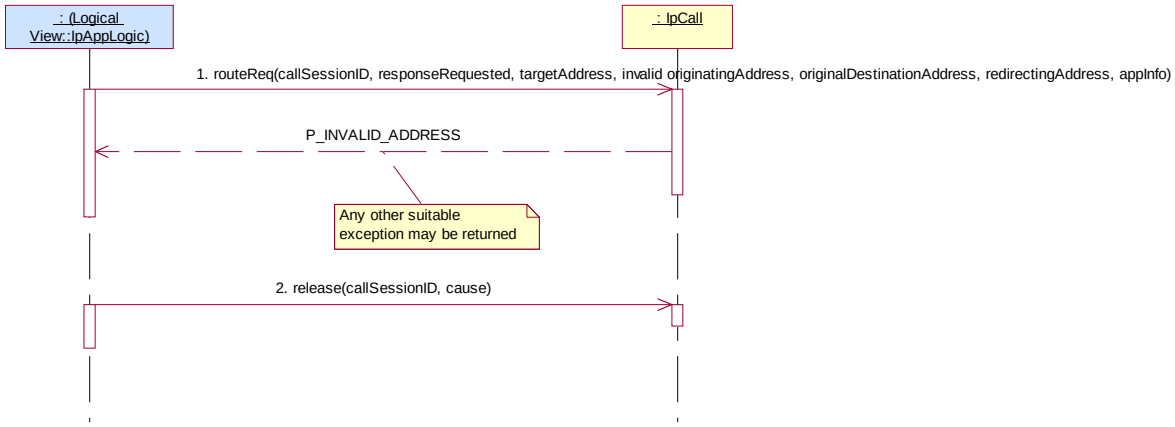
Reference: ES 201 915-4 [1], clauses 6.3.1 and 6.3.3.

Preamble: Same as GCC_IPCALL_01

Test Sequence:

1. Method call **routeReq()**
   Parameters: valid callSessionID reported in preamble, valid responseRequested, valid targetAddress, invalid originatingAddress, valid originalDestinationAddress or null value, valid redirectingAddress or null value, valid appInfo
   Check: P_INVALID_ADDRESS, or another suitable exception, is returned

2. Method call **release()**
   Parameters: valid callSessionID reported in preamble.
   Check: no exception is returned



Test GCC_IPCALL _06

Summary: IpCall, routeReq, P_INVALID_ADDRESS

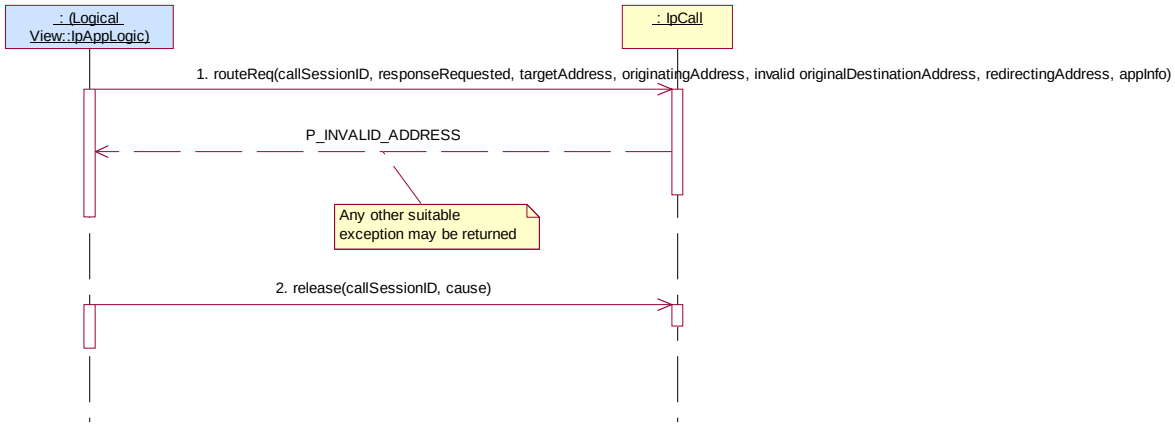
Reference: ES 201 915-4 [1], clauses 6.3.1 and 6.3.3.

Preamble: Same as GCC_IPCALL_01

Test Sequence:

1. Method call **routeReq()**
   Parameters: valid callSessionID reported in preamble, valid responseRequested, valid targetAddress, valid originatingAddress, invalid originalDestinationAddress, valid redirectingAddress, valid appInfo
   Check: P_INVALID_ADDRESS, or another suitable exception, is returned

2. Method call **release()**
   Parameters: valid callSessionID reported in preamble.
   Check: no exception is returned



Test GCC_IPCALL _07

Summary: IpCall, routeReq, P_INVALID_ADDRESS

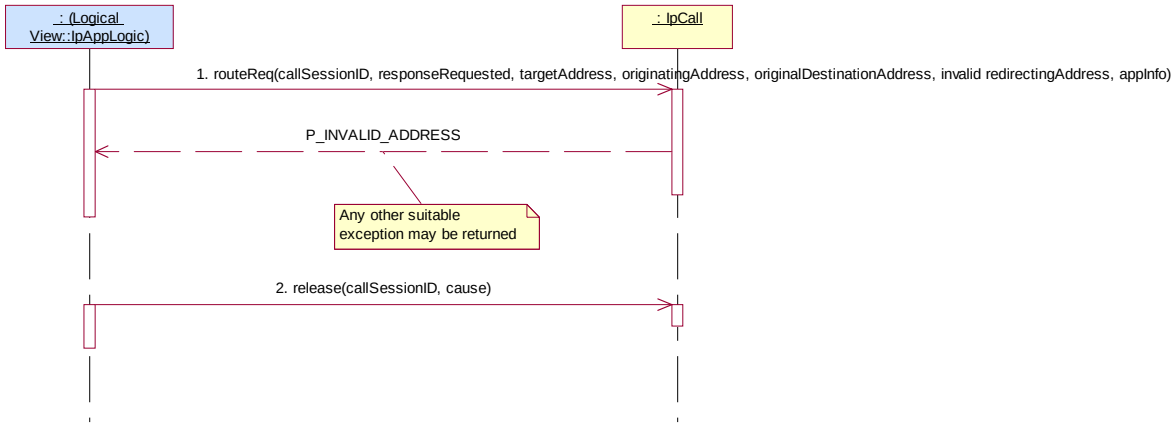
Reference: ES 201 915-4 [1], clauses 6.3.1 and 6.3.3.

Preamble: Same as GCC_IPCALL_01

Test Sequence:

1. Method call **routeReq()**
   Parameters: valid callSessionID reported in preamble, valid responseRequested, valid targetAddress, valid originatingAddress, valid originalDestinationAddress, invalid redirectingAddress, valid appInfo
   Check: P_INVALID_ADDRESS, or another suitable exception, is returned

2. Method call **release()**
   Parameters: valid callSessionID reported in preamble.
   Check: no exception is returned



Test GCC_IPCALL _08

Summary: IpCall, deassignCall, P_INVALID_SESSION_ID

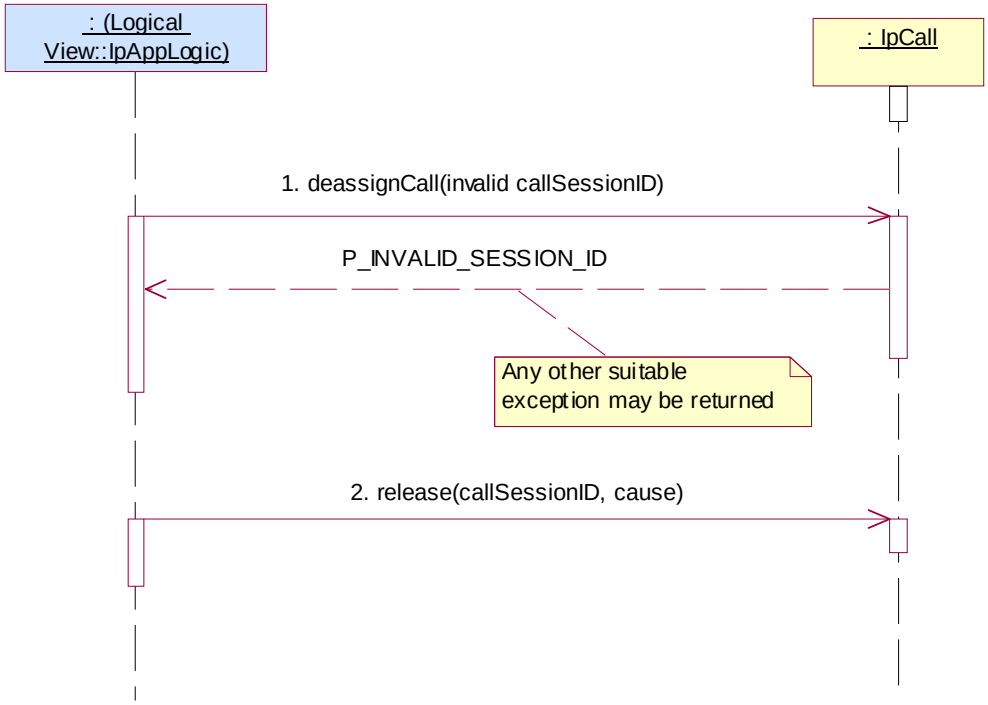
Reference: ES 201 915-4 [1], clause 6.3.3.

Preamble: Same as GCC_IPCALL_01

Test Sequence:

1. Method call **deassignCall()**
   Parameters: invalid callSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned

2. Method call **release()**
   Parameters: valid callSessionID reported in preamble.
   Check: no exception is returned



Test GCC_IPCALL _09

Summary: IpCall, release, P_INVALID_SESSION_ID

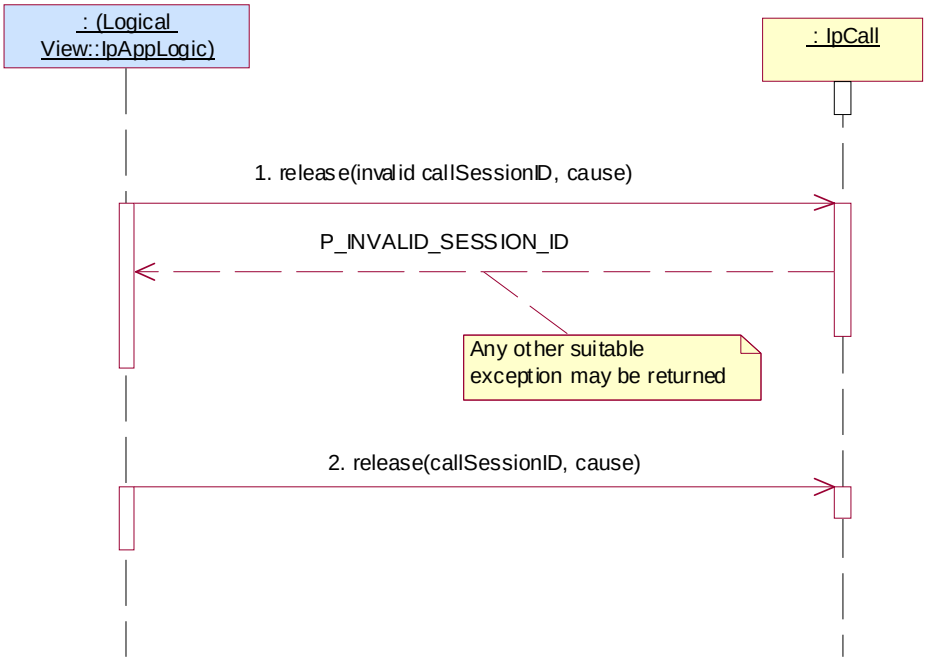
Reference: ES 201 915-4 [1], clause 6.3.3.

Preamble: Same as GCC_IPCALL_01

Test Sequence:

1. Method call **release()**
   Parameters: invalid callSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned

2. Method call **release()**
   Parameters: valid callSessionID reported in preamble.
   Check: no exception is returned



5.2.1.2.3 Optional, valid behaviour

Test GCC_IPCALL _10

Summary: IpCall, getCallInfoReq, successful

Reference: ES 201 915-4 [1], clauses 6.3.1 and 6.3.3.

Preamble: Same as GCC_IPCALL_01

Condition: getCallInfoReq method is supported.

Test Sequence:

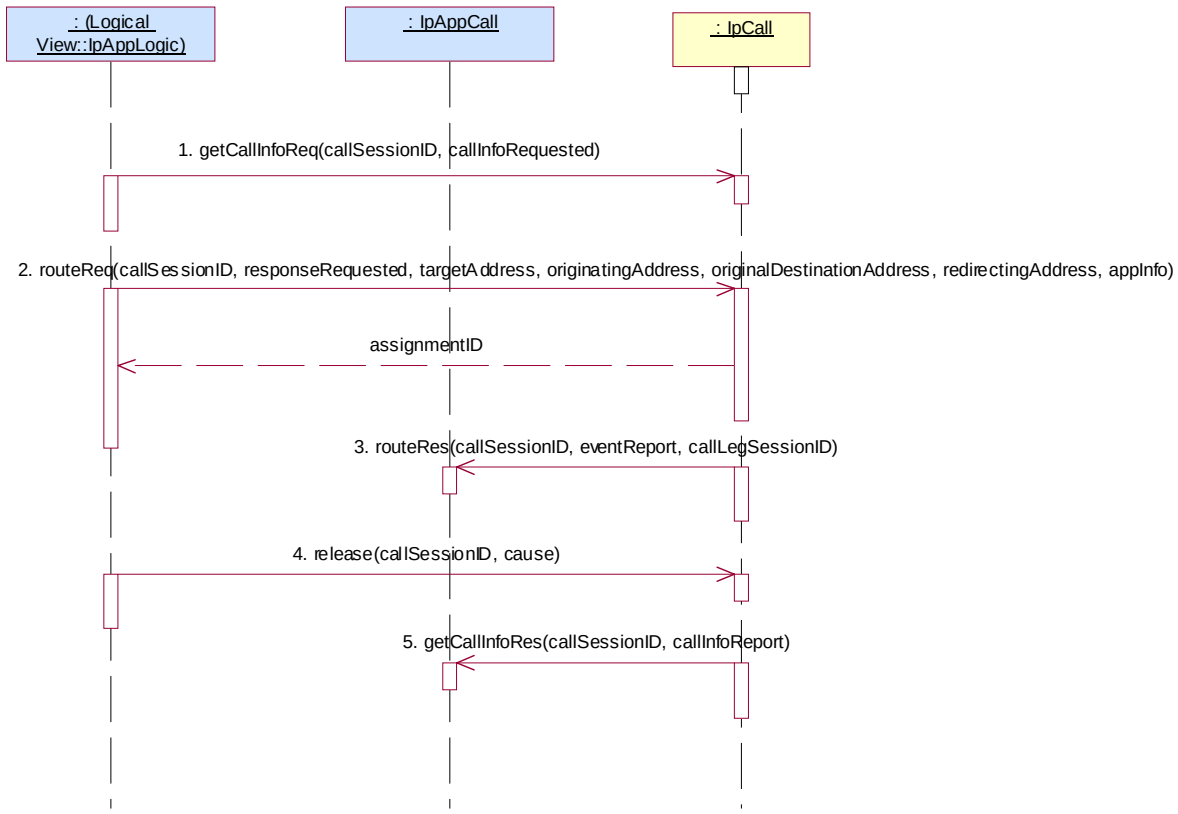
1. Method call **getCallInfoReq()**
   Parameters: valid callSessionID reported in preamble, valid callInforequested
   Check: no exception is returned

2. Method call **routeReq()**
   Parameters: valid callSessionID reported in preamble, valid responseRequested, valid targetAddress, valid originatingAddress, valid originalDestinationAddress, valid redirectingAddress, valid appInfo
   Check: Valid value of TpSessionID is returned

3. Triggered action: cause IUT to call **routeRes()** method on tester's (Application) **IpAppCall** interface.
   Parameters: callSessionID given in 1, valid eventReport, valid callLegSessionID.

4. Method call **release()**
   Parameters: valid callSessionID reported in preamble.
   Check: no exception is returned

5 Triggered action: cause IUT to call **getCallInfoRes()** method on the tester's (Application) **IpAppCall** interface.
   Parameters: callSessionID given in preamble, valid callInfoReport.



Test GCC_IPCALL _11

Summary: IpCall, setCallChargePlan, successful

Reference: ES 201 915-4 [1], clauses 6.3.1 and 6.3.3.

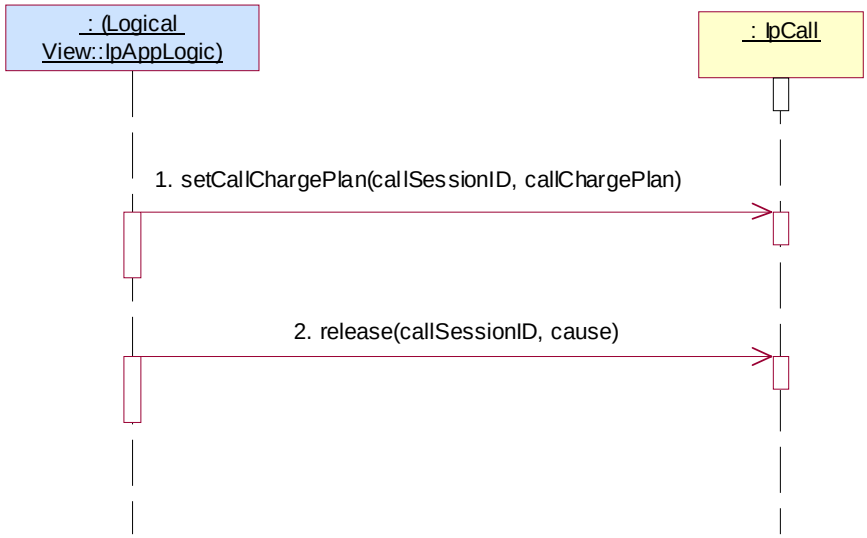
Preamble: Same as GCC_IPCALL_01

Condition: setCallChargePlan method is supported.

Test Sequence:

1. Method call **setCallChargePlan()**
   Parameters: valid callSessionID reported in preamble, valid callChargePlan
   Check: no exception is returned

2. Method call **release()**
   Parameters: valid callSessionID reported in preamble
   Check: no exception is returned



Test GCC_IPCALL _12

Summary: IpCall, setAdviceOfCharge, successful

Reference: ES 201 915-4 [1], clauses 6.3.1 and 6.3.3.

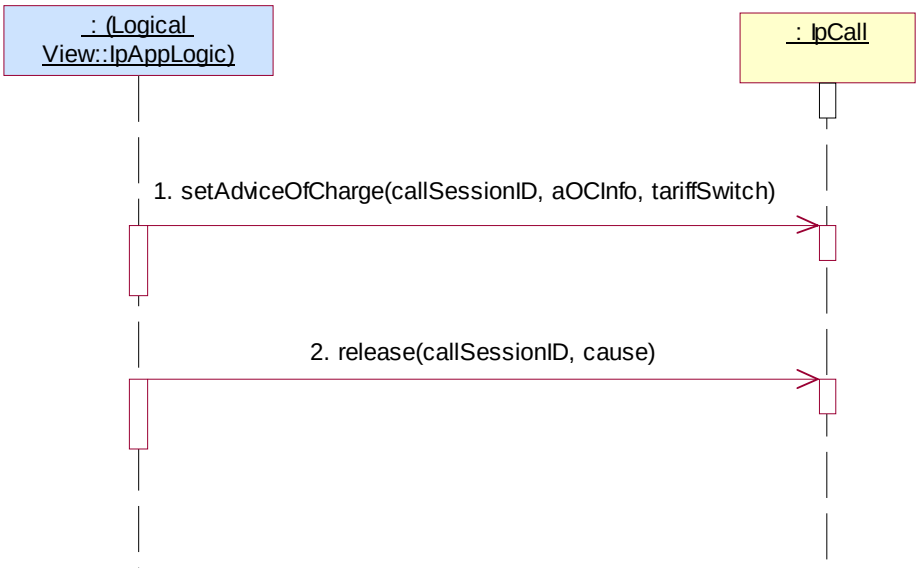
Preamble: Same as GCC_IPCALL_01

Condition: setAdviceOfCharge method is supported.

Test Sequence:

1. Method call **setAdviceOfCharge()**
   Parameters: valid callSessionID reported in preamble, valid aOCInfo, valid tariffSwitch
   Check: no exception is returned

2. Method call **release()**
   Parameters: valid callSessionID reported in preamble.
   Check: no exception is returned



Test GCC_IPCALL _13

Summary: IpCall, getMoreDialledDigitsReq, successful

Reference: ES 201 915-4 [1], clauses 6.3.1 and 6.3.3.

Condition: getMoreDialledDigitsReq method is supported.

Test Sequence:

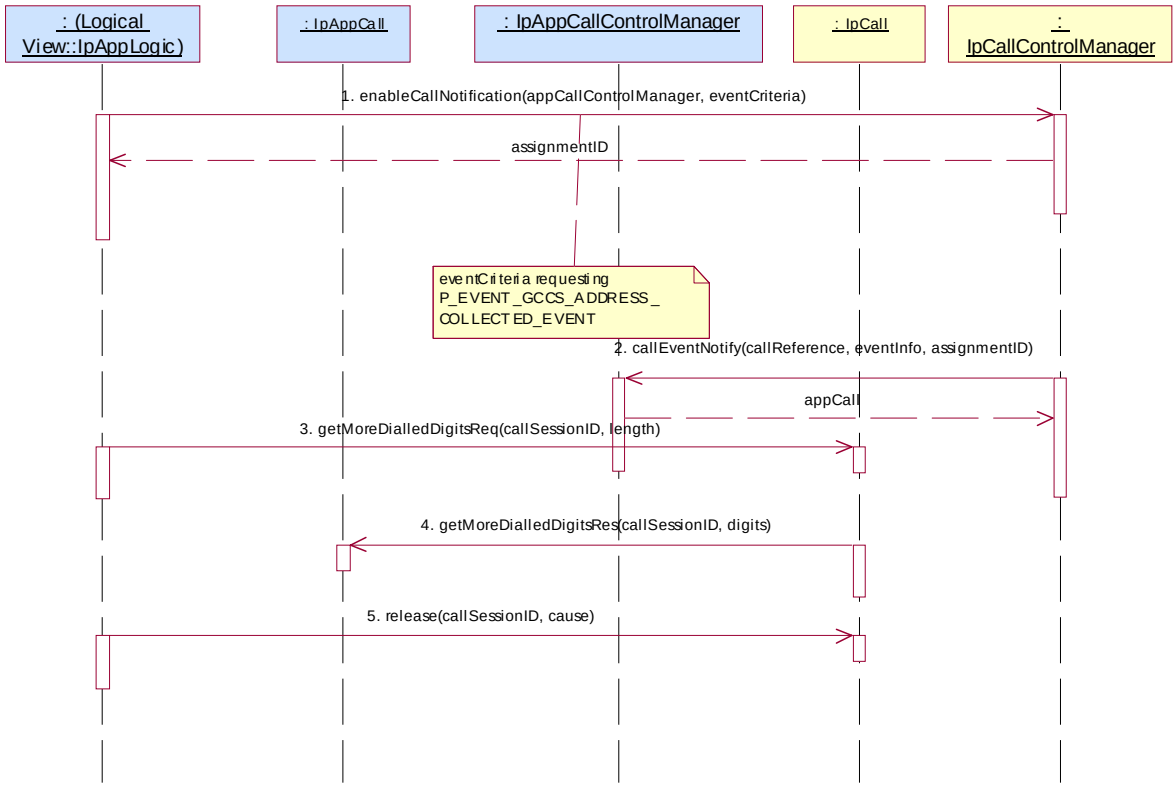
1. Method call **enableCallNotification()**
   Parameters: appCallControlManager with valid, not null, value, valid eventCriteria requesting P_EVENT_GCCS_ADDRESS_COLLECTED_EVENT.
   Check: valid value of TpAssignmentID is returned

2 Triggered action: cause IUT to call **callEventNotify()** method on the tester's (Application) **IpAppCallControlManager** interface.
   Parameters: valid callReference, valid eventInfo, assignmentID returned in 1.

3. Method call **getMoreDialledDigitsReq()**
   Parameters: valid callSessionID reported in 2., valid length
   Check: no exception is returned

4 Triggered action: cause IUT to call **getMoreDialledDigitsRes()** method on the tester's (Application) **IpAppCall** interface.
   Parameters: callSessionID given in 2., valid digits

5. Method call **release()**
   Parameters: valid callSessionID reported in2.
   Check: no exception is returned



Test GCC_IPCALL _14

Summary: IpCall, superviseCallReq, successful

Reference: ES 201 915-4 [1], clauses 6.3.1 and 6.3.3.

Preamble: Same as GCC_IPCALL_01

Condition: superviseCallReq method is supported.

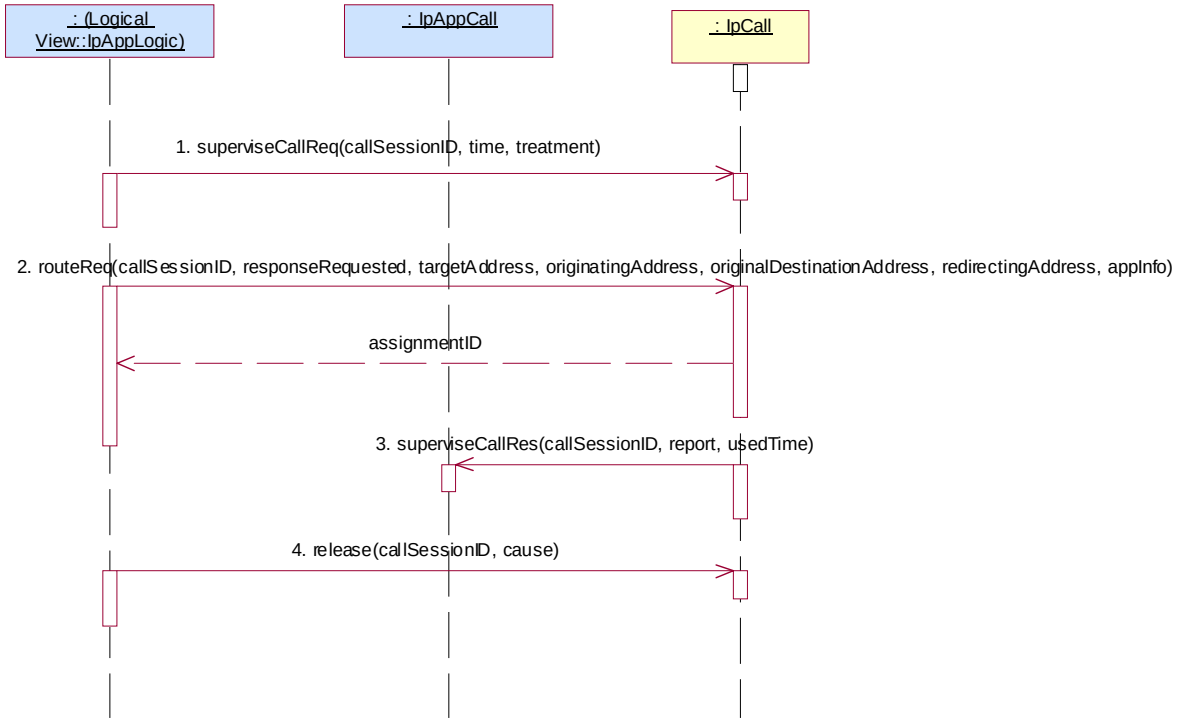
Test Sequence:

1. Method call **superviseCallReq()**
   Parameters: valid callSessionID reported in preamble, valid time, valid treatment
   Check: no exception is returned

2. Method call **routeReq()**
   Parameters: valid callSessionID reported in preamble, valid responseRequested, valid targetAddress, valid originatingAddress, valid originalDestinationAddress, valid redirectingAddress, valid appInfo
   Check: Valid value of TpSessionID is returned

3 Triggered action: cause IUT to call **superviseCallRes()** method on the tester's (Application) **IpAppCall** interface.
   Parameters: callSessionID given in 1., valid report, valid usedTime.

4. Method call **release()**
   Parameters: valid callSessionID reported in preamble.
   Check: no exception is returned



5.2.1.2.4 Optional, invalid behaviour

Test GCC_IPCALL _15

Summary: IpCall, getCallInfoReq, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 6.3.3.

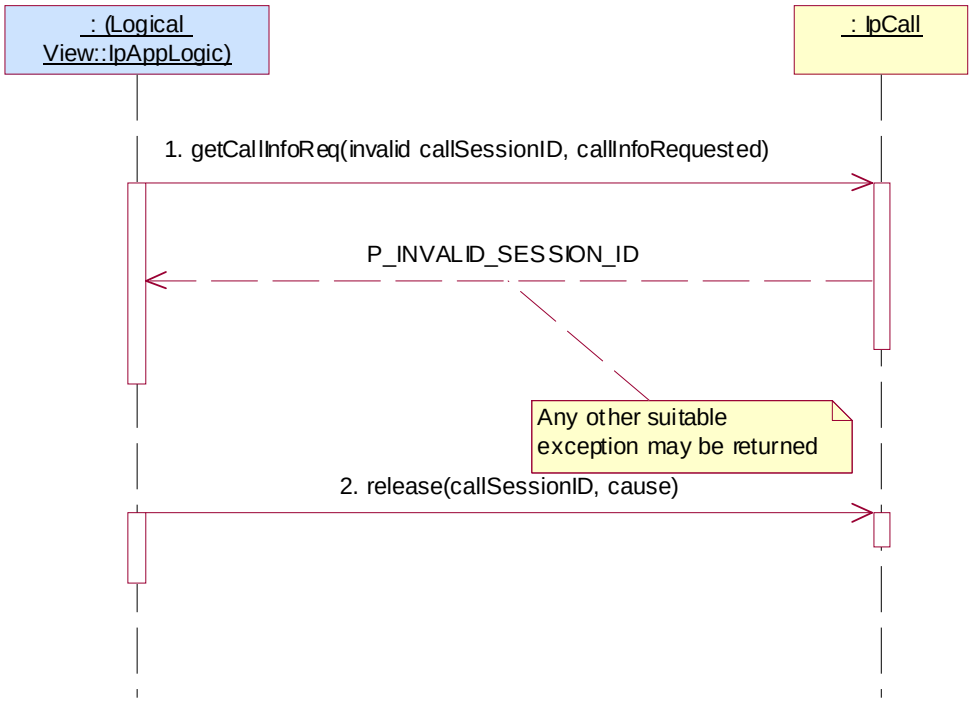
Preamble: Same as GCC_IPCALL_01

Condition: getCallInfoReq is supported.

Test Sequence:

1. Method call **getCallInfoReq()**
   Parameters: invalid callSessionID, valid callInforequested
   Check: P_INVALID_SESSION_ID, or another suitable exception, exception is returned

2. Method call **release()**
   Parameters: valid callSessionID reported in preamble.
   Check: no exception is returned



Test GCC_IPCALL _16

Summary: IpCall, setCallChargePlan, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 6.3.3.

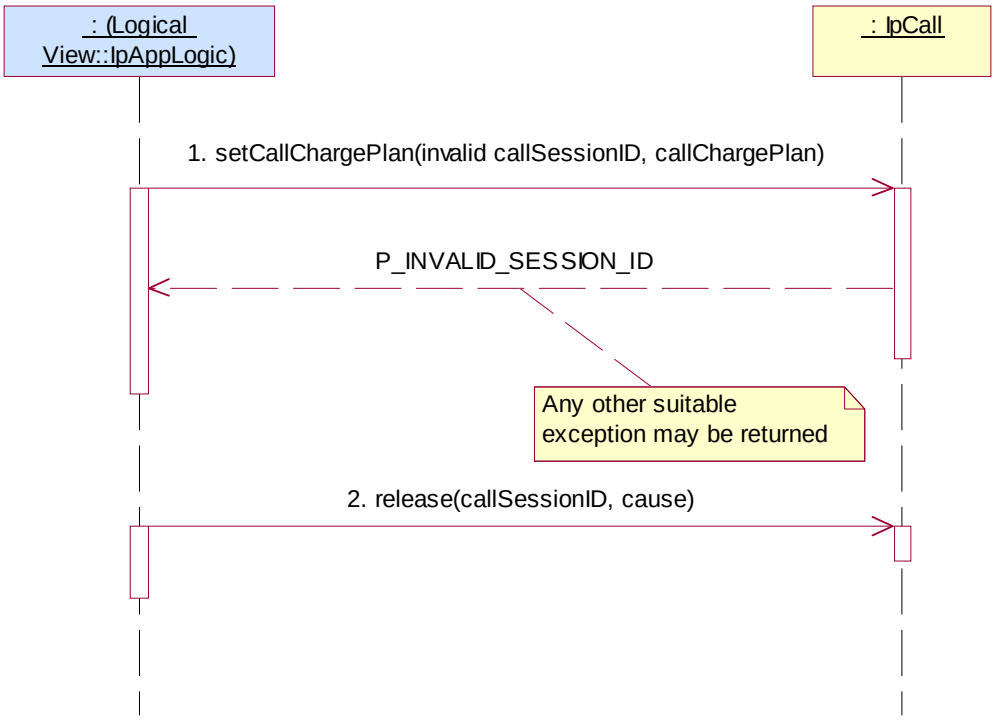
Preamble: Same as GCC_IPCALL_01

Condition: setCallChargePlan is supported.

Test Sequence:

1. Method call **setCallChargePlan()**
   Parameters: invalid callSessionID, valid callChargePlan
   Check: P_INVALID_SESSION_ID, or another suitable exception, exception is returned

2. Method call **release()**
   Parameters: valid callSessionID reported in preamble.
   Check: no exception is returned



Test GCC_IPCALL _17

Summary: IpCall, setAdviceOfCharge, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 6.3.3.

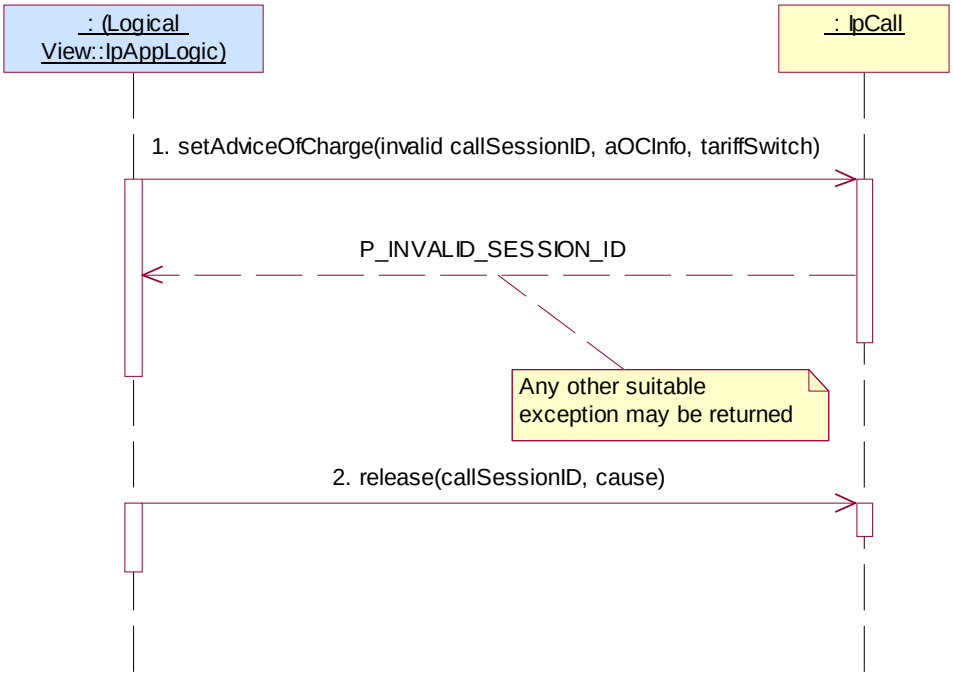
Preamble: Same as GCC_IPCALL_01

Condition: setAdviceOfCharge is supported.

Test Sequence:

1. Method call **setAdviceOfCharge()**
   Parameters: invalid callSessionID, valid aOCInfo, valid tariffSwitch
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned

2. Method call **release()**
   Parameters: valid callSessionID reported in preamble.
   Check: no exception is returned



Test GCC_IPCALL _18

Summary: IpCall, getMoreDialledDigitsReq, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 6.3.3.

Condition: getMoreDialledDigitsReq is supported.

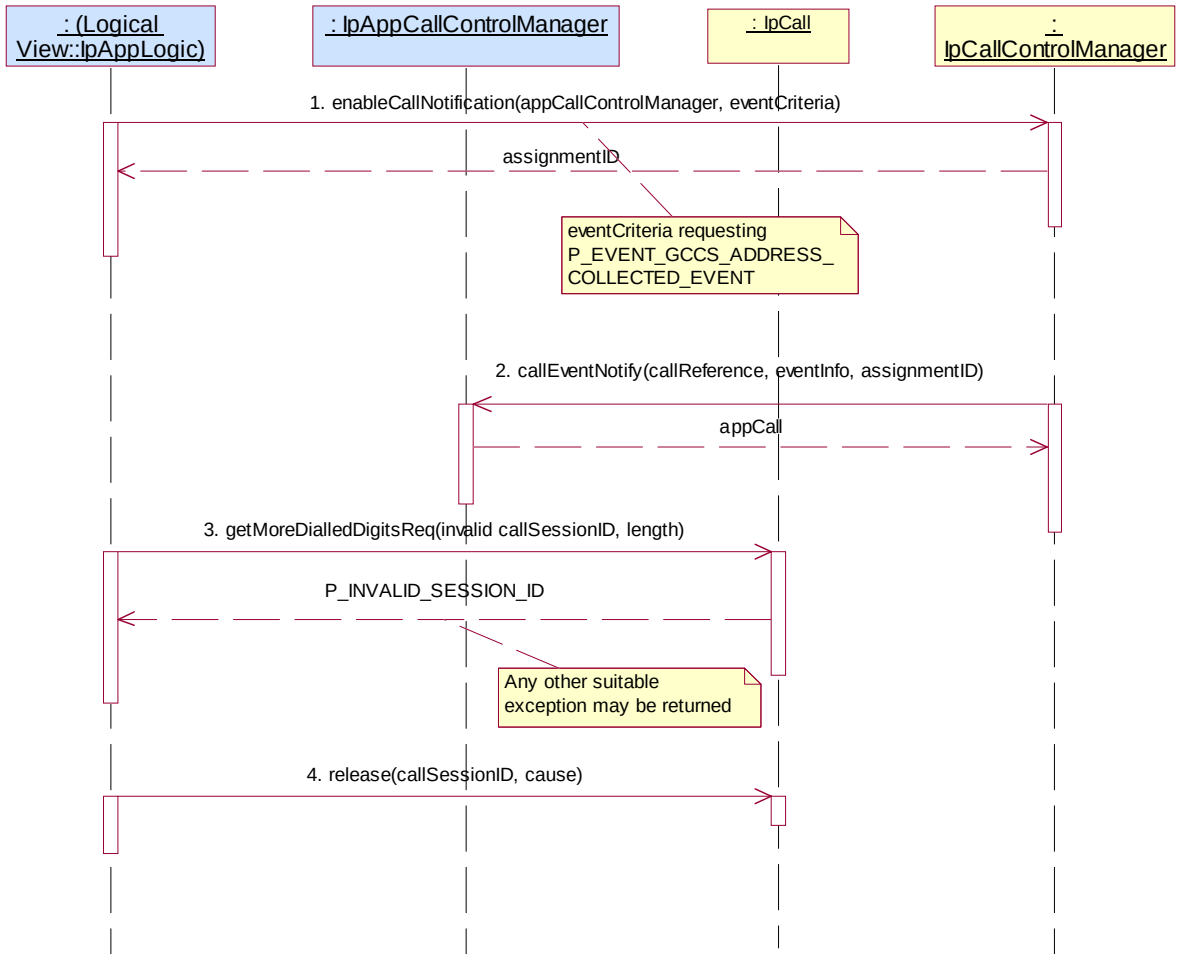
Test Sequence:

1. Method call **enableCallNotification()**
   Parameters: appCallControlManager with valid, not null, value, valid eventCriteria requesting `P_EVENT_GCCS_ADDRESS_COLLECTED_EVENT.`
   Check: valid value of TpAssignmentID is returned

2 Triggered action: cause IUT to call **callEventNotify()** method on the tester's (Application) **IpAppCallControlManager** interface.
   Parameters: valid callReference, valid eventInfo, assignmentID returned in 1.

3. Method call **getMoreDialledDigitsReq()**
   Parameters: invalid callSessionID, valid length
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned

4. Method call **release()**
   Parameters: valid callSessionID reported in 2.
   Check: no exception is returned



Test GCC_IPCALL _19

Summary: IpCall, superviseCallReq, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 6.3.3.

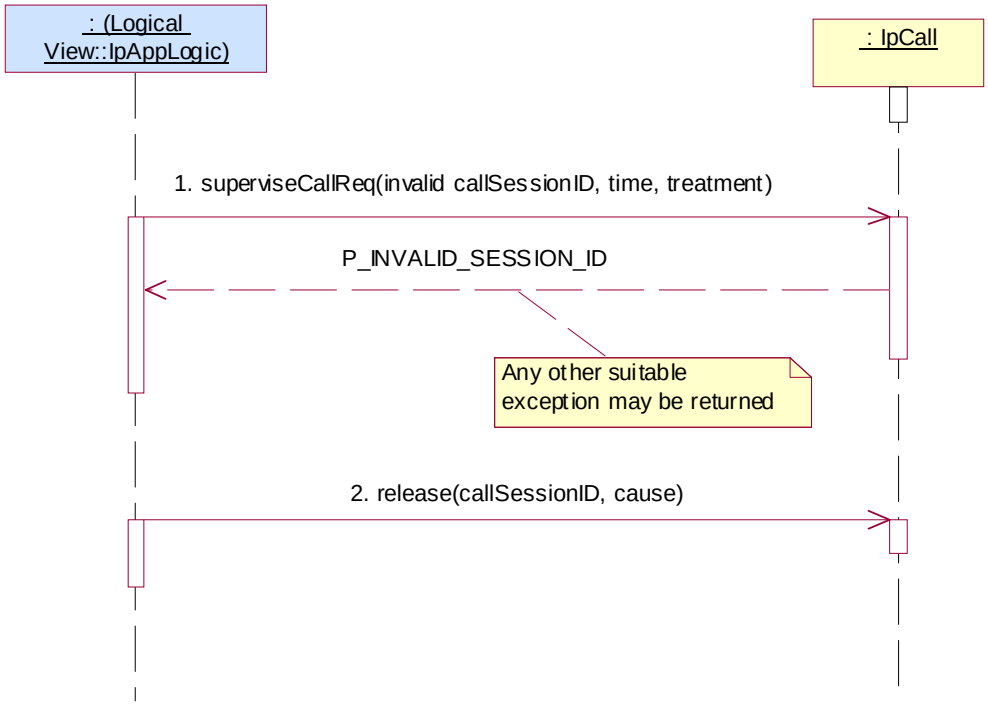
Preamble: Same as GCC_IPCALL_01

Condition: superviseCallReq is supported.

Test Sequence:

1. Method call **superviseCallReq()**
   Parameters: invalid callSessionID, valid time, valid treatment
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned

2. Method call **release()**
   Parameters: valid callSessionID reported in preamble.
   Check: no exception is returned



5.2.2 MultiParty Call Control Service (MPCC)

5.2.2.1 IpMultiPartyCallControlManager

5.2.2.1.1 Mandatory, valid behaviour

Test MPCC_ IpMultiPartyCallControlManager _01

Summary: IpMultiPartyCallControlManager, all mandatory methods, successful

Reference: ES 201 915-4 [1], clause 7.3.1

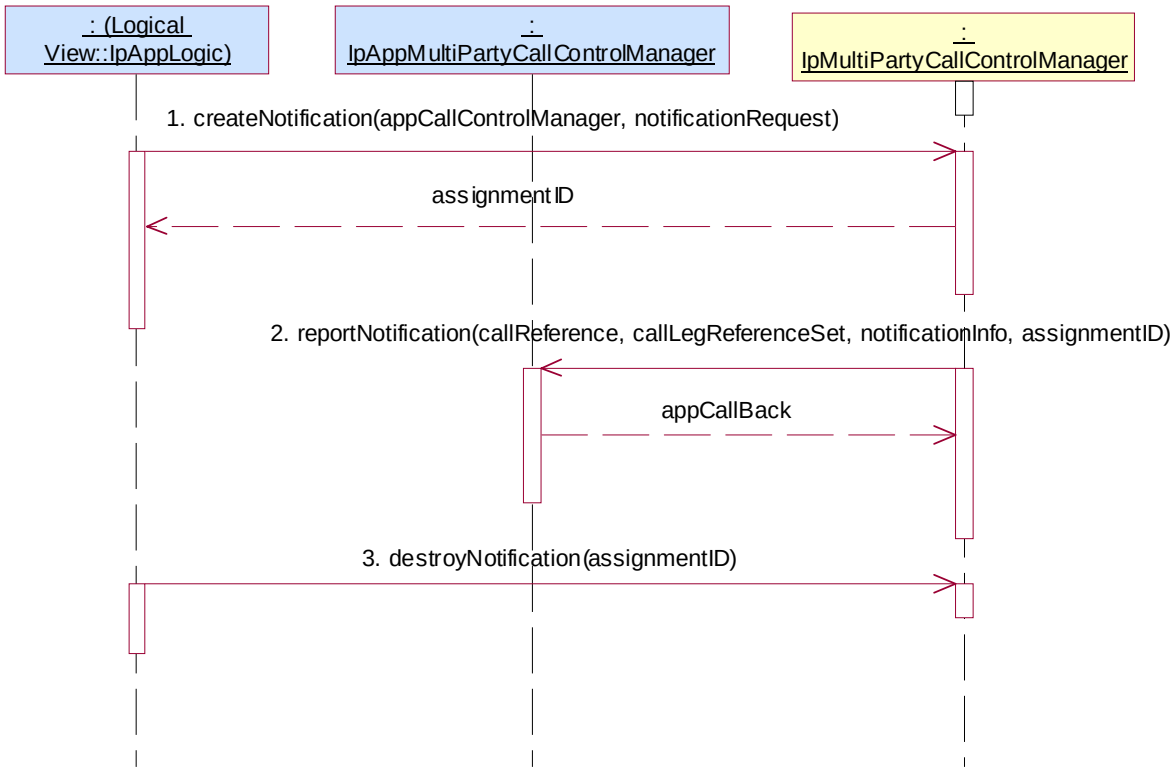
Condition: createNotification method is supported.

Test Sequence:

1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, non-null, value, valid notificationRequest
   Check: valid value of TpAssignmentID is returned

2. Triggered action: cause IUT to call Method **reportNotification()** method on the tester's (application) **IpAppMultiPartyCallControlManager** interface.
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID

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



Test MPCC_ IpMultiPartyCallControlManager _02

Summary: IpMultiPartyCallControlManager, all mandatory methods, successful

Reference: ES 201 915-4 [1], clause 7.3.1

Condition: createNotification method is supported.

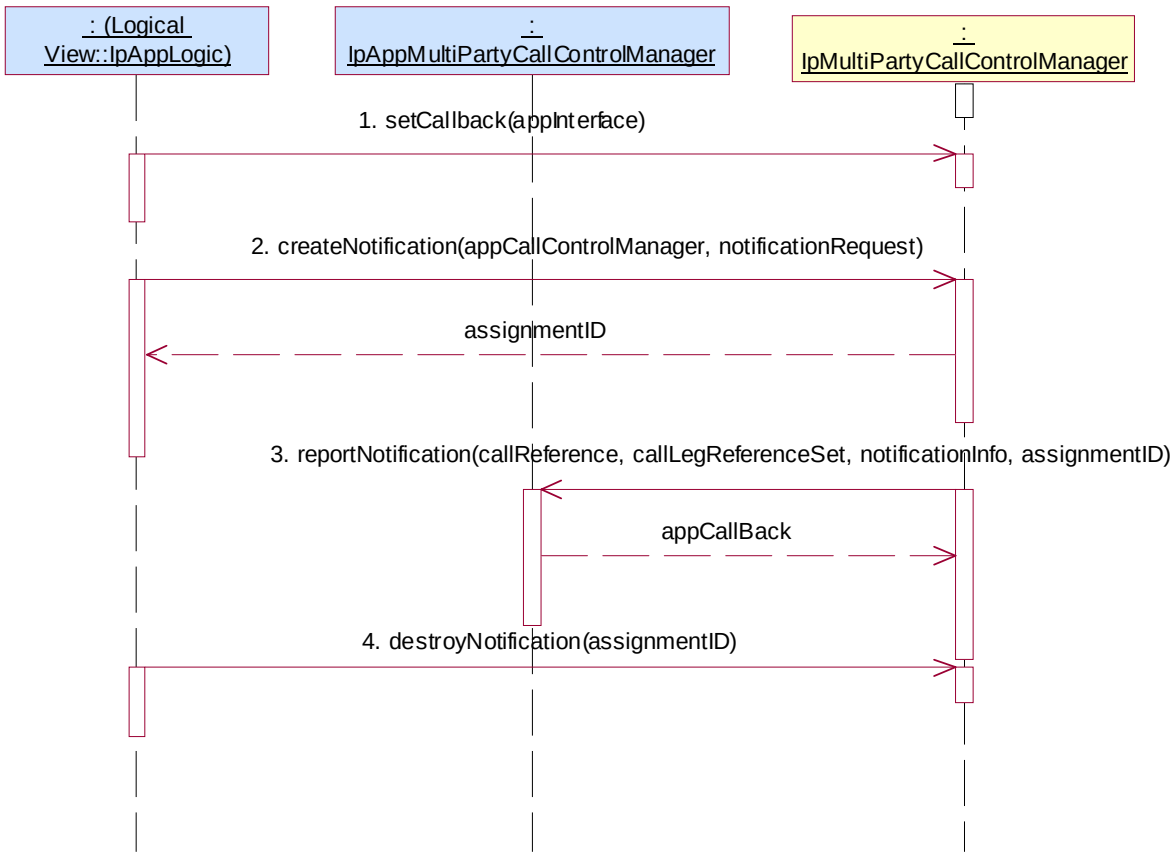
Test Sequence:

1. Method call **setCallback()** on IpCallControlManager
   Parameters: valid, non-null, value of appInterface parameter
   Check: no exception is returned

2. Method call **createNotification()**
   Parameters: appCallControlManager with null, value, valid notificationRequest
   Check: valid value of TpAssignmentID is returned

3. Triggered action: cause IUT to call Method **reportNotification()** method on the tester's (application) **IpAppMultiPartyCallControlManager** interface.
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID

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



Test MPCC_ IpMultiPartyCallControlManager _03

Summary: IpMultiPartyCallControlManager, all mandatory methods, successful

Reference: ES 201 915-4 [1], clause 7.3.1

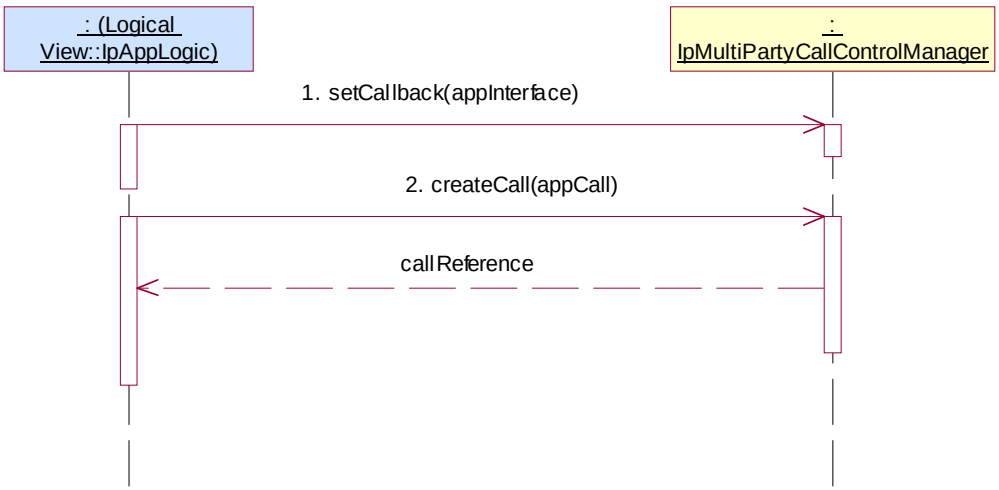
Preamble: Application has a reference interface used for callbacks.

Condition: createCall method is supported.

Test Sequence:

1. Method call **setCallback()** on IpMultiPartyCallControlManager
   Parameters: valid, non-null, value of appInterface parameter
   Check: no exception is returned

2. Method call **createCall()**
   Parameters: valid appCall
   Check: valid value of TpMultiPartyCallIdentifier is returned



5.2.2.1.2 Mandatory, invalid behaviour

Test MPCC_ IpMultiPartyCallControlManager _04

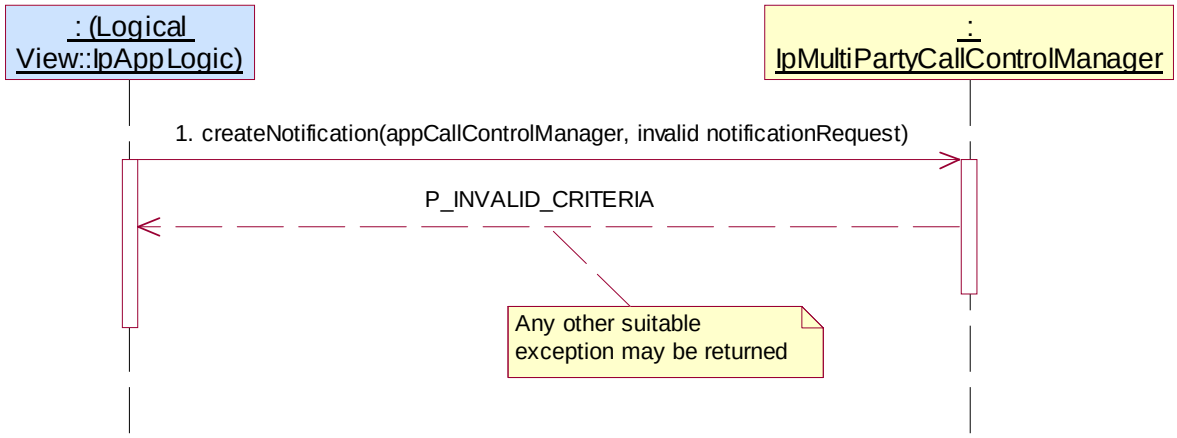
Summary: IpMultiPartyCallControlManager, createNotification, P_INVALID_CRITERIA

Reference: ES 201 915-4 [1], clause 7.3.1

Condition: createNotification method is supported.

Test Sequence:

1. Method call **createNotification()**
   Parameters: appCallControlManager with null value, invalid notificationRequest
   Check: P_INVALID_CRITERIA, or another suitable exception, is returned



Test MPCC_ IpMultiPartyCallControlManager _05

Summary: IpMultiPartyCallControlManager , destroyNotification, P_INVALID_ASSIGNMENT_ID

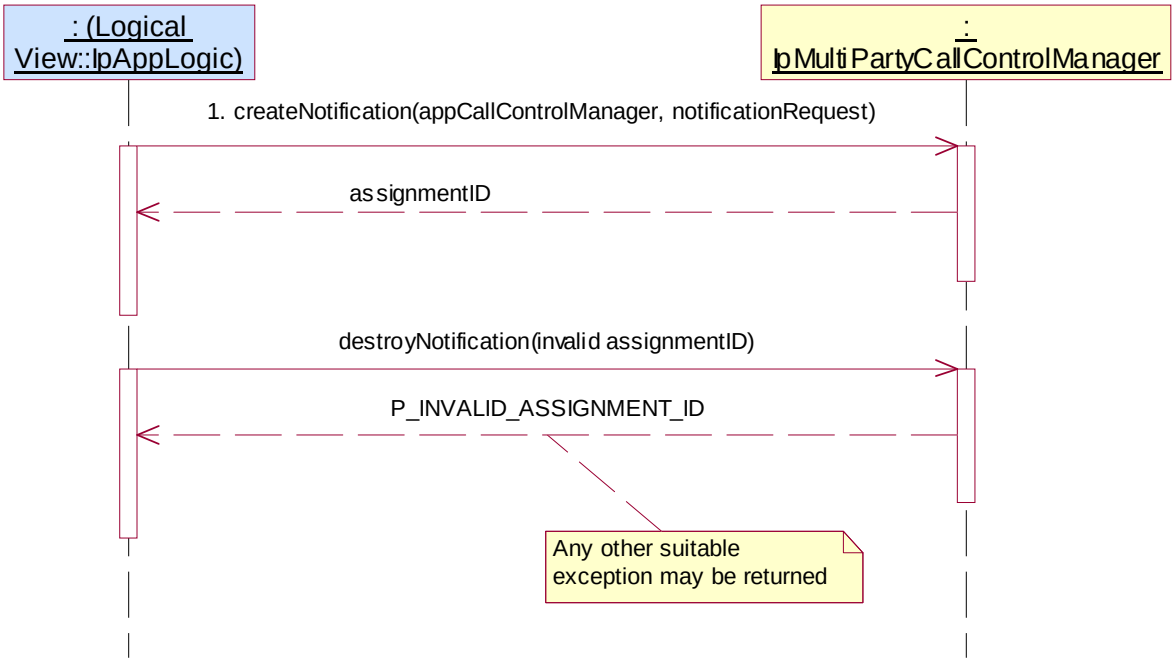
Reference: ES 201 915-4 [1], clause 7.3.1

Condition: createNotification and destroyNotification methods are supported.

Test Sequence:

1. Method call **createNotification()**
   Parameters: appCallControlManager with null value, valid notificationRequest
   Check: valid value of TpAssignmentID is returned

2. Method call **destroyNotification**()
   Parameters: INVALID assignmentID
   Check: P_INVALID_ASSIGNMENT_, or another suitable exception, is returned



Test MPCC_ IpMultiPartyCallControlManager _06

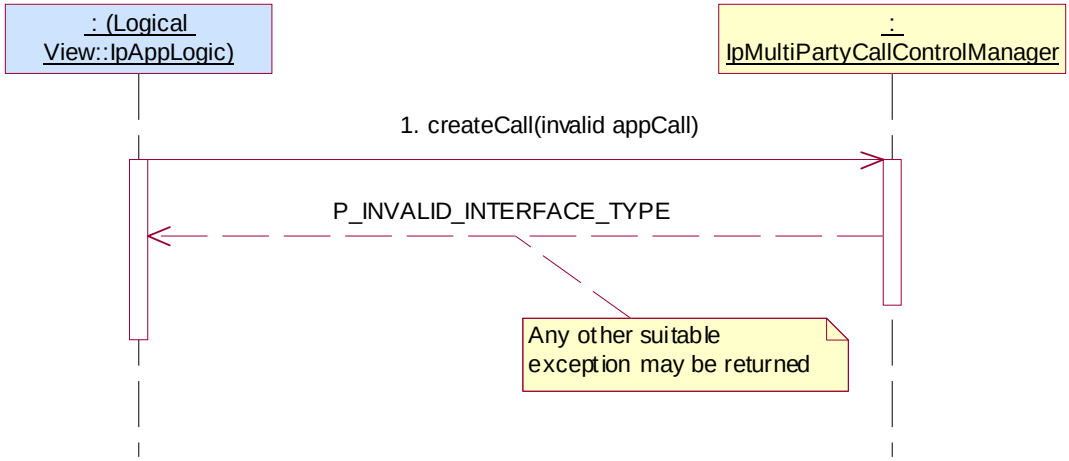
Summary: IpMultiPartyCallControlManager, createCall , P_INVALID_INTERFACE_TYPE

Reference: ES 201 915-4 [1], clause 7.3.1

Condition: createCall method is supported.

Test Sequence:

1. Method call **createCall()**
   Parameters: invalid value of appCall
   Check: P_INVALID_INTERFACE_TYPE, or another suitable exception, is returned



5.2.2.1.3 Optional, valid behaviour

Test MPCC_ IpMultiPartyCallControlManager _07

Summary: IpMultiPartyCallControlManager, getNotification, successful

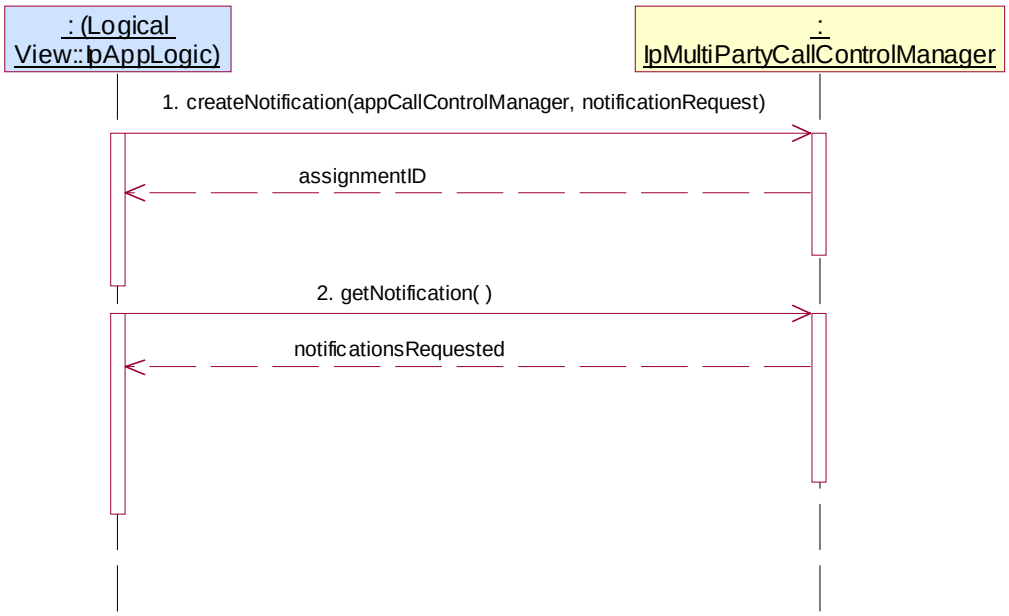
Reference: ES 201 915-4 [1], clause 7.3.1

Condition: getNotification method is supported.

Test Sequence:

1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, non-null, value, valid notificationRequest
   Check: valid value of TpAssignmentID is returned

2. Method call **getNotification()**
   Parameters: None
   Check: valid value of TpNotificationRequestedSet is returned where notificationRequest given in 1. is included as a value of this TpCallEventCriteriaResult



Test MPCC_ IpMultiPartyCallControlManager _08

Summary: IpMultiPartyCallControlManager, changeNotification, successful

Reference: ES 201 915-4 [1], clause 7.3.1

Condition: createNotification, getNotification and changeNotification methods are supported.

Test Sequence:

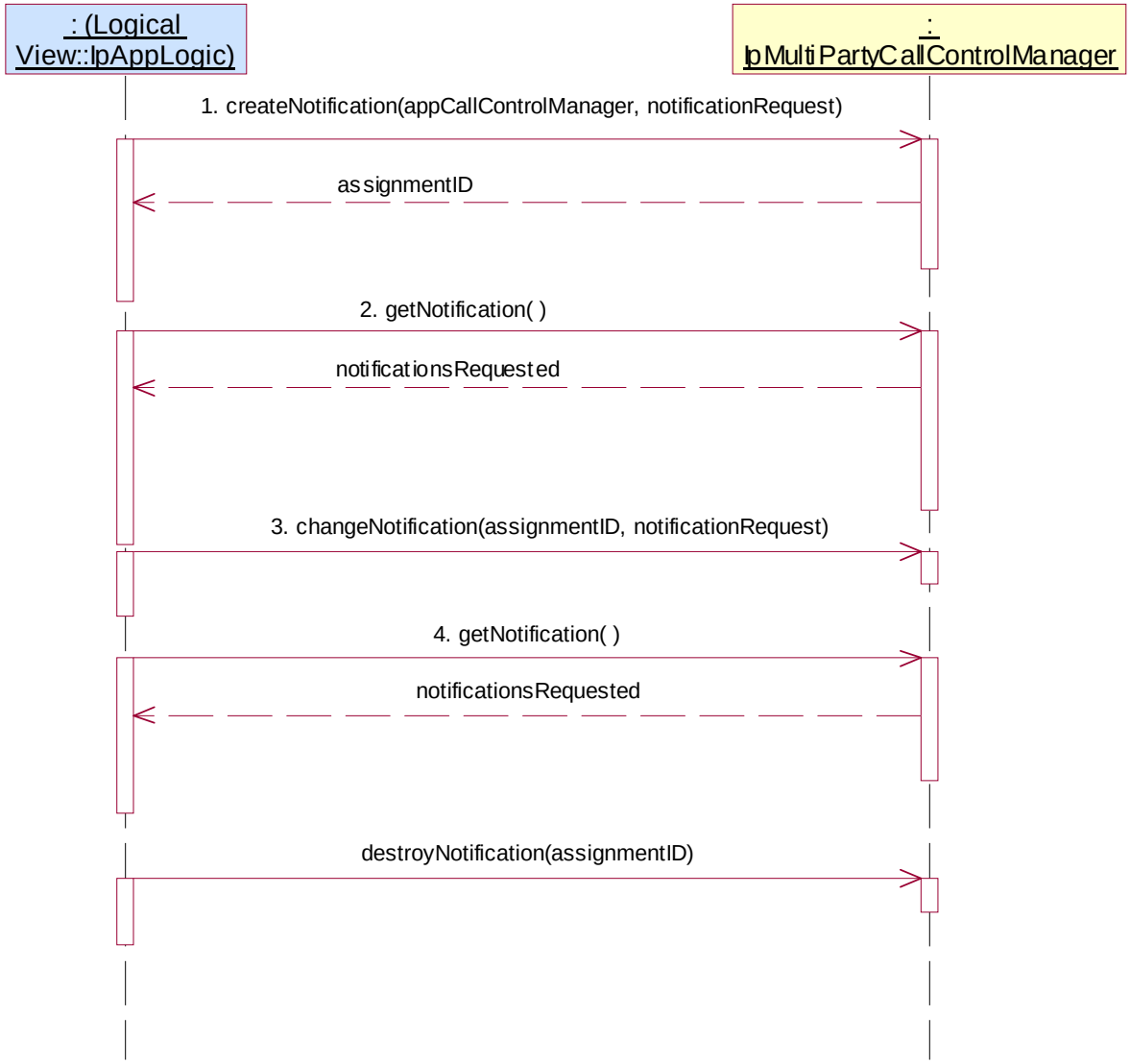
1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, non-null, value, valid notificationRequest
   Check: valid value of TpAssignmentID is returned

2. Method call **getNotification()**
   Parameters: None
   Check: valid value of TpNotificationRequestedSet is returned where notificationRequest given in 1. is included as a value of this TpCallEventCriteriaResult

3. Method call **changeNotification**()
   Parameters: assignmentID returned in 1., valid notificationRequest different from this given in 1.
   Check: no exception is returned

4. Method call **getNotification()**
   Parameters: None
   Check: valid value of TpNotificationRequestedSet is returned where notificationRequest given in 1. is included as a value of this TpCallEventCriteriaResult

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


: (Logical View::IpAppLogic)
: IpMultiPartyCallControlManager
1. createNotification(appCallControlManager, notificationRequest)
assignmentID
2. getNotification( )
notificationsRequested
3. changeNotification(assignmentID, notificationRequest)
4. getNotification( )
notificationsRequested
destroyNotification(assignmentID)

Test MPCC_ IpMultiPartyCallControlManager _09

Summary: IpMultiPartyCallControlManager, all methods, successful

Reference: ES 201 915-4 [1], clause 7.3.1

Condition: createNotification, setCallLoadControl, callOverLoadEncountered and callOverLoadCeased methods are supported.

Test Sequence:

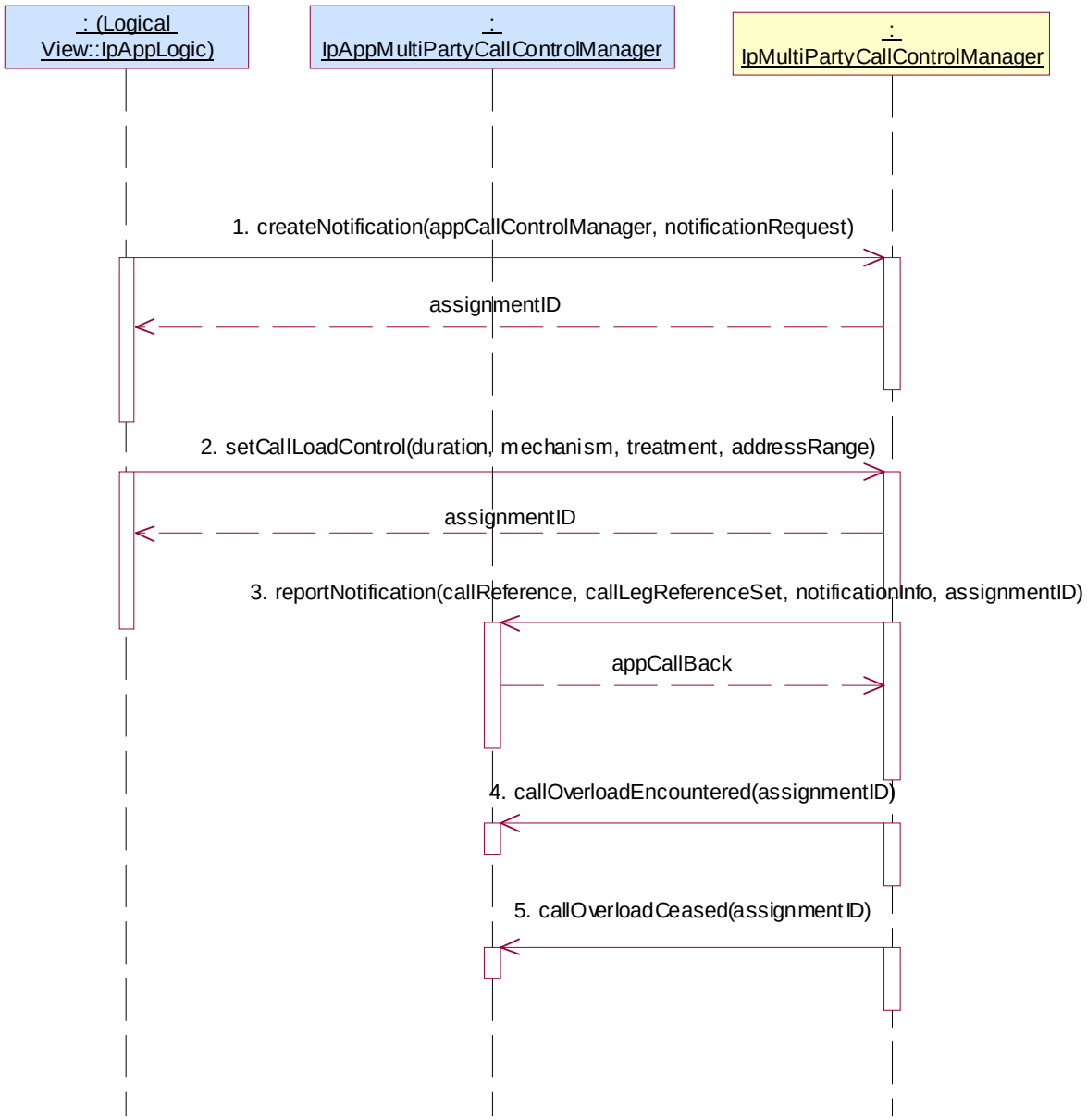
1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, non-null, value, valid notificationReques
   Check: valid value of TpAssignmentID is returned

2. Method call **setCallLoadControl()**
   Parameters: valid duration, valid mechanism, valid treatment, valid addressRange
   Check: valid value of TpAssignmentID is returned

3. Triggered action: cause IUT to call reportNotification() numerous times during the following sequence, and attempt to provoke an overload condition and then remove it.

4. Triggered action: cause IUT to call **callOverLoadEncountered()** method on the tester's (Application) **IpAppMultiPartyCallControlManager** interface.
   Parameters: valid assignmentID

5. Triggered action: cause IUT to call **callOverLoadCeased()** method on the tester's (Application) **IpAppMultiPartyCallControlManager** interface.
   Parameters: valid assignmentID


: (Logical View::IpAppLogic)
: IpAppMultiPartyCallControlManager
: IpMultiPartyCallControlManager
1. createNotification(appCallControlManager, notificationRequest)
assignmentID
2. setCallLoadControl(duration, mechanism, treatment, addressRange)
assignmentID
3. reportNotification(callReference, callLegReferenceSet, notificationInfo, assignmentID)
appCallBack
4. callOverloadEncountered(assignmentID)
5. callOverloadCeased(assignmentID)

Test MPCC_ IpMultiPartyCallControlManager _10

Summary: IpMultiPartyCallControlManager, all methods, successful

Reference: ES 201 915-4 [1], clause 7.3.1

Condition: createNotification, managerInterrupted methods are supported.

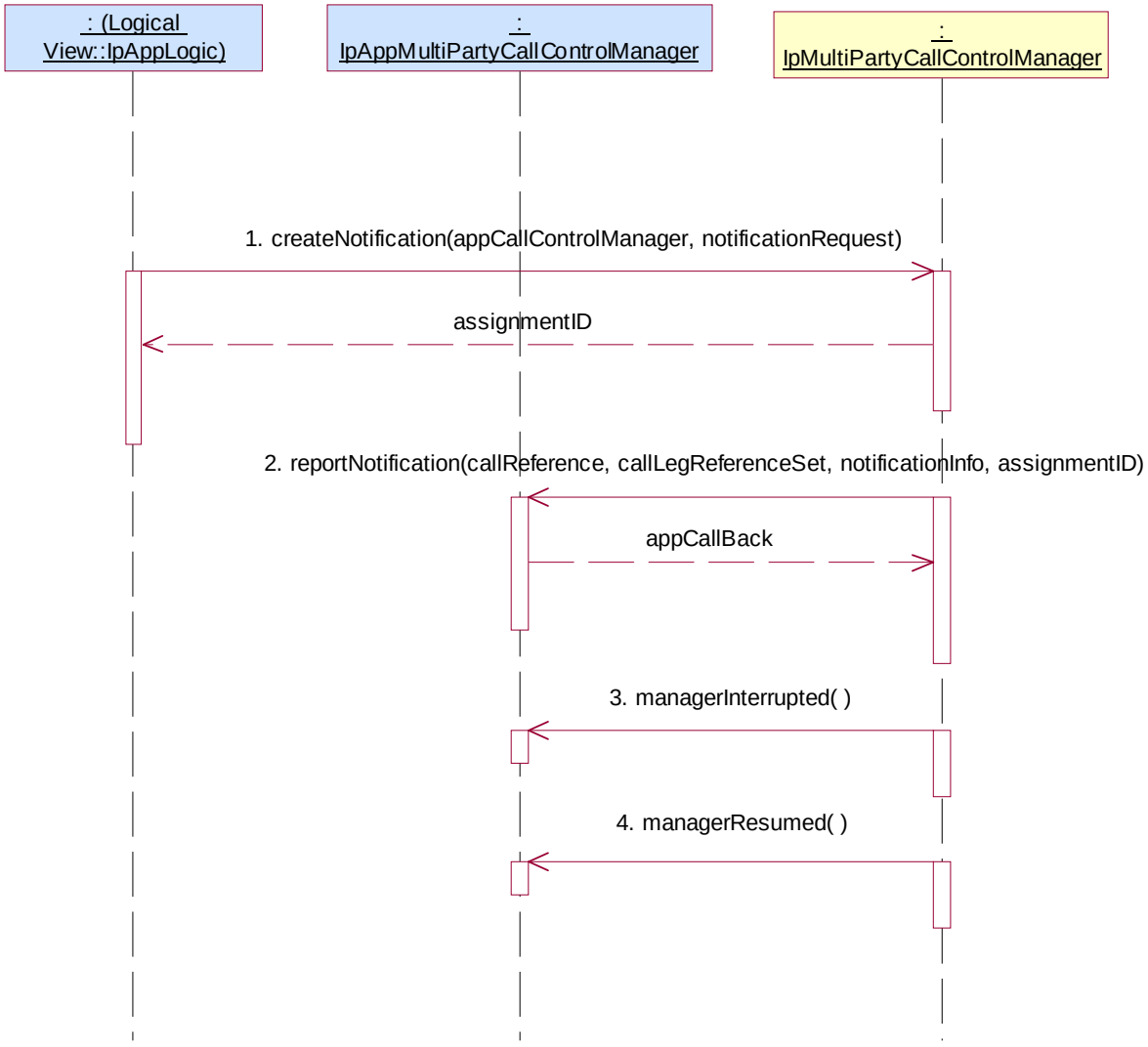
Test Sequence:

1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, not null, value, valid eventCriteria
   Check: valid value of TpAssignmentID is returned

2. Triggered action: cause IUT to call **reportNotification**()method on the tester's (Application) **IpAppMultiPartyCallControlManager** interface.

3. Triggered action: cause IUT to call **managerInterrupted()** method on the tester's (Application) **IpAppMultiPartyCallControlManager** interface.
   Parameters: None

4. Triggered action: cause IUT to call **managerResumed()** method on the tester's (Application) **IpAppMultiPartyCallControlManager** interface.
   Parameters: None



Test MPCC_ IpMultiPartyCallControlManager _11

Summary: IpMultiPartyCallControlManager, all methods, successful

Reference: ES 201 915-4 [1], clause 7.3.1

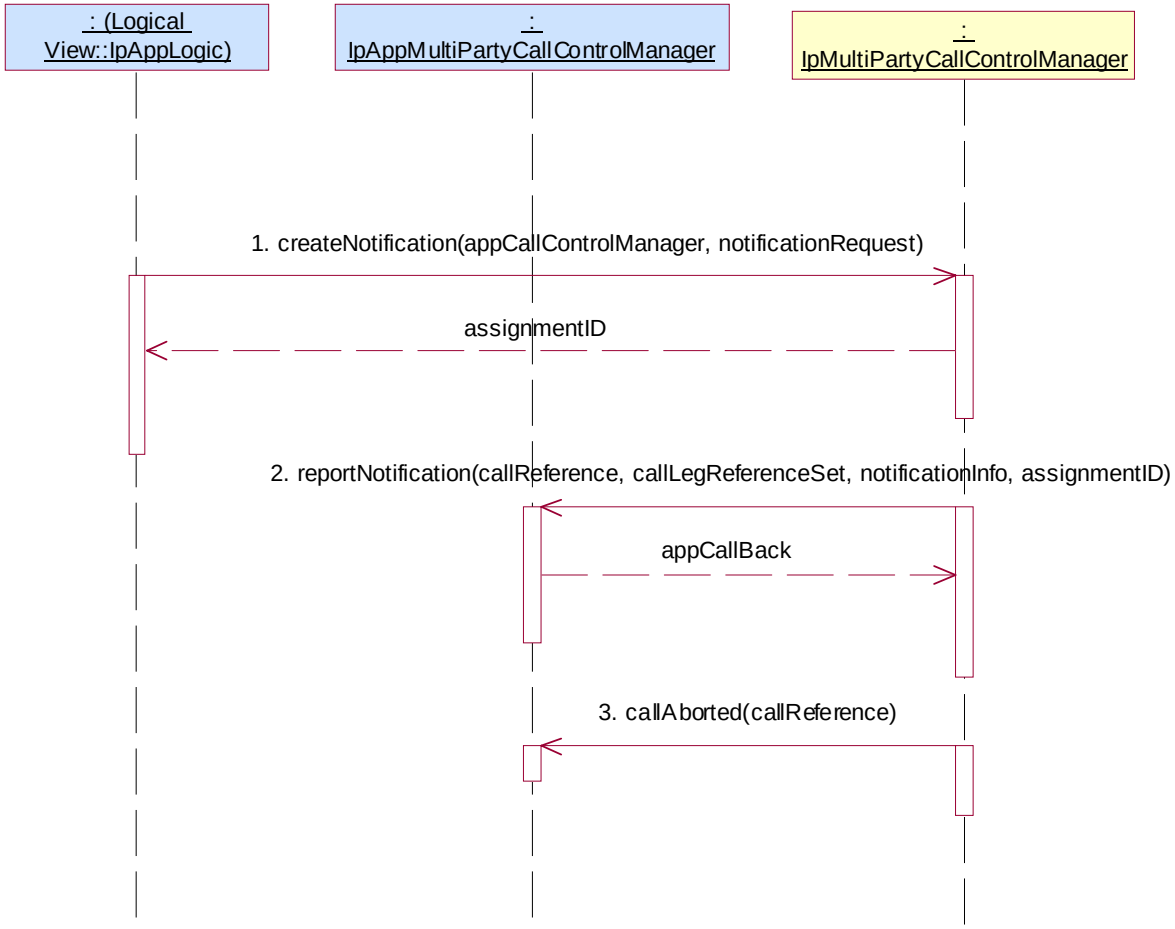
Condition: createNotification and callAborted methods are supported.

Test Sequence:

1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, not null, value, valid eventCriteria
   Check: valid value of TpAssignmentID is returned

2. Triggered action: cause IUT to call **reportNotification**() method on the tester's (Application) **IpAppMultiPartyCallControlManager** interface.

3. Triggered action: cause IUT to call **callAborted()** method on the tester's (Application) **IpAppMultiPartyCallControlManager** interface.
   Parameters: valid assignmentID as reported in reportNotification.



5.2.2.1.4 Optional, invalid behaviour

Test MPCC_ IpMultiPartyCallControlManager _12

Summary: IpMultiPartyCallControlManager , changeNotification, P_INVALID_ASSIGNMENT_ID

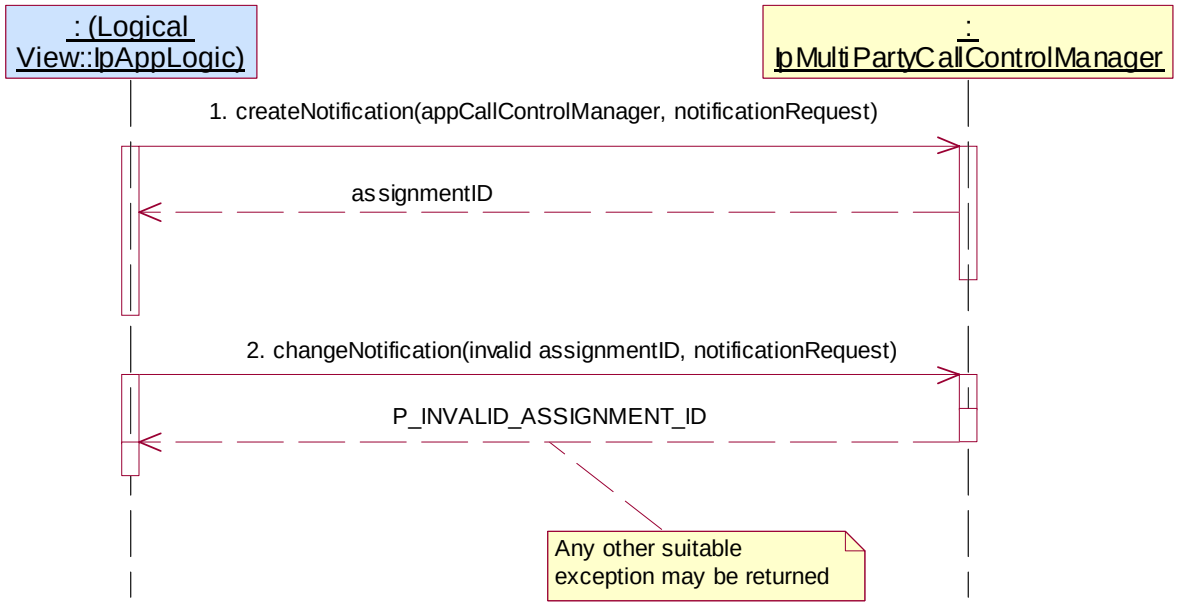
Reference: ES 201 915-4 [1], clause 7.3.1

Condition: createNotification and changeNotification methods are supported.

Test Sequence:

1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, non-null, value, valid notificationRequest
   Check: valid value of TpAssignmentID is returned

2. Method call **changeNotification**()
   Parameters: invalid assignmentID, valid notificationRequest
   Check: P_INVALID_ASSIGNMENT_ID, or another suitable exception, is returned



Test MPCC_ IpMultiPartyCallControlManager _13

Summary: IpMultiPartyCallControlManager, changeNotification, P_INVALID_CRITERIA

Reference: ES 201 915-4 [1], clause 7.3.1

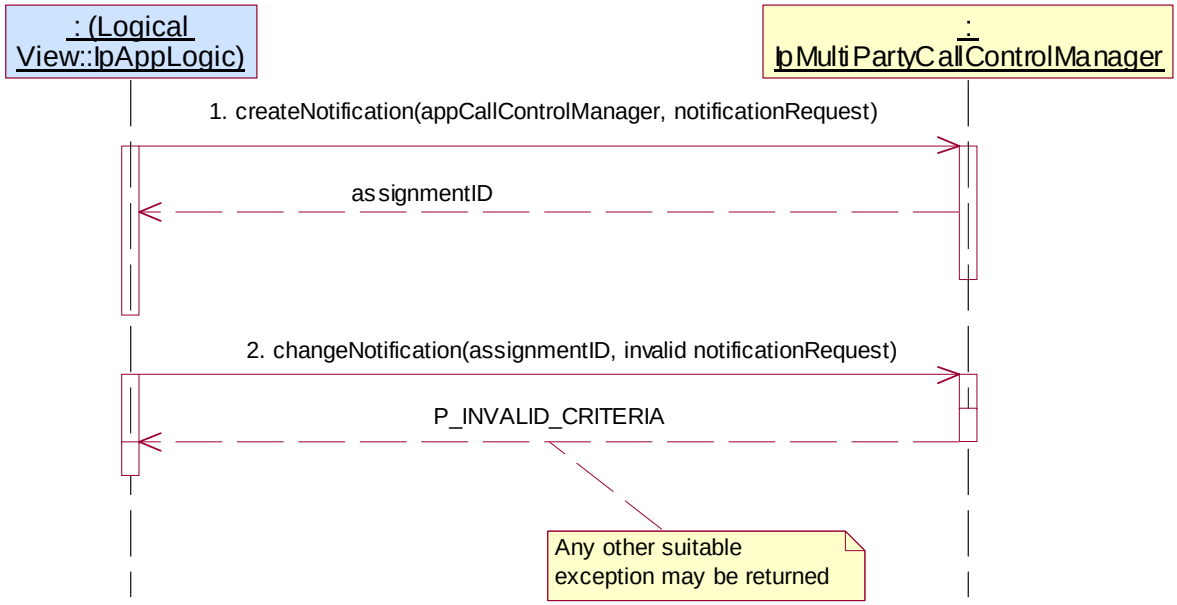
Preamble: Application has a reference interface used for callbacks.

Condition: createNotification and changeNotification methods are supported.

Test Sequence:

1. Method call **createNotification()**
   Parameters: appCallControlManager with null value, valid notificationRequest
   Check: valid value of TpAssignmentID is returned

2. Method call **changeNotification**()
   Parameters: assignmentID returned in 1., invalid notificationRequest
   Check: P_INVALID_CRITERIA, or another suitable exception, is returned



Test MPCC_ IpMultiPartyCallControlManager _14

Summary: IpMultiPartyCallControlManager, setCallLoadControl, P_INVALID_ADDRESS

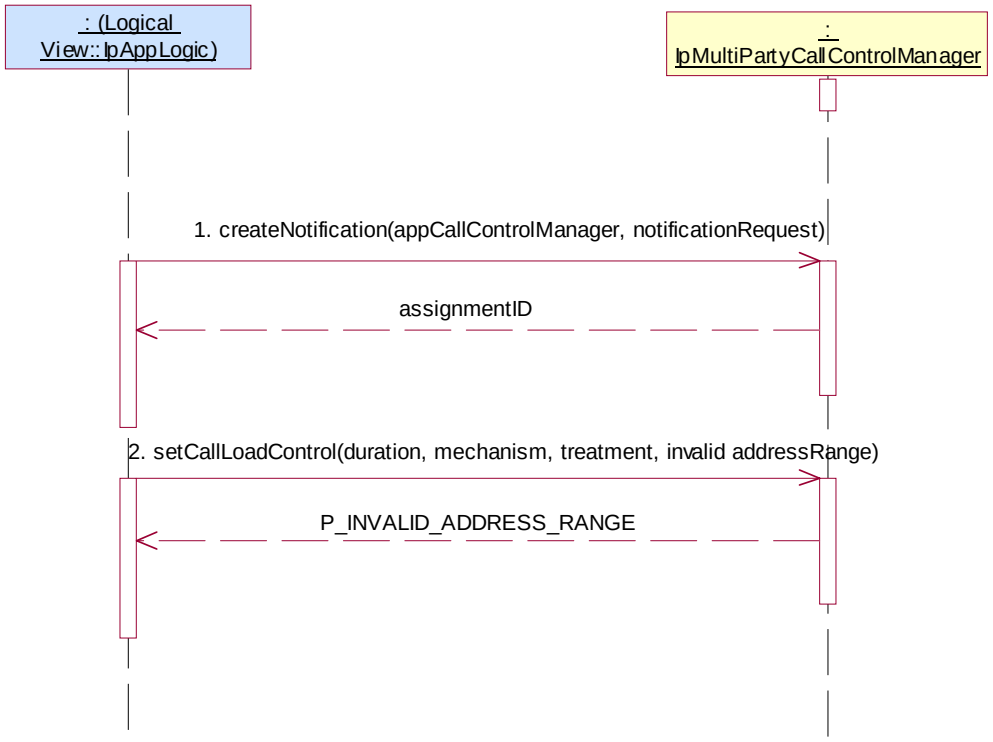
Reference: ES 201 915-4 [1], clause 7.3.1

Condition: createNotification and setCallLoadControl methods are supported.

Test Sequence:

1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, non-null, value, valid notificationRequest
   Check: valid value of TpAssignmentID is returned

2. Method call **setCallLoadControl()**
   Parameters: valid duration, valid mechanism, valid treatment, invalid addressRange
   Check: P_INVALID_ADDRESS, or another suitable exception, is returned



5.2.2.2 IpMultiPartyCall

5.2.2.2.1 Mandatory, valid behaviour

Test MPCC_ IpMultiPartyCall _01

Summary: IpMultiPartyCall, all methods mandatory, successful

Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

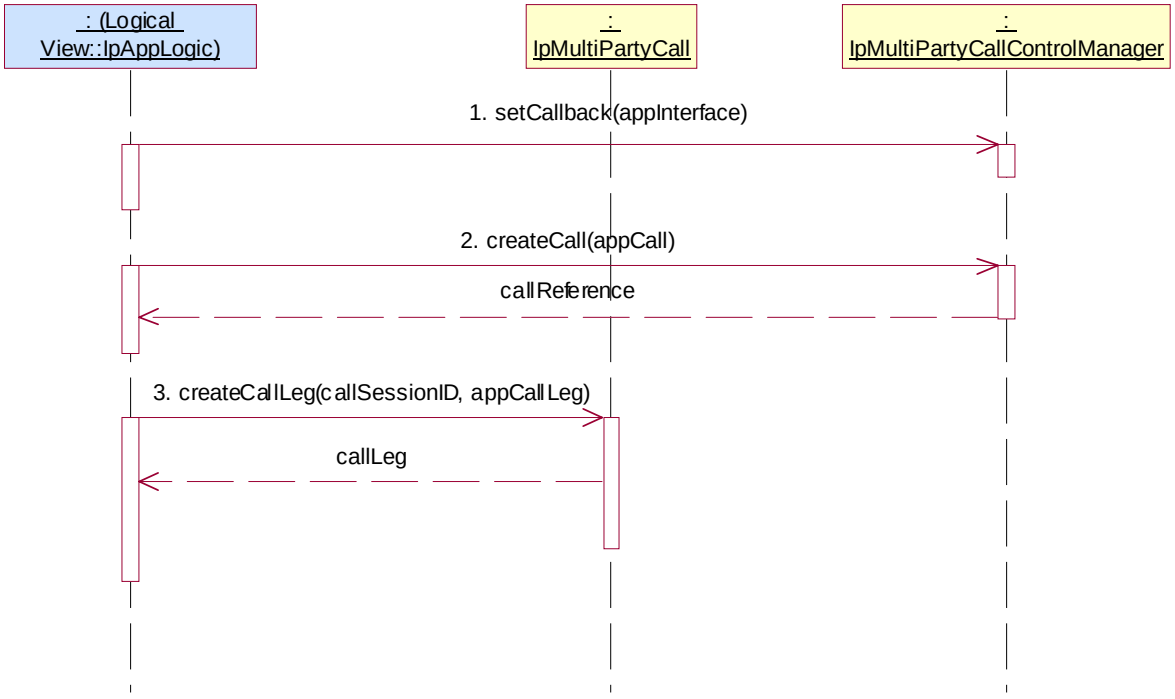
Condition: createCall and CreateCallLeg methods are supported.

Test sequence:

1. Method call **setCallback()** on IpMultiPartyCallControlManager
   Parameters: valid, non-null, value of appInterface parameter
   Check: no exception is returned

2. Method call **createCall()**
   Parameters: valid appCall
   Check: valid value of TpMultiPartyCallIdentifier is returned

3. Method call **createCallLeg()**
   Parameters: valid callSessionID returned in 1., valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned



Test MPCC_ IpMultiPartyCall _02

Summary: IpMultiPartyCall, all methods mandatory, successful

Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

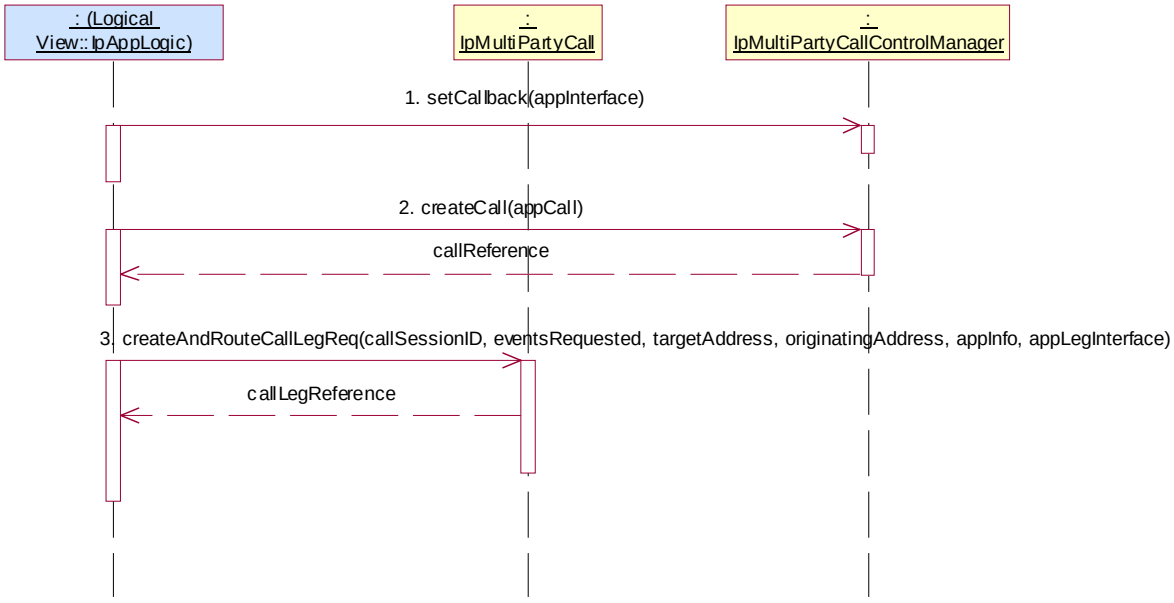
Condition: createCall and CreateAndRouteCallLeg methods are supported.

Test sequence:

1. Method call **setCallback()** on IpMultiPartyCallControlManager
   Parameters: valid, non-null, value of appInterface parameter
   Check: no exception is returned

2. Method call **createCall()**
   Parameters: valid appCall
   Check: valid value of TpMultiPartyCallIdentifier is returned

3. Method call **createAndRouteCallLegReq()**
   Parameters: valid callSessionID returned in 2., valid eventsRequested, valid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: valid value of TpCallLegIdentifier



Test MPCC_ IpMultiPartyCall _03

Summary: IpMultiPartyCall, all methods mandatory, successful

Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Application has a valid callSessionID returned by one of the two following sequence:

1. Method call **setCallback()** on IpMultiPartyCallControlManager
   Parameters: valid, non-null, value of appInterface parameter
   Check: no exception is returned

2. Method call **createCall()**
   Parameters: valid appCall
   Check: valid value of TpMultiPartyCallIdentifier is returned

either

3. Method call **createCallLeg()**
   Parameters: valid callSessionID returned in 2., valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

4. Method call **routeReq()**
   Parameters: valid callLegSessionID returned in 3, valid targetAddress, valid originatingAddress, , valid ppInfo, valid connectionProperties
   Check: no exception is returned

or

3. Method call **createAndRouteCallLegReq()**
   Parameters: valid callSessionID returned in 2., valid eventsRequested, valid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: valid value of TpCallLegIdentifier

or

1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, non-null, value, valid notificationRequest
   Check: valid value of TpAssignmentID is returned

2. Triggered action: cause IUT to call Method **reportNotification()** method on the tester's (application) **IpAppMultiPartyCallControlManager** interface.
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID

either

3. Method call **createCallLeg()**
   Parameters: valid callSessionID reported in 2., valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

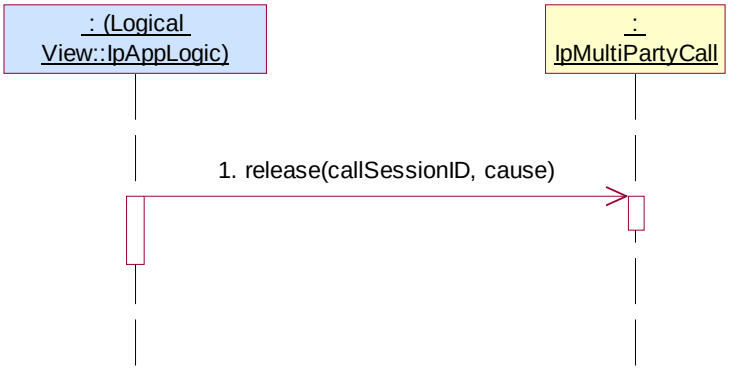
4. Method call **routeReq()**
   Parameters: valid callLegSessionID returned in 3, valid targetAddress, valid originatingAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

or

3. Method call **createAndRouteCallLegReq()**
   Parameters: valid callSessionID reported in 2., valid eventsRequested, valid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: valid value of TpCallLegIdentifier

Test Sequence:

1. Method call **release()**
   Parameters: valid callSessionID reported inpreamble, valid cause
   Check: no exception is returned



Test MPCC_ IpMultiPartyCall _04

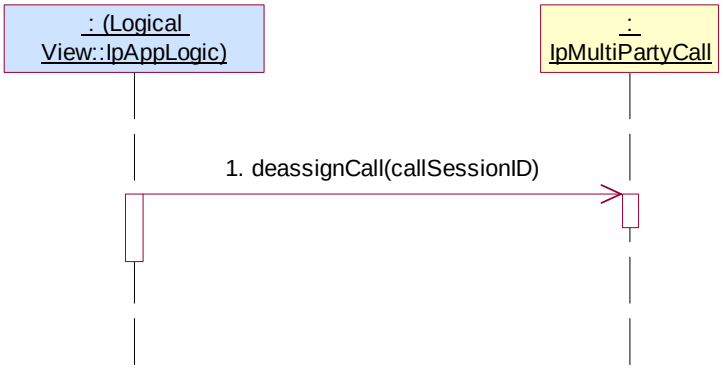
Summary: IpMultiPartyCall, all methods mandatory, successful

Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as MPCC_ IpMultiPartyCall _03

Test Sequence:

1. Method call **deassignCall()**
   Parameters: valid callSessionID reported in preamble.
   Check: no exception is returned



5.2.2.2.2 Mandatory, invalid behaviour

Test MPCC_ IpMultiPartyCall _05

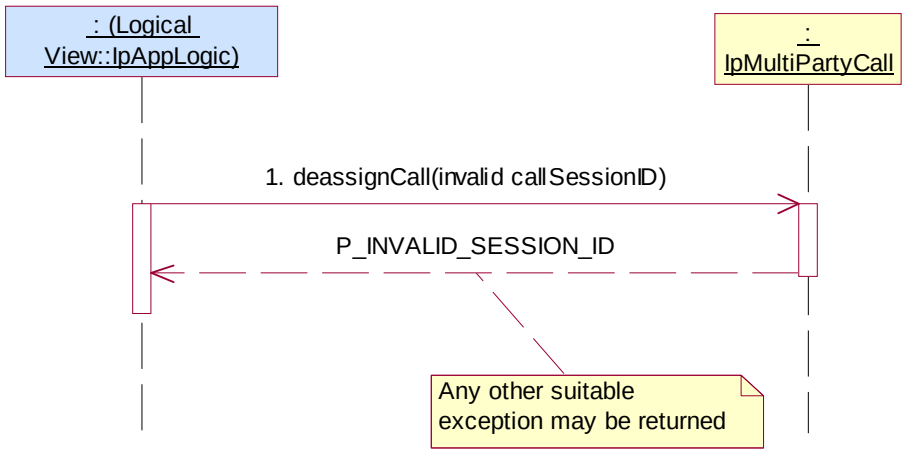
Summary: IpMultiPartyCall, deassignCall, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Same as MPCC_ IpMultiPartyCall _03

Test Sequence:

1. Method call **deassignCall()**
   Parameters: invalid callSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MPCC_ IpMultiPartyCall _06

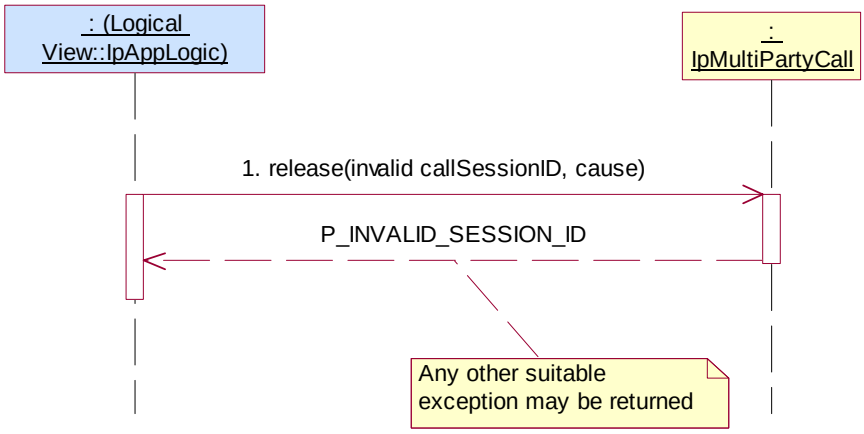
Summary: IpMultiPartyCall, release, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Same as MPCC_ IpMultiPartyCall _03

Test Sequence:

1. Method call **release()**
   Parameters: invalid callSessionID, valid cause
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MPCC_ IpMultiPartyCall _07

Summary: IpMultiPartyCall, createCallLeg, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Application has a valid callSessionID returned by one of the two following sequence:

1. Method call **setCallback()** on IpMultiPartyCallControlManager
   Parameters: valid, non-null, value of appInterface parameter
   Check: no exception is returned

2. Method call **createCall()**
   Parameters: valid appCall
   Check: valid value of TpMultiPartyCallIdentifier is returned

or

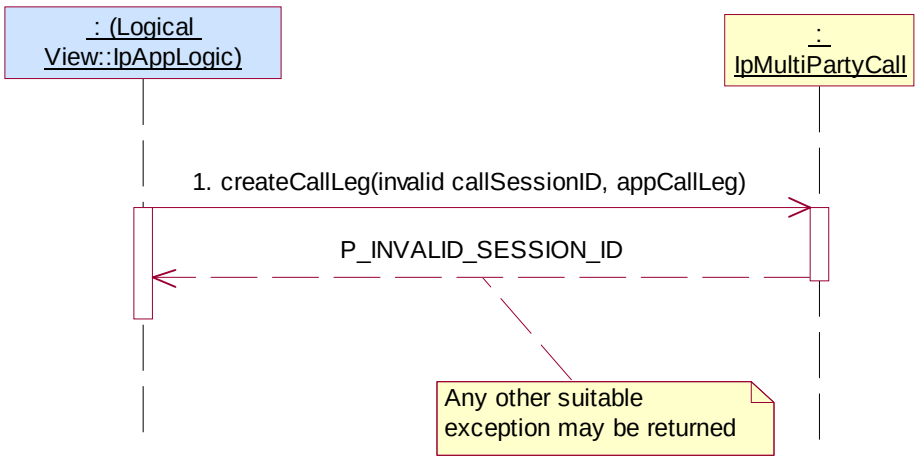
1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, non-null, value, valid notificationRequest
   Check: valid value of TpAssignmentID is returned

2. Triggered action: cause IUT to call Method **reportNotification()** method on the tester's (application) **IpAppMultiPartyCallControlManager** interface.
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID

Condition: createCallLeg method is supported.

Test Sequence:

1. Method call **createCallLeg()**
   Parameters: invalid callSessionID, valid appCallLeg
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MPCC_ IpMultiPartyCall _08

Summary: IpMultiPartyCall, createCallLeg, P_INVALID_INTERFACE_TYPE

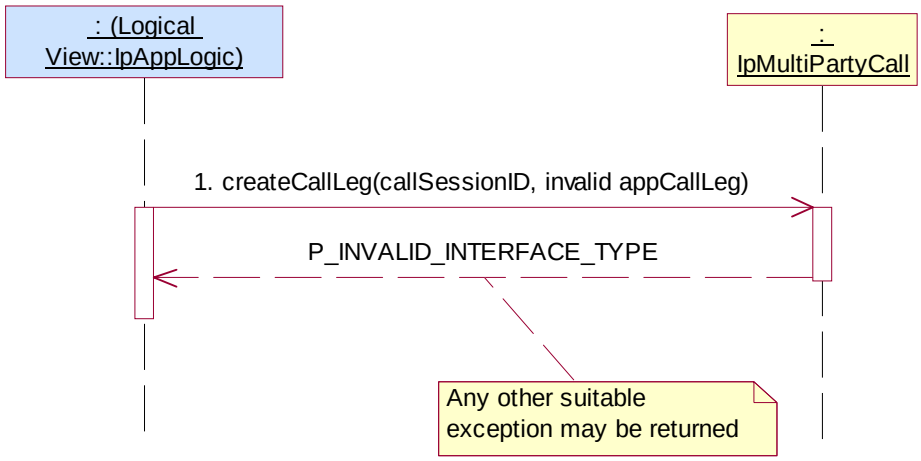
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as MPCC_ IpMultiPartyCall _07

Condition: createCallLeg method is supported.

Test Sequence:

1. Method call **createCallLeg()**
   Parameters: valid callSessionID reported in preamble, invalid appCallLeg
   Check: P_INVALID_INTERFACE_TYPE, or another suitable exception, is returned



Test MPCC_ IpMultiPartyCall _09

Summary: IpMultiPartyCall, createAndRouteCallLegReq, P_INVALID_SESSION_ID

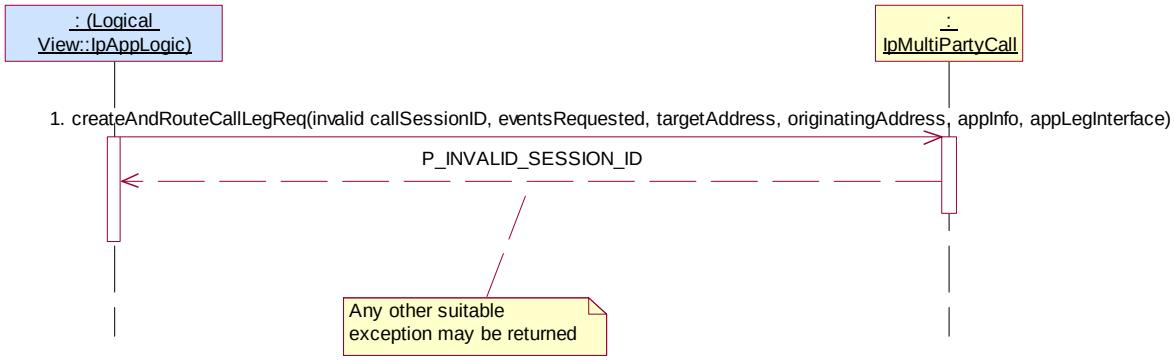
Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Same as MPCC_ IpMultiPartyCall _07

Condition: createAndRouteCallLegReq method is supported.

Test Sequence:

1. Method call **createAndRouteCallLegReq()**
   Parameters: invalid callSessionID, valid eventsRequested, valid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MPCC_ IpMultiPartyCall _10

Summary: IpMultiPartyCall, createAndRouteCallLegReq, P_INVALID_INTERFACE_TYPE

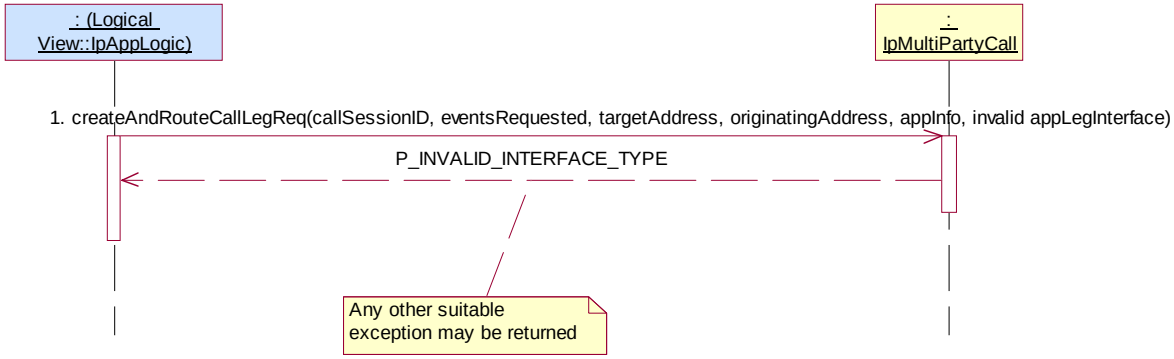
Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Same as MPCC_ IpMultiPartyCall _07

Condition: createAndRouteCallLegReq method is supported.

Test Sequence:

1. Method call **createAndRouteCallLegReq()**
   Parameters: valid callSessionID, valid eventsRequested, valid targetAddress, valid originatingAddress, valid appInfo, invalid appLegInterface
   Check: P_INVALID_INTERFACE_TYPE, or another suitable exception, is returned



Test MPCC_ IpMultiPartyCall _11

Summary: IpMultiPartyCall, createAndRouteCallLegReq, P_INVALID_ADDRESS

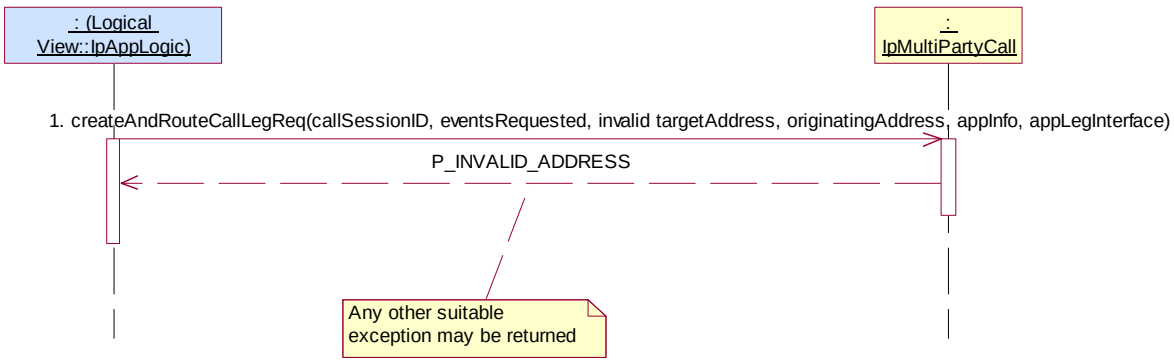
Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Same as MPCC_ IpMultiPartyCall _07

Condition: createAndRouteCallLegReq method is supported.

Test Sequence:

1. Method call **createAndRouteCallLegReq()**
   Parameters: valid callSessionID, valid eventsRequested, invalid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: P_INVALID_ADDRESS, or another suitable exception, is returned



Test MPCC_ IpMultiPartyCall _12

Summary: IpMultiPartyCall, createAndRouteCallLegReq, P_INVALID_ADDRESS

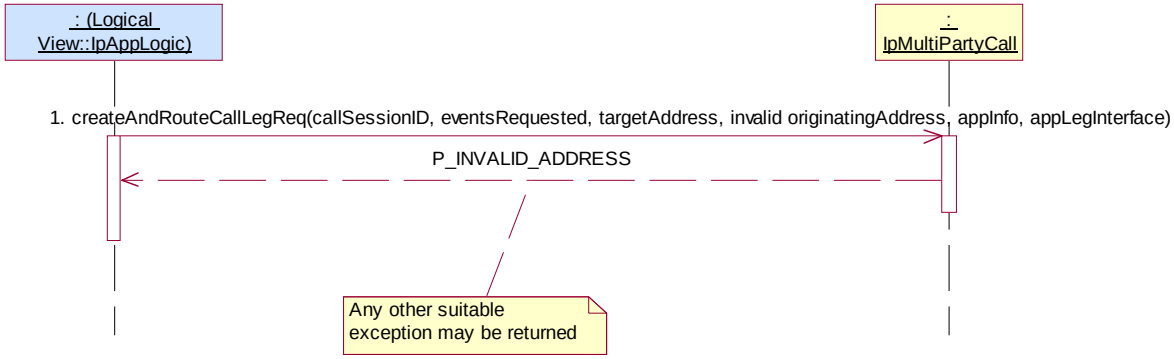
Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Same as MPCC_ IpMultiPartyCall _07

Condition: createAndRouteCallLegReq method is supported.

Test Sequence:

1. Method call **createAndRouteCallLegReq()**
   Parameters: valid callSessionID, valid eventsRequested, valid targetAddress, invalid originatingAddress, valid appInfo, valid appLegInterface
   Check: P_INVALID_ADDRESS, or another suitable exception, is returned



Test MPCC_ IpMultiPartyCall _13

Summary: IpMultiPartyCall, createAndRouteCallLegReq, P_INVALID_CRITERIA

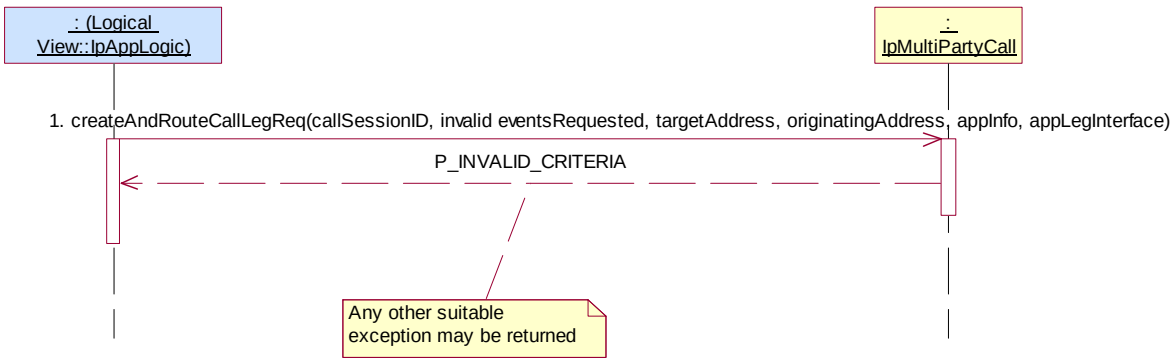
Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Same as MPCC_ IpMultiPartyCall _07

Condition: createAndRouteCallLegReq method is supported.

Test Sequence:

1. Method call **createAndRouteCallLegReq()**
   Parameters: valid callSessionID, invalid eventsRequested, valid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: P_INVALID_CRITERIA, or another suitable exception, is returned



5.2.2.2.3 Optional, valid behaviour

Test MPCC_ IpMultiPartyCall _14

Summary: IpMultiPartyCall, getInfoReq, successful

Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Application has a valid callSessionID returned by one of the two following sequence:

1. Method call **setCallback()** on IpMultiPartyCallControlManager
   Parameters: valid, non-null, value of appInterface parameter
   Check: no exception is returned

2. Method call **createCall()**
   Parameters: valid appCall
   Check: valid value of TpMultiPartyCallIdentifier is returned

3. Method call **createCallLeg()**
   Parameters: valid callSessionID returned in 2., valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

or

1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, non-null, value, valid notificationRequest
   Check: valid value of TpAssignmentID is returned

2. Triggered action: cause IUT to call Method **reportNotification()** method on the tester's (application) **IpAppMultiPartyCallControlManager** interface.
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID

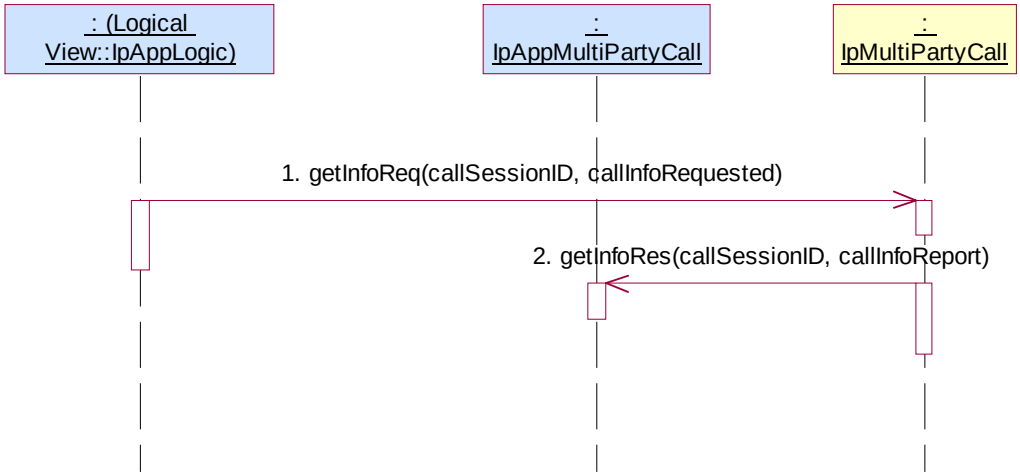
3. Method call **createCallLeg ()**
   Parameters: valid callSessionID reported in 2., valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

Condition: createCallLeg and getInfoReq methods are supported.

Test Sequence:

1. Method call **getInfoReq()**
   Parameters: valid callSessionID reported inpreamble, valid callInfoRequested
   Check: no exception is returned

2 Triggered action: cause IUT to call **getInfoRes()** method on the tester's (Application) **IpAppMultiPartyCall** interface.
   Parameters: callSessionID given in 1., valid callInfoReport.
   Check: no exception is returned



Test MPCC_ IpMultiPartyCall _15

Summary: IpMultiPartyCall, setChargePlan, successful

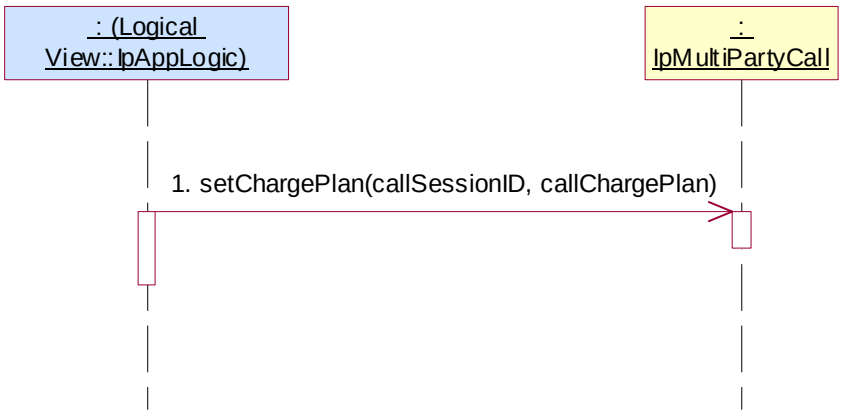
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as MPCC_ IpMultiPartyCall _14

Condition: createCallLeg and setChargePlan methods are supported.

Test Sequence:

1. Method call **setChargePlan()**
   Parameters: valid callSessionID reported in preamble, valid callChargePlan
   Check: no exception is returned



Test MPCC_ IpMultiPartyCall _16

Summary: IpMultiPartyCall, setAdviceOfCharge, successful

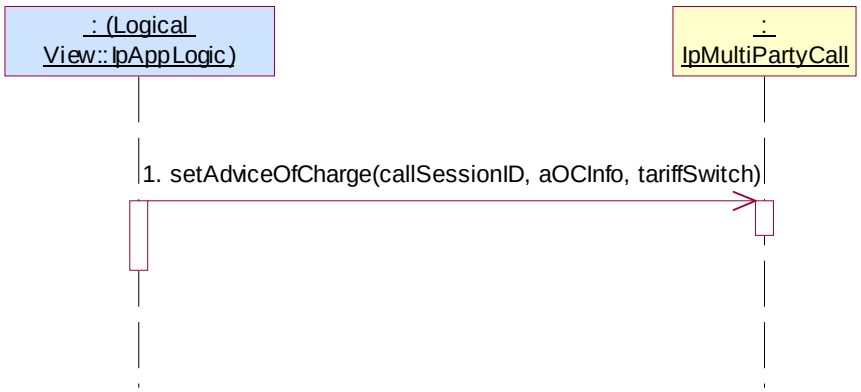
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as MPCC_ IpMultiPartyCall _14

Condition: createCallLeg and setAdviceOfCharge method are supported.

Test Sequence:

1. Method call **setAdviceOfCharge()**
   Parameters: valid callSessionID reported in preamble, valid aOCInfo, valid tariffSwitch
   Check: no exception is returned



Test MPCC_ IpMultiPartyCall _17

Summary: IpMultiPartyCall, superviseReq, successful

Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as MPCC_ IpMultiPartyCall _14

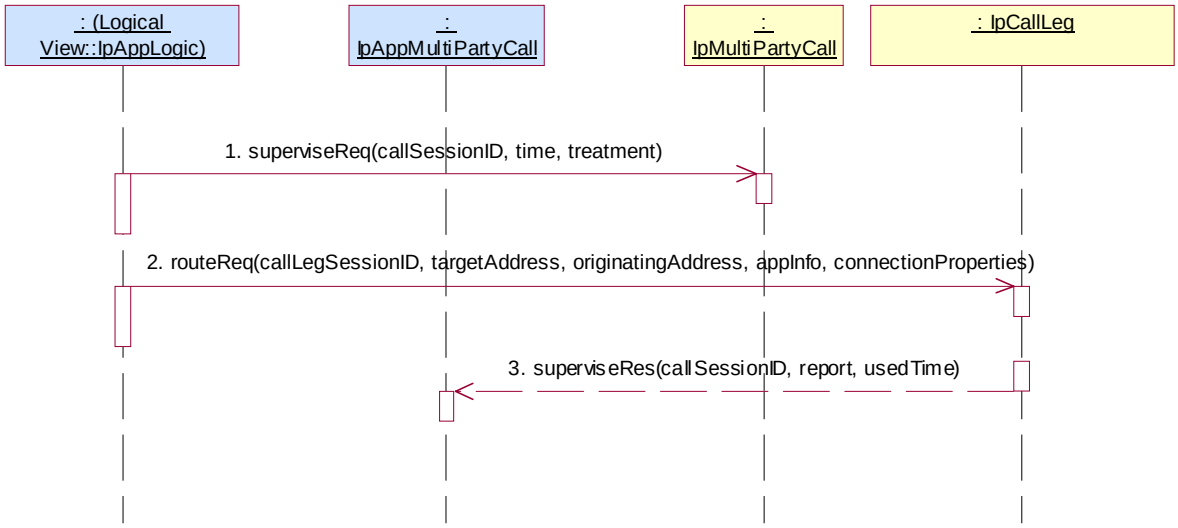
Condition: createCallLeg and superviseReq methods are supported.

Test Sequence:

1. Method call **superviseReq()**
   Parameters: valid callSessionID reported inpreamble, valid time, valid treatment
   Check: no exception is returned

2. Method call **routeReq()**
   Parameters: valid callLegSessionID reported in preamble, valid targetAddress, valid originatingAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

3 Triggered action: cause IUT to call **superviseRes()** method on the tester's (Application) **IpAppMultiPartyCall** interface.
   Parameters: callSessionID given in preamble, valid report, valid usedTime.



Test MPCC_ IpMultiPartyCall _18

Summary: IpMultiPartyCall, getCallLegs, successful

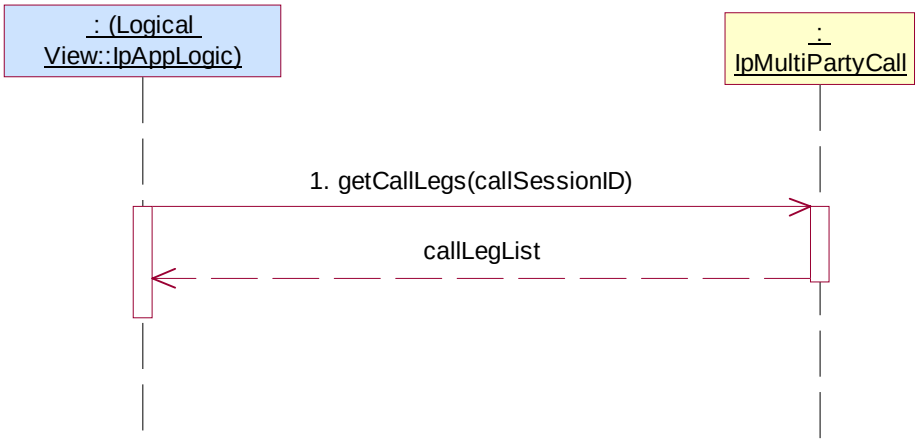
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as MPCC_ IpMultiPartyCall _03

Condition: getCallLegs method is supported.

Test Sequence:

1. Method call **getCallLegs()**
   Parameters: valid callSessionID reported in preamble.
   Check: valid value of TpCallLegIdentifierSet which contains CallLegIdentifier returned in preamble.



5.2.2.2.4 Optional, invalid behaviour

Test MPCC_ IpMultiPartyCall _19

Summary: IpMultiPartyCall, getInfoReq, P_INVALID_SESSION_ID

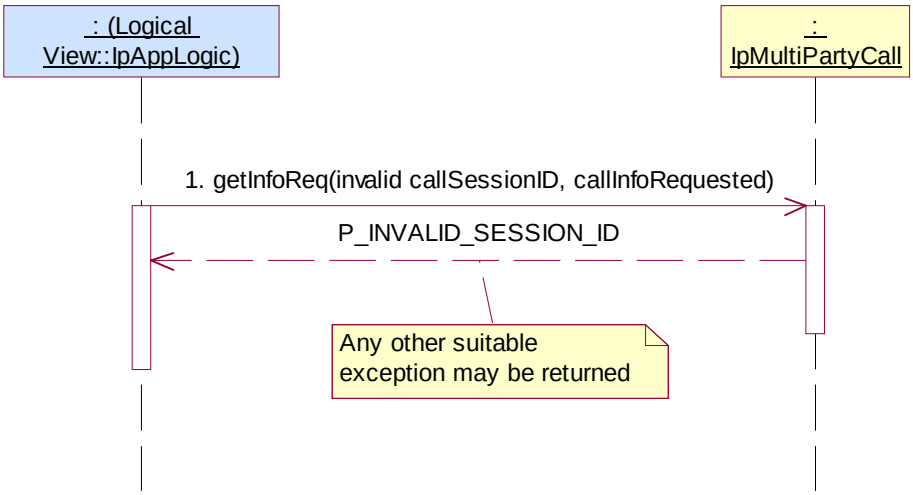
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as MPCC_ IpMultiPartyCall _14

Condition: createCallLeg and getInfoReq methods are supported.

Test Sequence:

1. Method call **getInfoReq()**
   Parameters: invalid callSessionID, valid callInfoRequested
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MPCC_ IpMultiPartyCall _20

Summary: IpMultiPartyCall, setChargePlan, P_INVALID_SESSION_ID

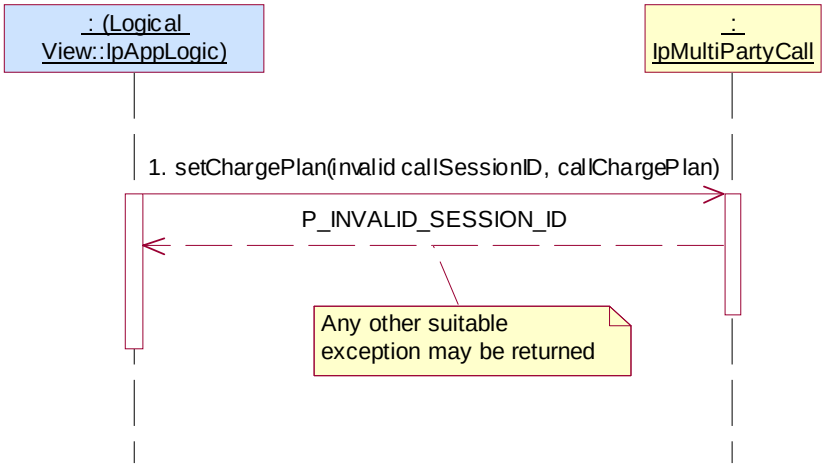
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as MPCC_ IpMultiPartyCall _14

Condition: createCallLeg and setChargePlan methods are supported.

Test Sequence:

1. Method call **setChargePlan()**
   Parameters: invalid callSessionID., valid callChargePlan
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MPCC_ IpMultiPartyCall _21

Summary: IpMultiPartyCall, setAdviceOfCharge, P_INVALID_SESSION_ID

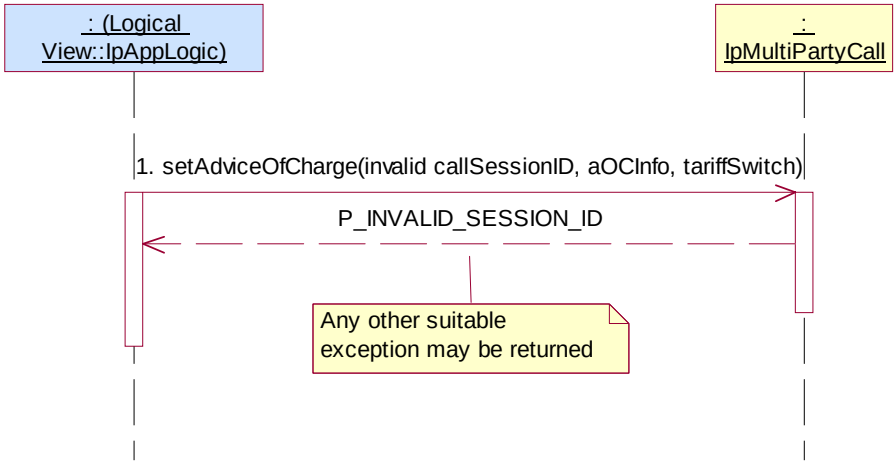
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as MPCC_ IpMultiPartyCall _14

Condition: createCallLeg and setAdviceOfCharge methods are supported.

Test Sequence:

1. Method call **setAdviceOfCharge()**
   Parameters: invalid callSessionID, valid aOCInfo, valid tariffSwitch
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MPCC_ IpMultiPartyCall _22

Summary: IpMultiPartyCall, setAdviceOfCharge, P_INVALID_CURRENCY

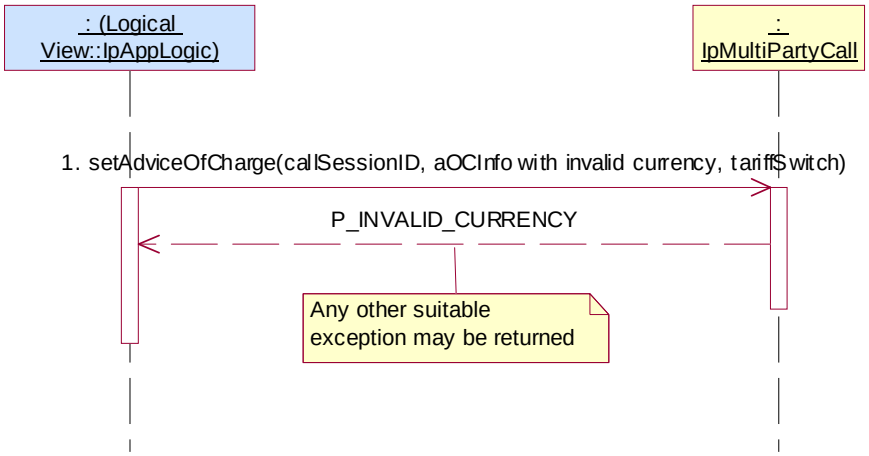
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as MPCC_ IpMultiPartyCall _14

Condition: createCallLeg and setAdviceOfCharge methods are supported.

Test Sequence:

1. Method call **setAdviceOfCharge()**
   Parameters: valid callSessionID reported in preamble, aOCInfo with invalid currency, valid tariffSwitch
   Check: P_INVALID_CURRENCY, or another suitable exception, is returned.



Test MPCC_ IpMultiPartyCall _23

Summary: IpMultiPartyCall, setAdviceOfCharge, P_INVALID_AMOUNT

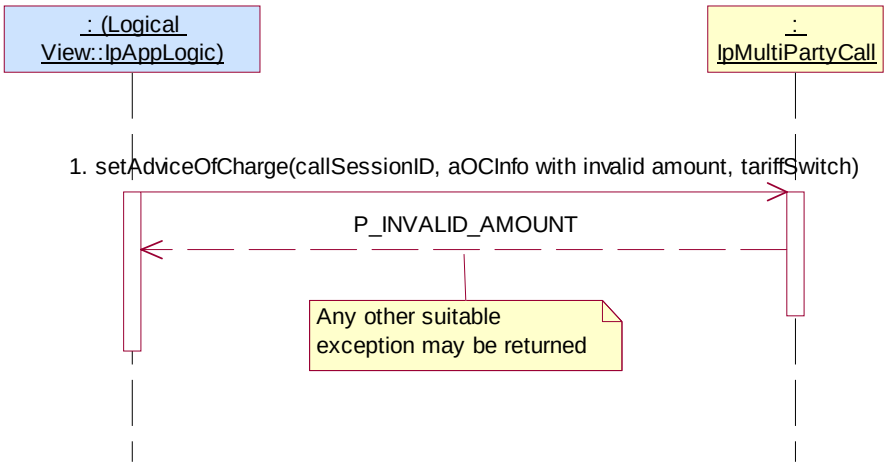
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as MPCC_ IpMultiPartyCall _14

Condition: createCallLeg and setAdviceOfCharge methods are supported.

Test Sequence:

1. Method call **setAdviceOfCharge()**
   Parameters: valid callSessionID reported in preamble, aOCInfo, with invalid amount, valid tariffSwitch
   Check: P_INVALID_AMOUNT, or another suitable exception, is returned.



Test MPCC_ IpMultiPartyCall _24

Summary: IpMultiPartyCall, superviseReq, P_INVALID_SESSION_ID

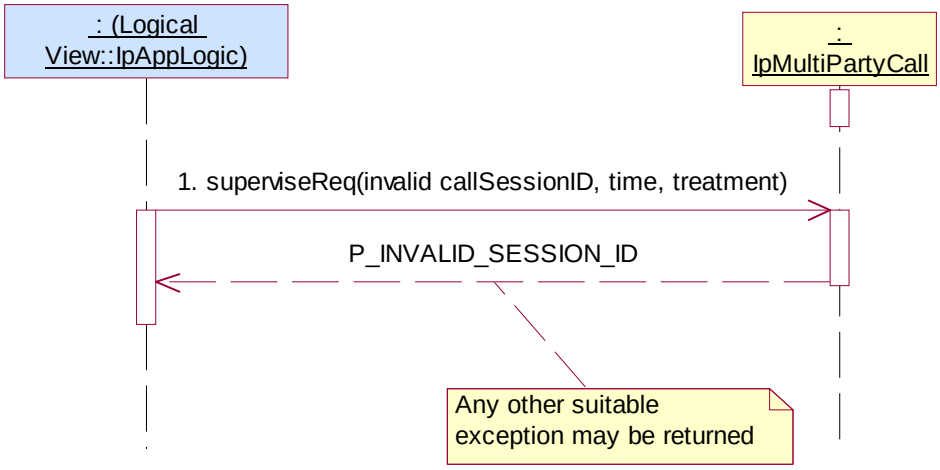
Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Same as MPCC_ IpMultiPartyCall _14

Condition: createCallLeg and superviseReq methods are supported.

Test Sequence:

1. Method call **superviseReq()**
   Parameters: invalid callSessionID, valid time, valid treatment
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MPCC_ IpMultiPartyCall _25

Summary: IpMultiPartyCall, getCallLegs, P_INVALID_SESSION_ID

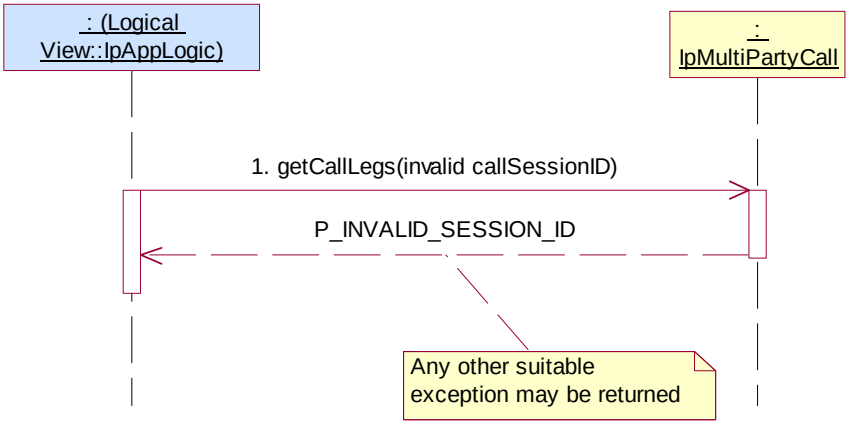
Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Same as MPCC_ IpMultiPartyCall _07

Condition: getCallLegs method is supported.

Test Sequence:

1. Method call **getCallLegs()**
   Parameters: invalid callSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



5.2.2.3 IpCallLeg

5.2.2.3.1 Mandatory, valid behaviour

Test MPCC_ IpCallLeg _01

Summary: IpCallLeg, all mandatory methods, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as MPCC_ IpMultiPartyCall _07

Condition: createCallLeg method is supported.

Test Sequence:

1. Method call **createCallLeg()**
   Parameters: valid callSessionID reported in preamble, valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

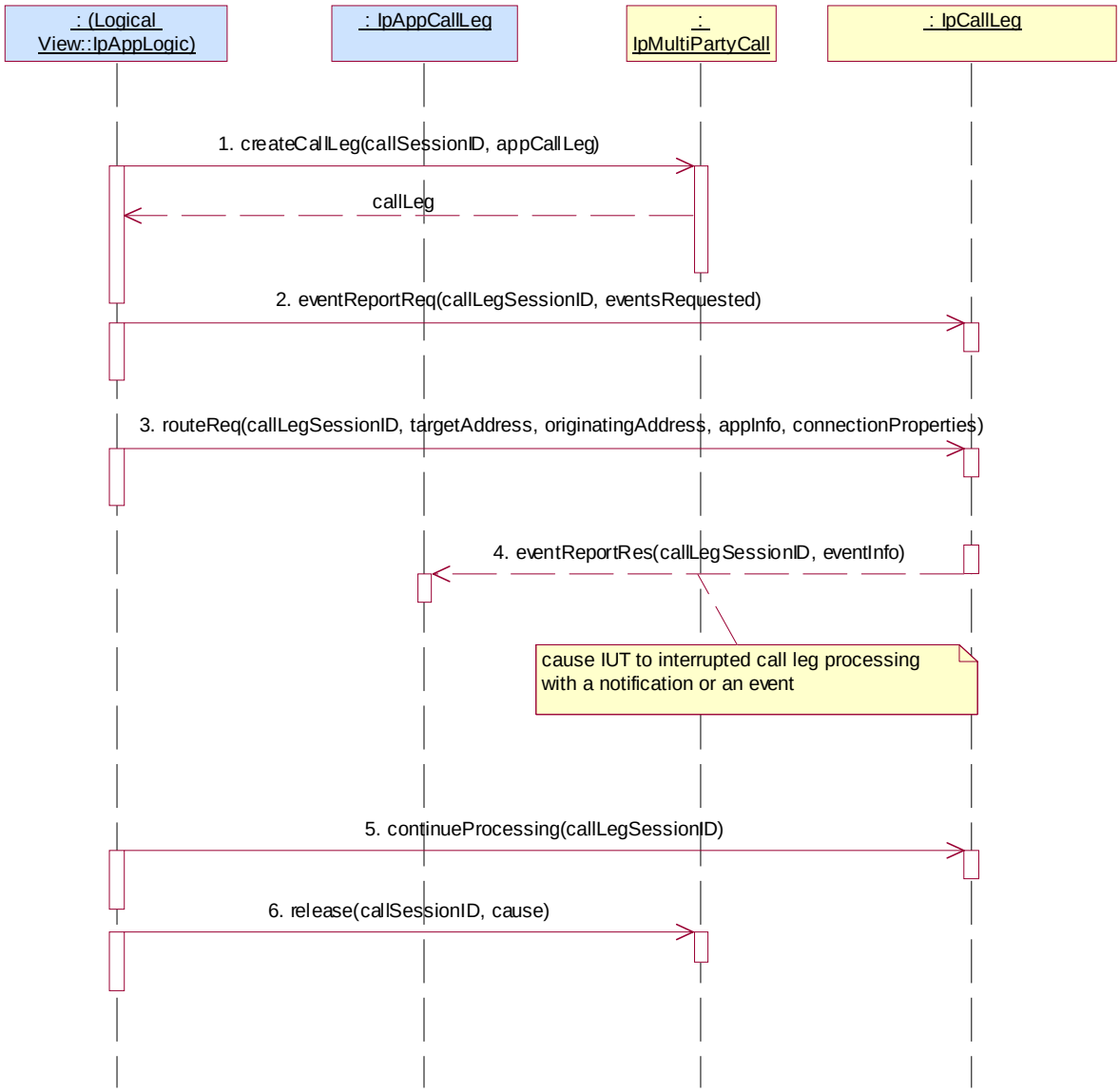
2. Method call **eventReportReq()**
   Parameters: valid callLegSessionID returned in 1, valid eventsRequested
   Check: no exception is returned

3. Method call **routeReq()**
   Parameters: valid callLegSessionID returned in 1, valid targetAddress, valid originatingAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

4 Triggered action: cause IUT to interrupted call leg processing with a notification or an event: cause IUT to call **eventReportRes()** method on the tester's (Application) **IpAppCallLeg** interface.
   Parameters: callLegSessionID, errorIndication

5. Method call **continueProcessing()**
   Parameters: valid callLegSessionID returned in 1.
   Check: no exception is returned

6. Method call **release()**
   Parameters: valid callLegSessionID returned in 1, valid cause
   Check: no exception is returned


: (Logical View::IpAppLogic)
: IpAppCallLeg
: IpMultiPartyCall
: IpCallLeg
1. createCallLeg(callSessionID, appCallLeg)
callLeg
2. eventReportReq(callLegSessionID, eventsRequested)
3. routeReq(callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties)
4. eventReportRes(callLegSessionID, eventInfo)
cause IUT to interrupted call leg processing with a notification or an event
5. continueProcessing(callLegSessionID)
6. release(callSessionID, cause)

Test MPCC_ IpCallLeg _02

Summary: IpCallLeg, all mandatory methods, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as MPCC_ IpMultiPartyCall _07

Condition: createAndRouteCallLeg method is supported.

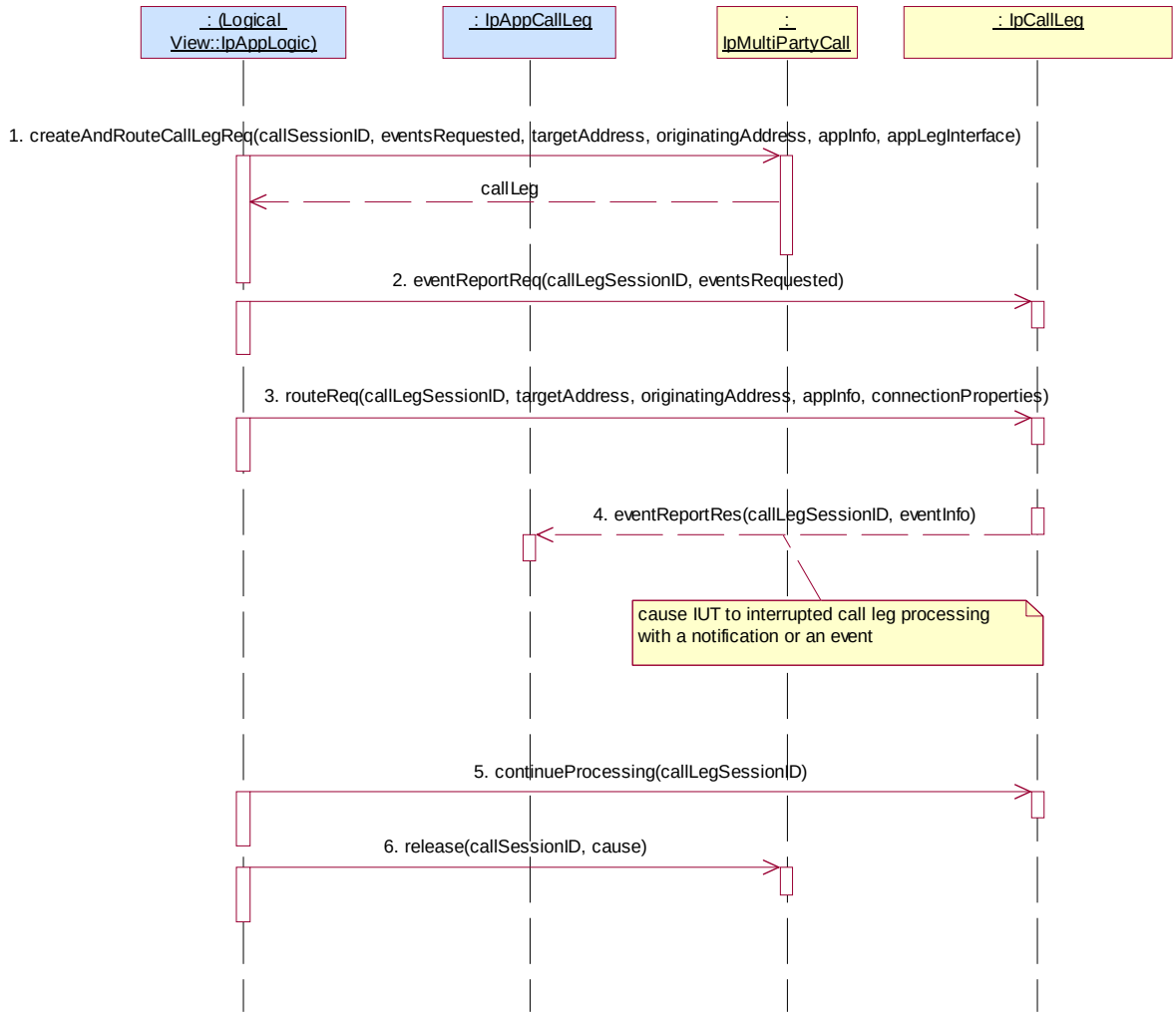
Test Sequence:

1. Method call **createAndRouteCallLegReq()**
   Parameters: valid callSessionID reported in preamble, valid eventsRequested, valid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: valid value of TpCallLegIdentifier

2. Triggered action: cause IUT to interrupted call leg processing with a notification or an event: cause IUT to call **eventReportRes()** method on the tester's (Application) **IpAppCallLeg** interface.
   Parameters: callLegSessionID, errorIndication

3. Method call **continueProcessing()**
   Parameters: valid callLegSessionID returned in 1.
   Check: no exception is returned

4. Method call **release()**
   Parameters: valid callLegSessionID returned in 1, valid cause
   Check: no exception is returned



Test MPCC_ IpCallLeg _03

Summary: IpCallLeg, all mandatory methods, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as MPCC_ IpMultiPartyCall _03

Test Sequence:

1. Method call **deassign()**
   Parameters: valid callLegSessionID returned in preamble.
   Check: no exception is returned

5.2.2.3.2 Mandatory, invalid behaviour

Test MPCC_ IpCallLeg _04

Summary: IpCallLeg, continueProcessing, P_INVALID_SESSION_ID

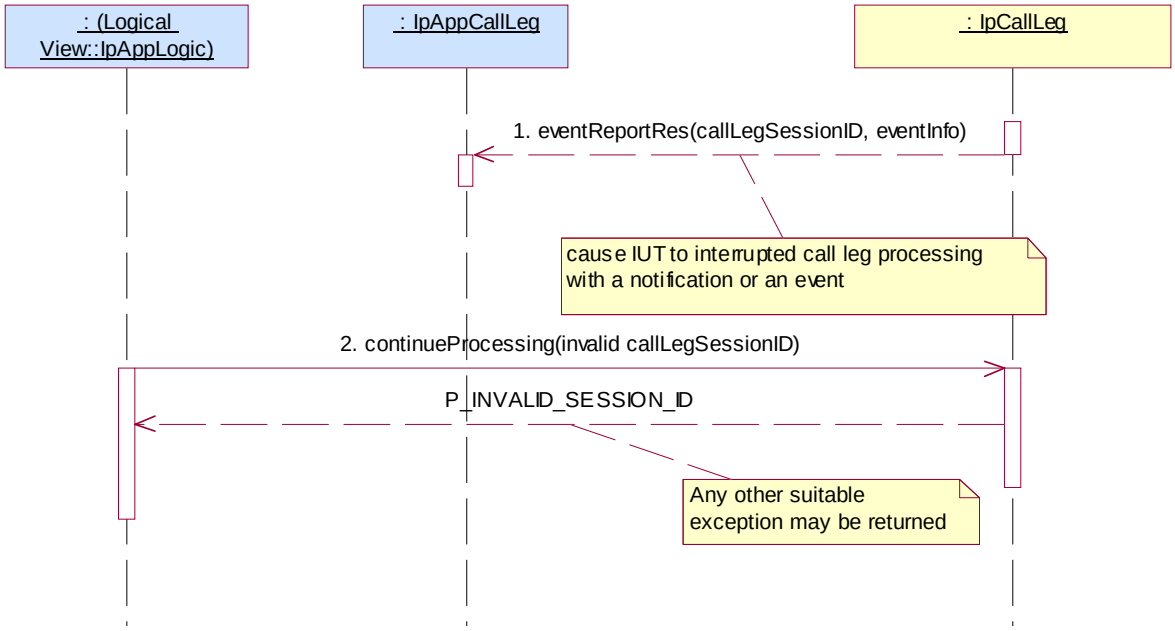
Reference: ES 201 915-4 [1], clause 7.3.5

Preamble: Same as MPCC_ IpMultiPartyCall _03

Test Sequence:

1 Triggered action: cause IUT to interrupted call leg processing with a notification or an event: cause IUT to call **eventReportRes()** method on the tester's (Application) **IpAppCallLeg** interface.
Parameters: callLegSessionID, errorIndication

2. Method call **continueProcessing()**
Parameters: invalid callLegSessionID
Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MPCC_ IpCallLeg _05

Summary: IpCallLeg, routeReq, P_INVALID_SESSION_ID

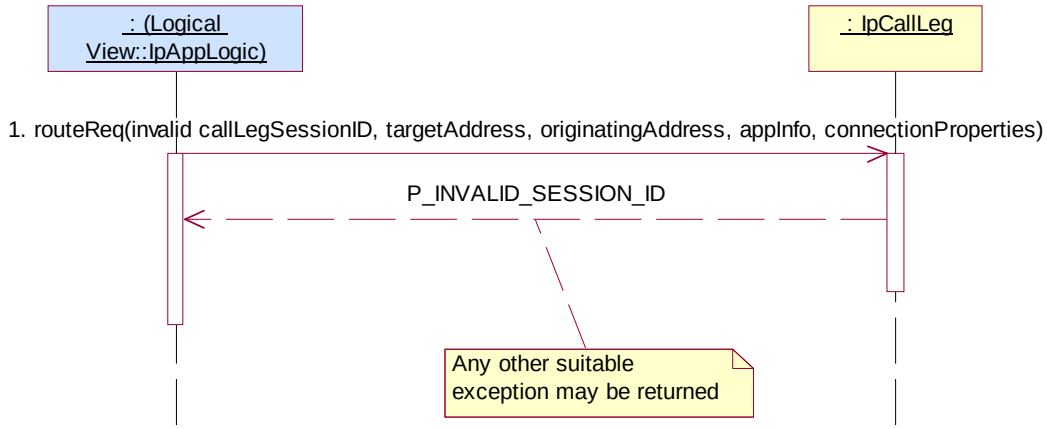
Reference: ES 201 915-4 [1], clause 7.3.5

Preamble: Same as MPCC_ IpMultiPartyCall _14

Condition: createCallLeg method is supported.

Test Sequence:

1. Method call **routeReq()**
   Parameters: invalid callLegSessionID, valid targetAddress, valid originatingAddress, valid appInfo, valid connectionProperties
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MPCC_ IpCallLeg _06

Summary: IpCallLeg, routeReq, P_INVALID_ADDRESS

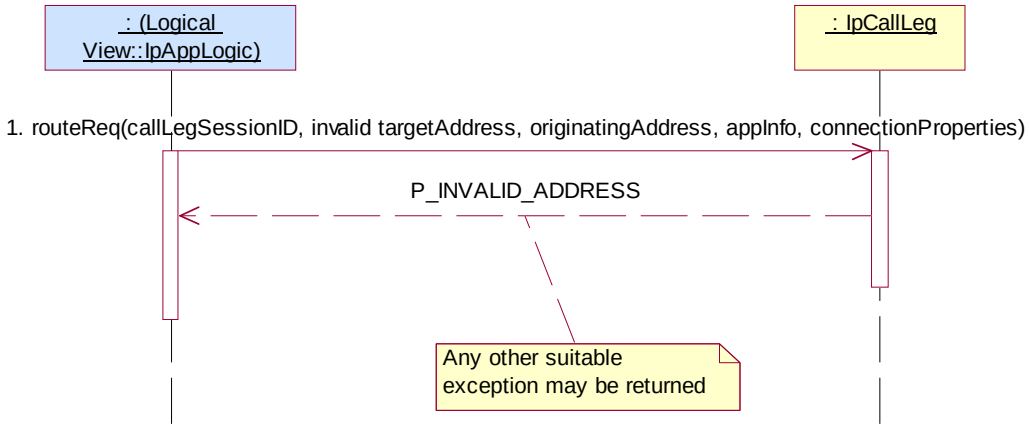
Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as MPCC_ IpMultiPartyCall _14

Condition: createCallLeg method is supported.

Test Sequence:

1. Method call **routeReq()**
   Parameters: valid callLegSessionID returned in preamble, invalid targetAddress, valid originatingAddress, valid appInfo, valid connectionProperties
   Check: P_INVALID_ADDRESS, or another suitable exception, is returned



Test MPCC_ IpCallLeg _07

Summary: IpCallLeg, routeReq, P_INVALID_ADDRESS

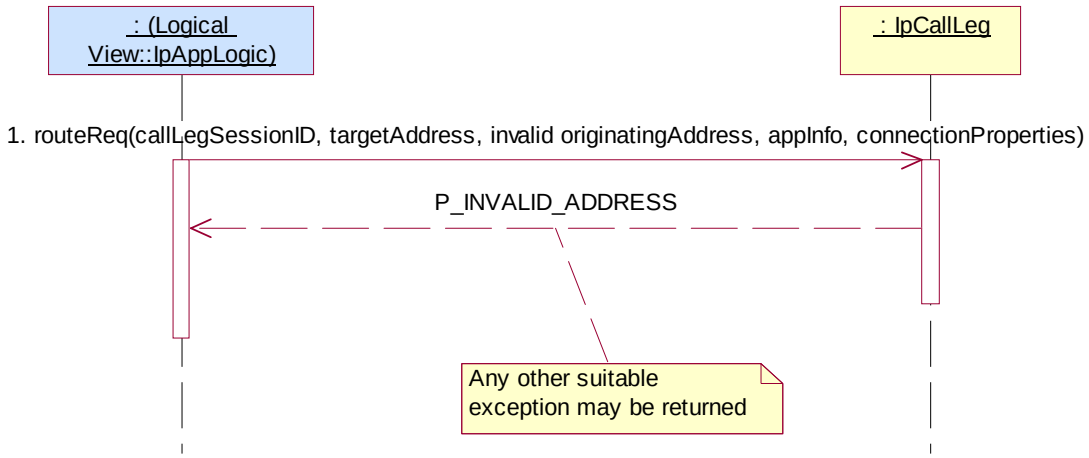
Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as MPCC_ IpMultiPartyCall _14

Condition: createCallLeg method is supported.

Test Sequence:

1. Method call **routeReq()**
   Parameters: valid callLegSessionID returned in preamble, valid targetAddress, invalid originatingAddress, valid appInfo, valid connectionProperties
   Check: P_INVALID_ADDRESS, or another suitable exception, is returned



Test MPCC_ IpCallLeg _08

Summary: IpCallLeg, eventReportReq**,** P_INVALID_SESSION_ID

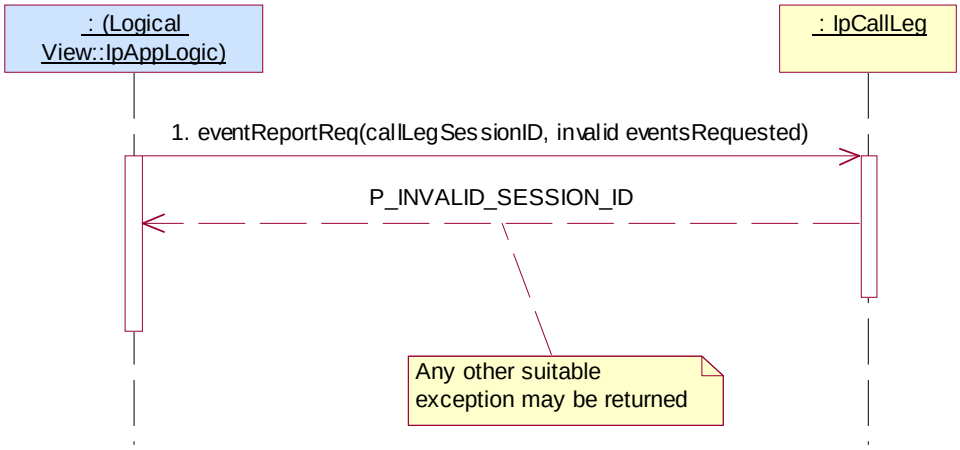
Reference: ES 201 915-4 [1], clause 7.3.5

Preamble: Same as MPCC_ IpMultiPartyCall _07

Condition: createCallLeg and eventReportReq methods are supported.

Test Sequence:

1. Method call **eventReportReq()**
   Parameters: invalid callLegSessionID, valid eventsRequested
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MPCC_ IpCallLeg _09

Summary: IpCallLeg, eventReportReq, P_INVALID_CRITERIA

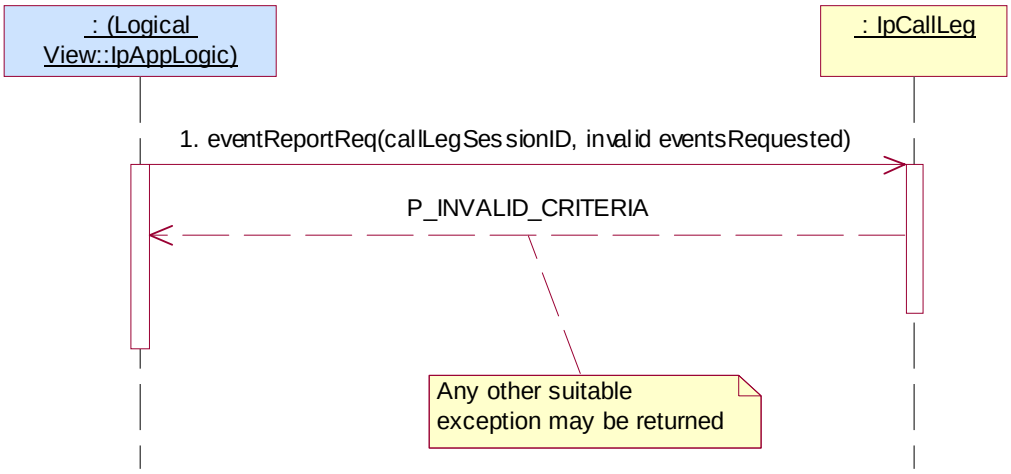
Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as MPCC_ IpMultiPartyCall _07

Condition: createCallLeg and eventReportReq methods are supported.

Test Sequence:

1. Method call **eventReportReq()**
   Parameters: valid callLegSessionID returned in preamble, invalid eventsRequested
   Check: P_INVALID_CRITERIA, or another suitable exception, is returned



Test MPCC_ IpCallLeg _10

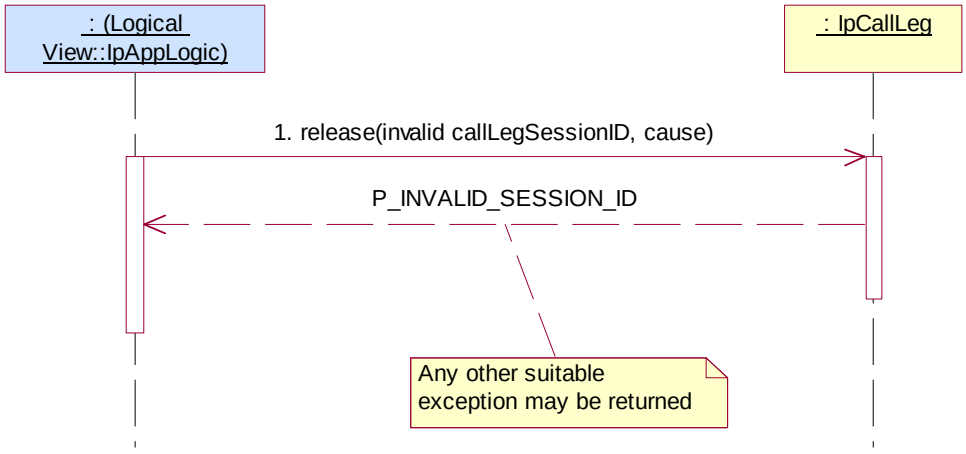
Summary: IpCallLeg, release**,** P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.5

Preamble: Same as MPCC_ IpMultiPartyCall _03

Test Sequence:

1. Method call **release()**
   Parameters: invalid callLegSessionID, valid cause
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MPCC_ IpCallLeg _11

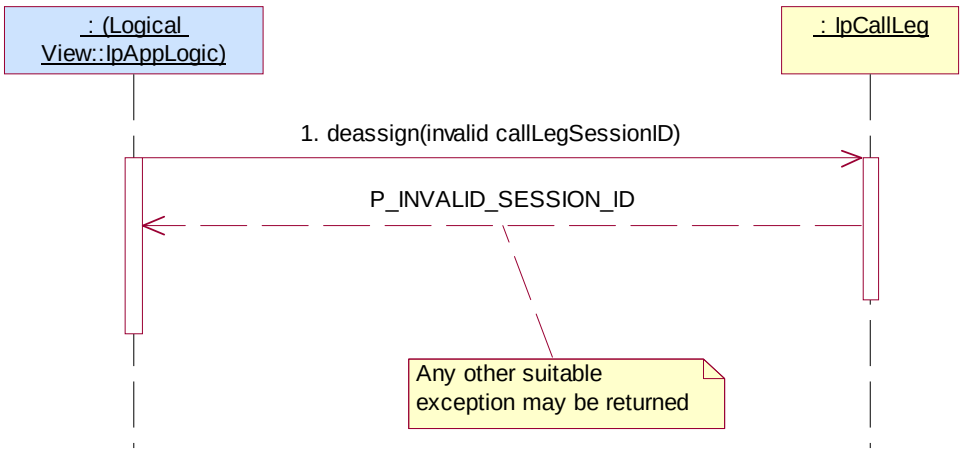
Summary: IpCallLeg, deassign**,** P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.5

Preamble: Same as MPCC_ IpMultiPartyCall _03

Test Sequence:

1. Method call **deassign()**
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned.



5.2.2.3.3 Optional, valid behaviour

Test MPCC_ IpCallLeg _12

Summary: IpCallLeg, getInfoReq, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as MPCC_ IpMultiPartyCall _14

Condition: createCallLeg and getInfoReq methods are supported.

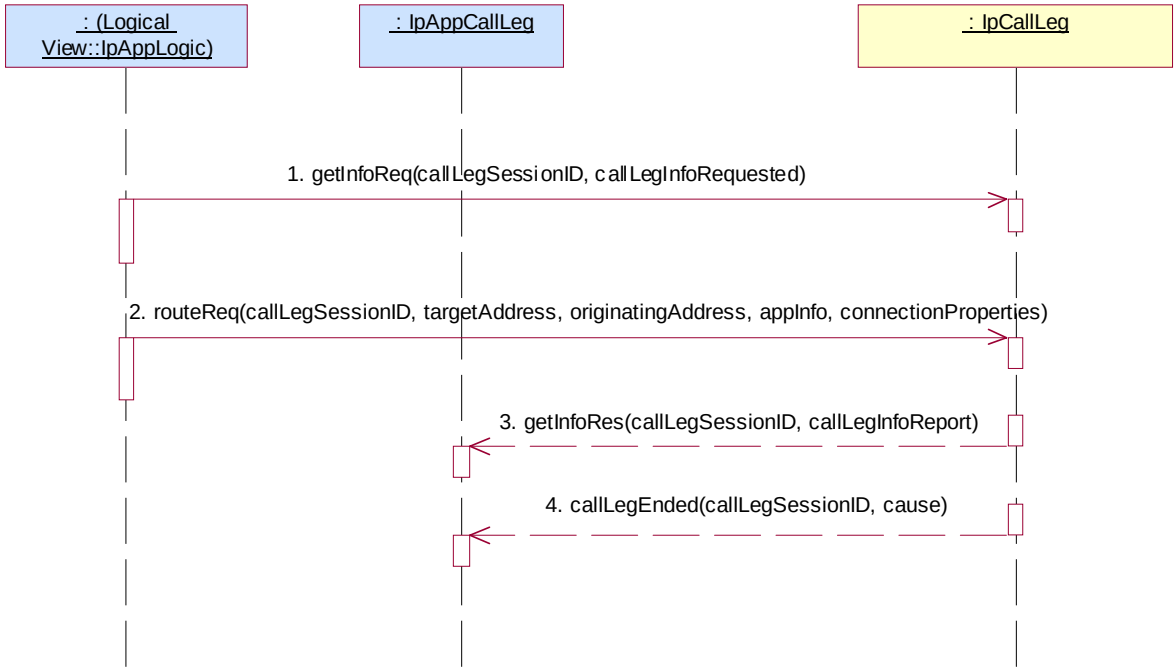
Test Sequence:

1. Method call **getInfoReq()**
   Parameters: valid callLegSessionID returned in preamble, valid callLegInfoRequested
   Check: no exception is returned

2. Method call **routeReq()**
   Parameters: valid callLegSessionID returned in preamble, valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

2 Triggered action: cause IUT to call **getInfoRes()** method on the tester's (Application) **IpAppCallLeg** interface.
Parameters: callLegSessionID given in 1., valid callLegInfoReport.

3 Triggered action: cause IUT to call **callLegEnded()** method on the tester's (Application) **IpAppCallLeg** interface.
Parameters: callLegSessionID given in 1, cause



Test MPCC_ IpCallLeg _13

Summary: IpCallLeg, attachMediaReq, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as MPCC_ IpMultiPartyCall _07

Condition: createCallLeg and attachMediaReq methods are supported.

Test Sequence:

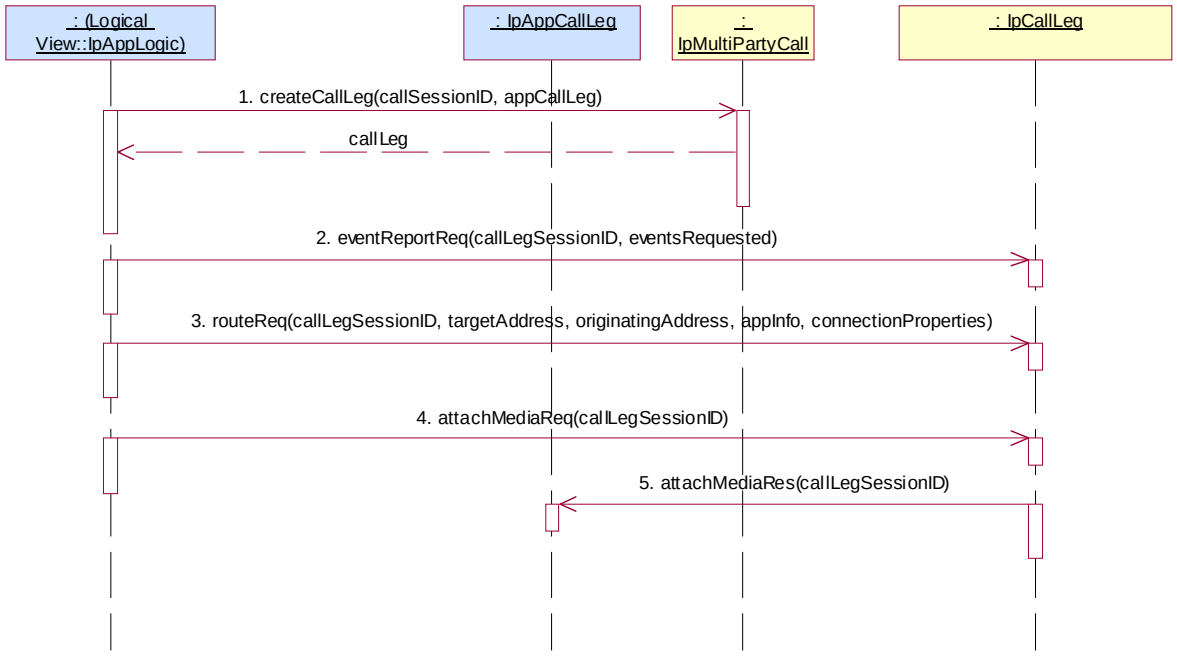
1. Method call **createCallLeg()**
   Parameters: valid callSessionID reported in preamble, valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

2. Method call **eventReportReq()**
   Parameters: valid callLegSessionID returned in 1, valid eventsRequested
   Check: no exception is returned

3. Method call **routeReq()**
   Parameters: valid callLegSessionID returned in 1, valid targetAddress, valid originatingAddress, valid appInfo, valid connectionProperties set to have explicit media management
   Check: no exception is returned

4. Method call **attachMediaReq()**
   Parameters: valid callLegSessionID returned in 1.
   Check: no exception is returned

5 Triggered action: cause IUT to call **attachMediaRes()** method on the tester's (Application) **IpAppCallLeg** interface.
   Parameters: callLegSessionID



Test MPCC_ IpCallLeg _14

Summary: IpCallLeg, detachMediaReq, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as MPCC_ IpMultiPartyCall _07

Condition: createCallLeg, attachMediaReq and detachMediaReq methods are supported.

Test Sequence:

1. Method call **createCallLeg()**
   Parameters: valid callSessionID reported in preamble, valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

2. Method call **eventReportReq()**
   Parameters: valid callLegSessionID returned in 1, valid eventsRequested
   Check: no exception is returned

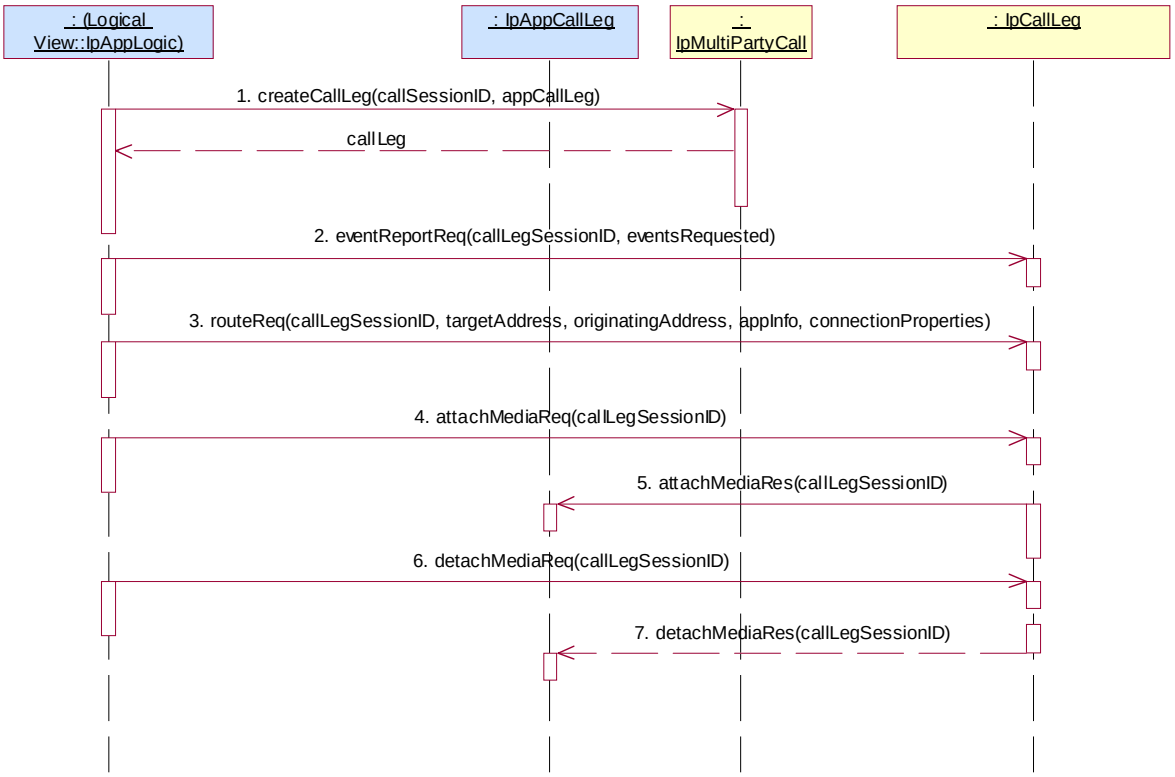
3. Method call **routeReq()**
   Parameters: valid callLegSessionID returned in 1, valid targetAddress, valid originatingAddress, valid appInfo, valid connectionProperties set to have explicit media management
   Check: no exception is returned

4. Method call **attachMediaReq()**
   Parameters: valid callLegSessionID returned in 1.
   Check: no exception is returned

5 Triggered action: cause IUT to call **attachMediaRes()** method on the tester's (Application) **IpAppCallLeg** interface.
   Parameters: callLegSessionID

6. Method call **detachMediaReq()**
   Parameters: valid callLegSessionID returned in 1.
   Check: no exception is returned

7 Triggered action: cause IUT to call **detachMediaRes()** method on the tester's (Application) **IpAppCallLeg** interface.
   Parameters: callLegSessionID



Test MPCC_ IpCallLeg _15

Summary: IpCallLeg, getCurrentDestinationAddress, successful

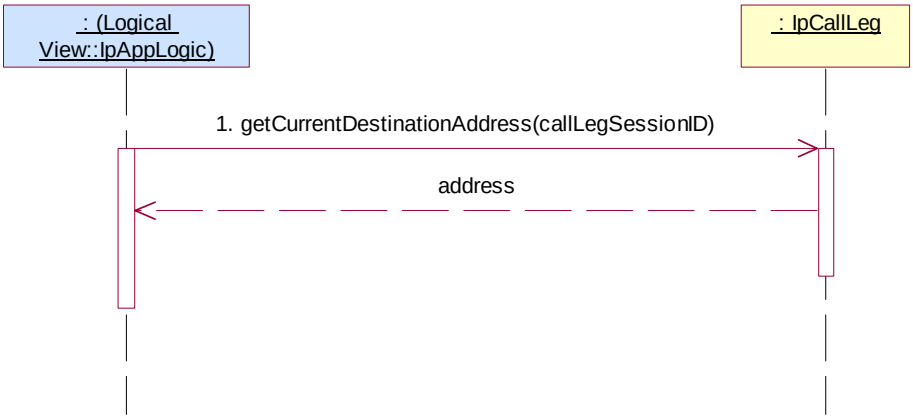
Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as MPCC_ IpMultiPartyCall _03

Condition: getCurrentDestinationAddress method is supported.

Test Sequence:

1. Method call **getCurrentDestinationAddress()**
   Parameters: valid callLegSessionID returned in preamble.
   Check: valid value of TpAddress is returned



Test MPCC_ IpCallLeg _16

Summary: IpCallLeg, setChargePlan, successful

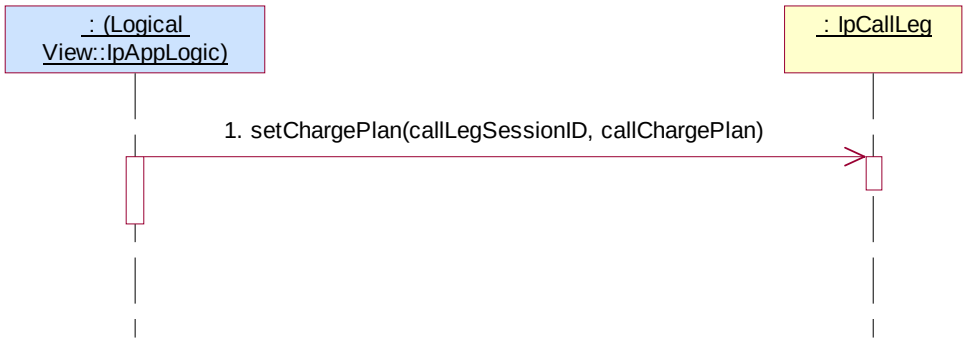
Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as MPCC_ IpMultiPartyCall _14

Condition: createCallLeg and setChargePlan methods are supported.

Test Sequence:

1. Method call **setChargePlan()**
   Parameters: valid callLegSessionID returned in preamble, valid callChargePlan
   Check: no exception is returned



Test MPCC_ IpCallLeg _17

Summary: IpCallLeg, setAdviceOfCharge, successful

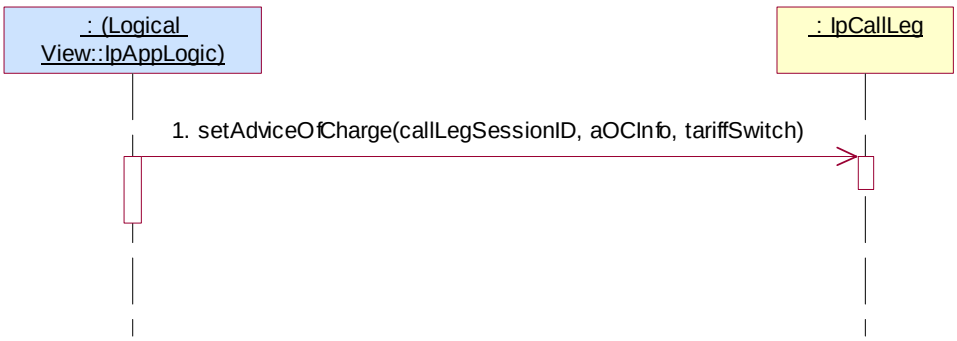
Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as MPCC_ IpMultiPartyCall _14

Condition: createCallLeg and setAdviceOfCharge methods are supported.

Test Sequence:

1. Method call **setAdviceOfCharge()**
   Parameters: valid callLegSessionID returned in preamble, valid aOCInfo, valid tariffSwitch
   Check: no exception is returned



Test MPCC_ IpCallLeg _18

Summary: IpCallLeg, superviseReq, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as MPCC_ IpMultiPartyCall _14

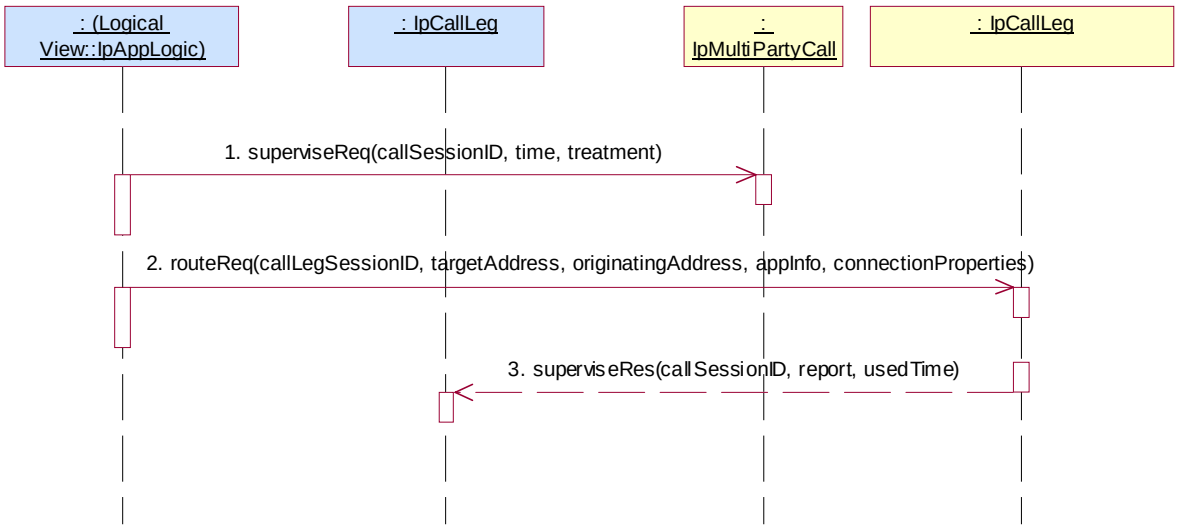
Condition: createCallLeg and superviseReq methods are supported.

Test Sequence:

1. Method call **superviseReq()**
   Parameters: valid callLegSessionID returned in preamble, valid time, valid treatment
   Check: no exception is returned

2. Method call **routeReq()**
   Parameters: valid callLegSessionID returned in preamble, valid targetAddress, valid originatingAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

3 Triggered action: cause IUT to call **superviseRes()** method on the tester's (Application) **IpAppCallLeg** interface.
   Parameters: callLegSessionID, report, usedTime



Test MPCC_ IpCallLeg _19

Summary: IpCallLeg, getCall, successful

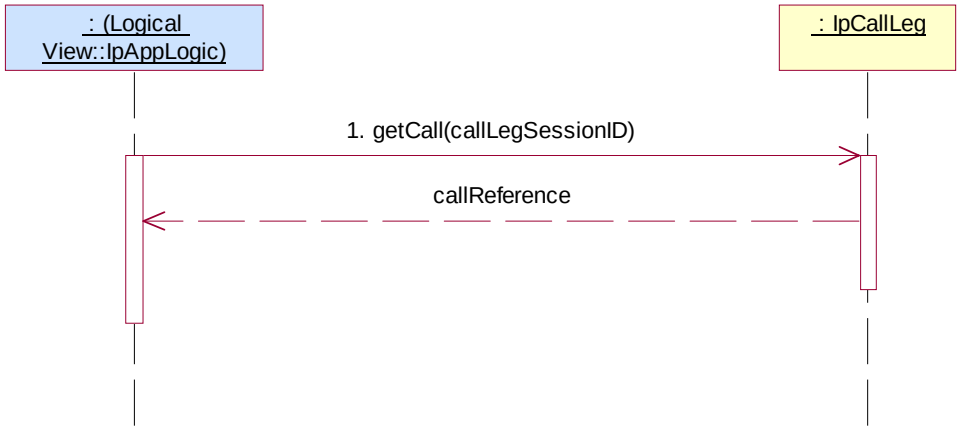
Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as MPCC_ IpMultiPartyCall _03

Condition: getCall method is supported.

Test Sequence:

1. Method call **getCall()**
   Parameters: valid callLegSessionID returned in preamble.
   Check: valid TpMultiPartyCallIdentifier is returned



5.2.2.3.4 Optional, invalid behaviour

Test MPCC_ IpCallLeg _20

Summary: IpCallLeg, getInfoReq, P_INVALID_SESSION_ID

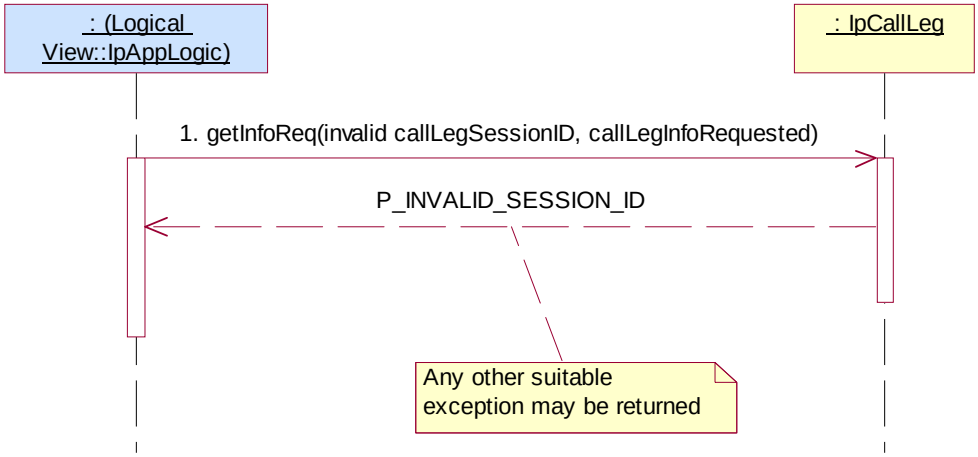
Reference: ES 201 915-4 [1], clause 7.3.5.

Preamble: Same as MPCC_ IpMultiPartyCall _14

Condition: createCallLeg and getInfoReq methods are supported.

Test Sequence:

1. Method call **getInfoReq()**
   Parameters: invalid callLegSessionID, valid callLegInfoRequested
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MPCC_ IpCallLeg _21

Summary: IpCallLeg, attachMediaReq, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.5.

Preamble: Same as MPCC_ IpMultiPartyCall _25

Condition: attachMediaReq method is supported.

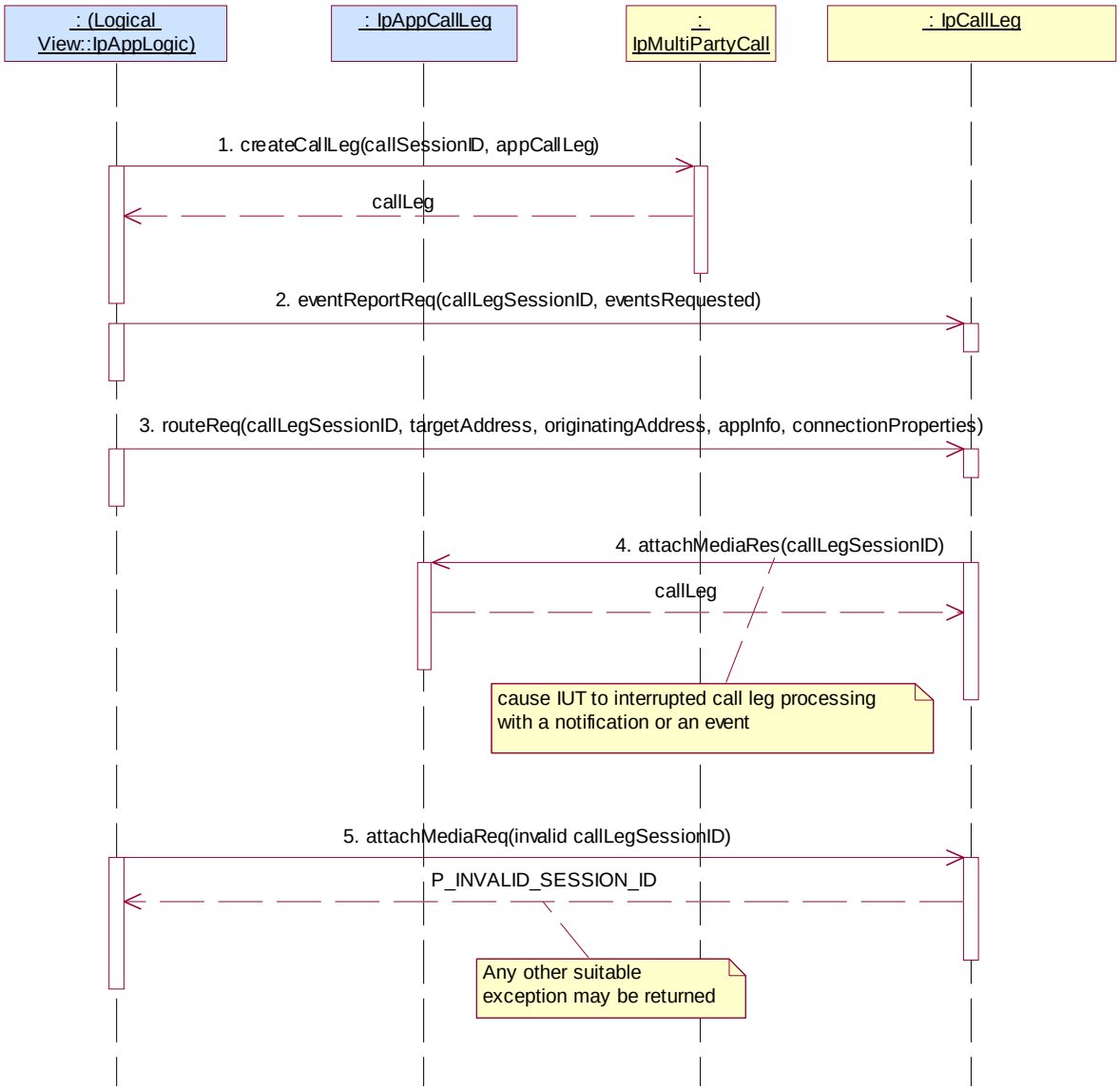
Test Sequence:

1. Method call **createCallLeg()**
   Parameters: valid callSessionID reported in preamble, valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

2. Method call **eventReportReq()**
   Parameters: valid callLegSessionID returned in 1, valid eventsRequested
   Check: no exception is returned

3. Method call **routeReq()**
   Parameters: valid callLegSessionID returned in 1, valid targetAddress, valid originatingAddress, valid appInfo, valid connectionProperties set to have implicit media management
   Check: no exception is returned

4. Method call **attachMediaReq()**
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned


: (Logical View::IpAppLogic)
: IpAppCallLeg
: IpMultiPartyCall
: IpCallLeg
1. createCallLeg(callSessionID, appCallLeg)
callLeg
2. eventReportReq(callLegSessionID, eventsRequested)
3. routeReq(callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties)
4. attachMediaRes(callLegSessionID)
callLeg
cause IUT to interrupted call leg processing with a notification or an event
5. attachMediaReq(invalid callLegSessionID)
P_INVALID_SESSION_ID
Any other suitable exception may be returned

Test MPCC_ IpCallLeg _22

Summary: IpCallLeg, detachMediaReq, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.5.

Preamble: Same as MPCC_ IpMultiPartyCall _25

Condition: attachMediaReq and detachMediaReq methods are supported.

Test Sequence:

1. Method call **createCallLeg()**
   Parameters: valid callSessionID reported in preamble, valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

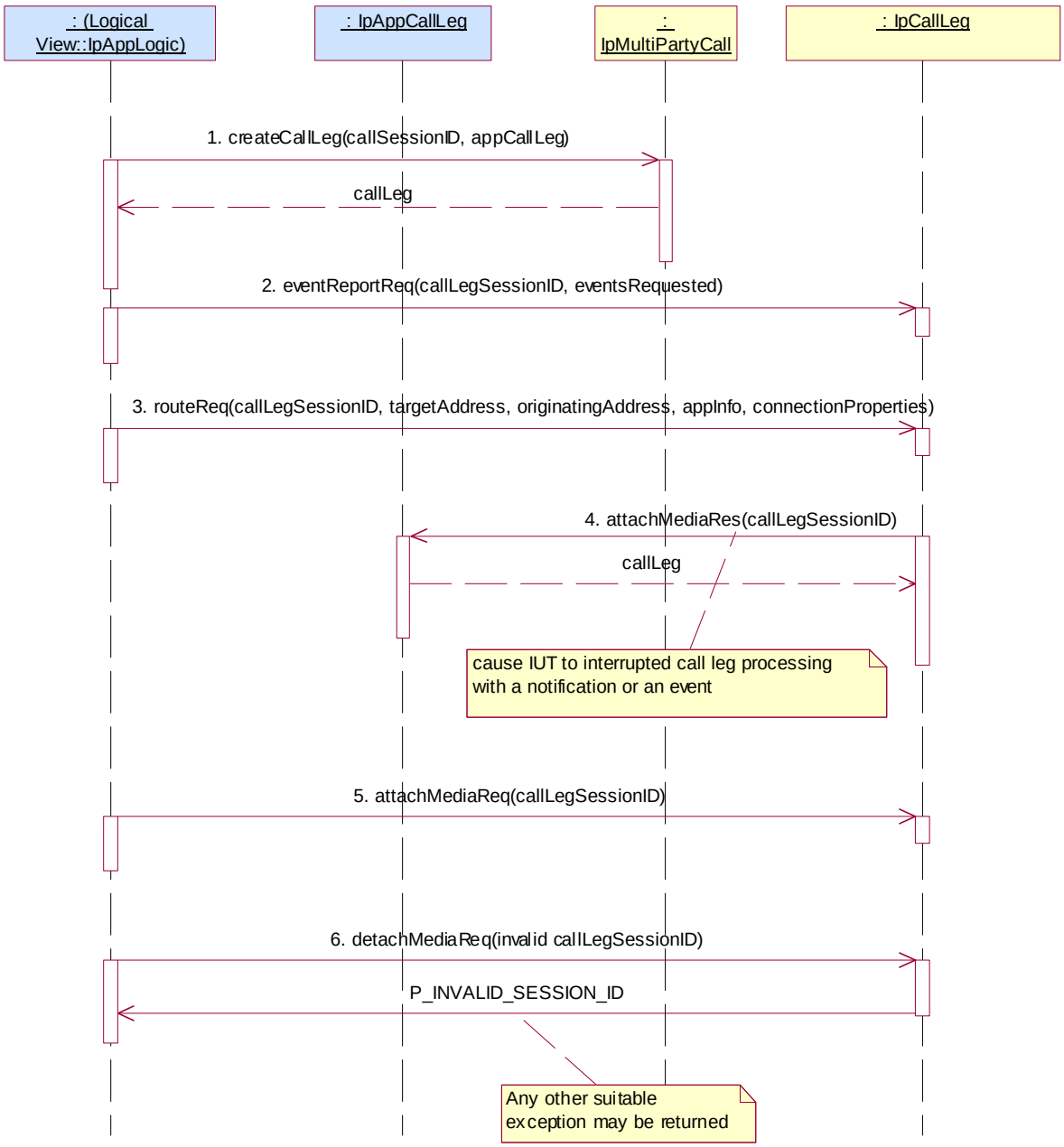
2. Method call **eventReportReq()**
   Parameters: valid callLegSessionID returned in 1, valid eventsRequested
   Check: no exception is returned

3. Method call **routeReq()**
   Parameters: valid callLegSessionID returned in 1, valid targetAddress, valid originatingAddress, valid appInfo, valid connectionProperties set to have explicit media management
   Check: no exception is returned

4. Method call **attachMediaReq()**
   Parameters: valid callLegSessionID returned in 1.
   Check: no exception is returned

5 Triggered action: cause IUT to call **attachMediaRes()** method on the tester's (Application) **IpAppCallLeg** interface.
   Parameters: callLegSessionID

6. Method call **detachMediaReq()**
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned


: (Logical View::IpAppLogic)
: IpAppCallLeg
: IpMultiPartyCall
: IpCallLeg
1. createCallLeg(callSessionID, appCallLeg)
callLeg
2. eventReportReq(callLegSessionID, eventsRequested)
3. routeReq(callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties)
4. attachMediaRes(callLegSessionID)
callLeg
cause IUT to interrupted call leg processing with a notification or an event
5. attachMediaReq(callLegSessionID)
6. detachMediaReq(invalid callLegSessionID)
P_INVALID_SESSION_ID
Any other suitable exception may be returned

Test MPCC_ IpCallLeg _23

Summary: IpCallLeg, getCurrentDestinationAddress, P_INVALID_SESSION_ID

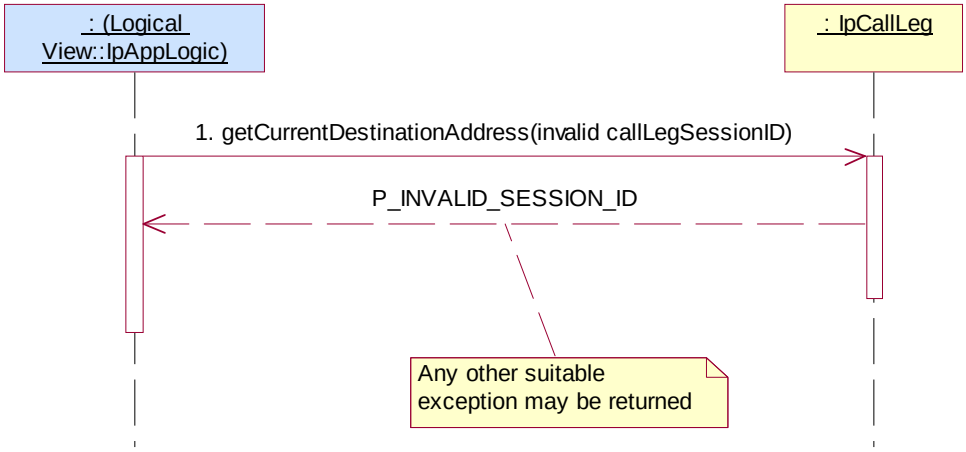
Reference: ES 201 915-4 [1], clause 7.3.5.

Preamble: Same as MPCC_ IpMultiPartyCall _03

Condition: getCurrentDestinationAddress method is supported.

Test Sequence:

1. Method call **getCurrentDestinationAddress()**
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MPCC_ IpCallLeg _24

Summary: IpCallLeg, setChargePlan, P_INVALID_SESSION_ID

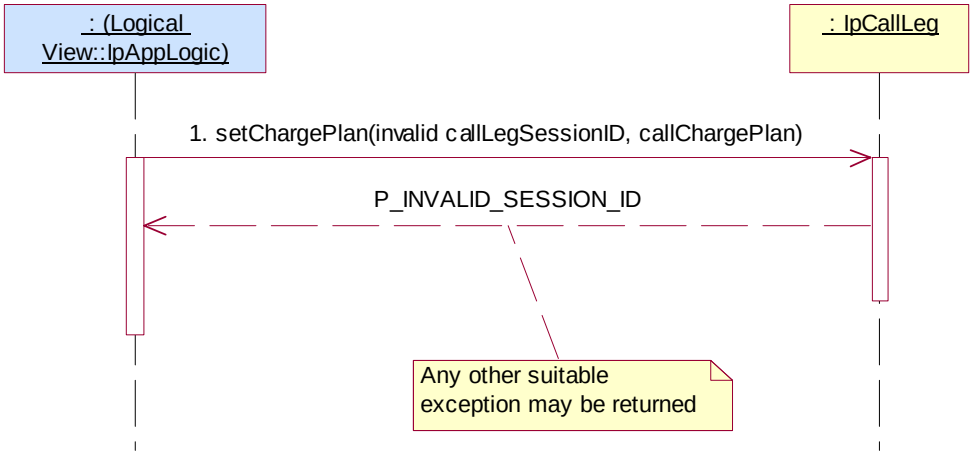
Reference: ES 201 915-4 [1], clause 7.3.5

Preamble: Same as MPCC_ IpMultiPartyCall _14

Condition: createCallLeg and setChargePlan methods are supported.

Test Sequence:

1. Method call **setChargePlan()**
   Parameters: invalid callLegSessionID, valid callChargePlan
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MPCC_ IpCallLeg _25

Summary: IpCallLeg, setAdviceOfCharge, P_INVALID_SESSION_ID

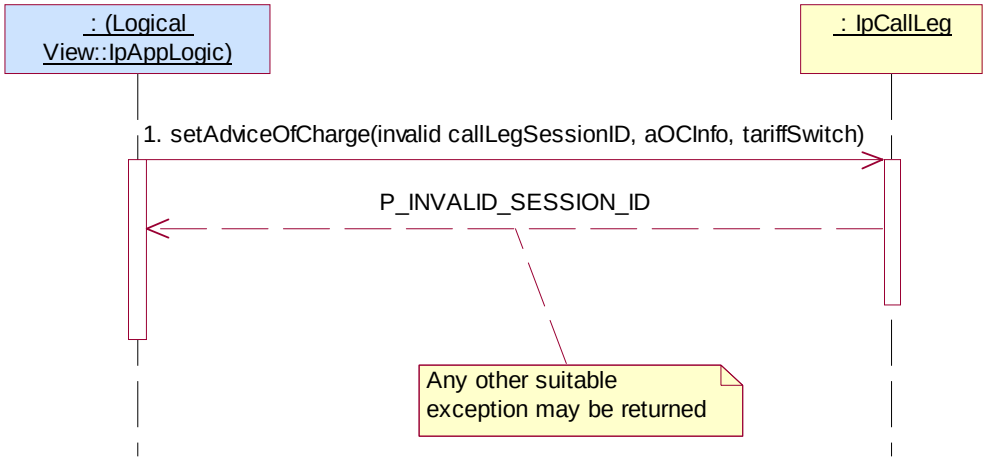
Reference: ES 201 915-4 [1], clause 7.3.5

Preamble: Same as MPCC_ IpMultiPartyCall _14

Condition: createCallLeg and setAdviceOfCharge methods are supported.

Test Sequence:

1. Method call **setAdviceOfCharge()**
   Parameters: invalid callLegSessionID, valid aOCInfo, valid tariffSwitch
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MPCC_ IpCallLeg _26

Summary: IpCallLeg, setAdviceOfCharge, P_INVALID_CURRENCY

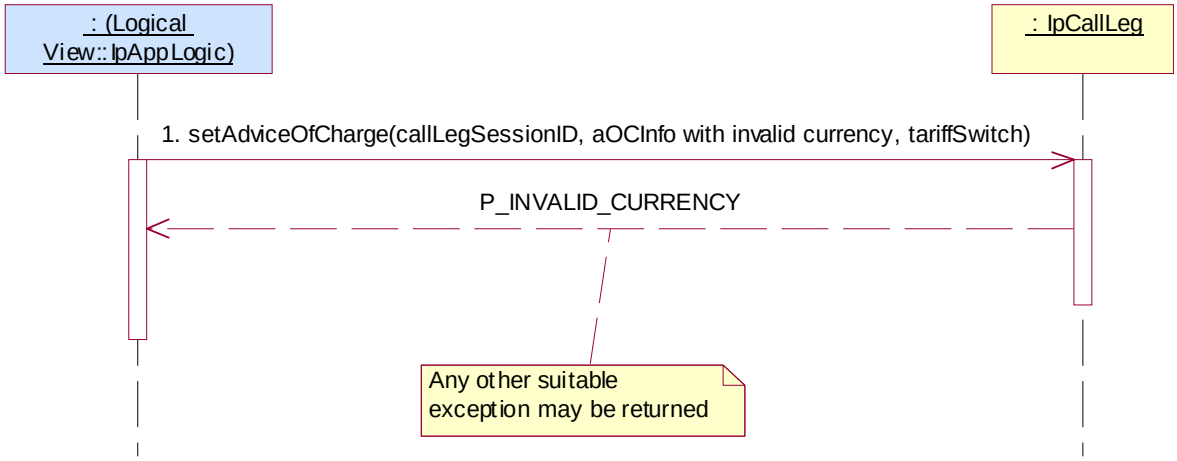
Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as MPCC_ IpMultiPartyCall _14

Condition: createCallLeg and setAdviceOfCharge methods are supported.

Test Sequence:

1. Method call **setAdviceOfCharge()**
   Parameters: valid callLegSessionID returned in preamble, aOCInfo with invalid currency, valid tariffSwitch
   Check: P_INVALID_CURRENCY, or another suitable exception, is returned



Test MPCC_ IpCallLeg _27

Summary: IpCallLeg, setAdviceOfCharge, P_INVALID_AMOUNT

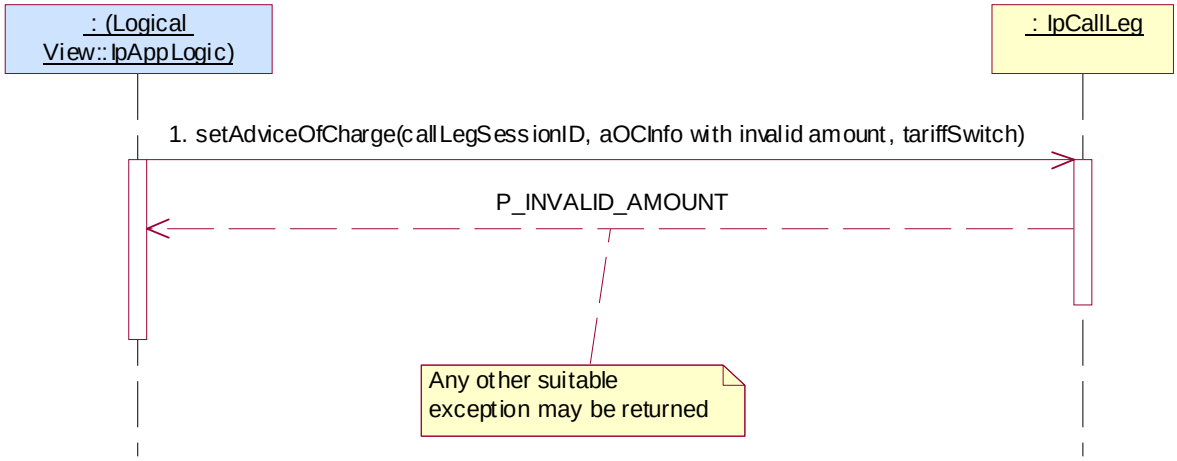
Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as MPCC_ IpMultiPartyCall _14

Condition: createCallLeg and setAdviceOfCharge methods are supported.

Test Sequence:

1. Method call **setAdviceOfCharge()**
   Parameters: valid callLegSessionID returned in preamble, aOCInfo with invalid amount, valid tariffSwitch
   Check: P_INVALID_AMOUNT, or another suitable exception, is returned



Test MPCC_ IpCallLeg _28

Summary: IpCallLeg, superviseReq, P_INVALID_SESSION_ID

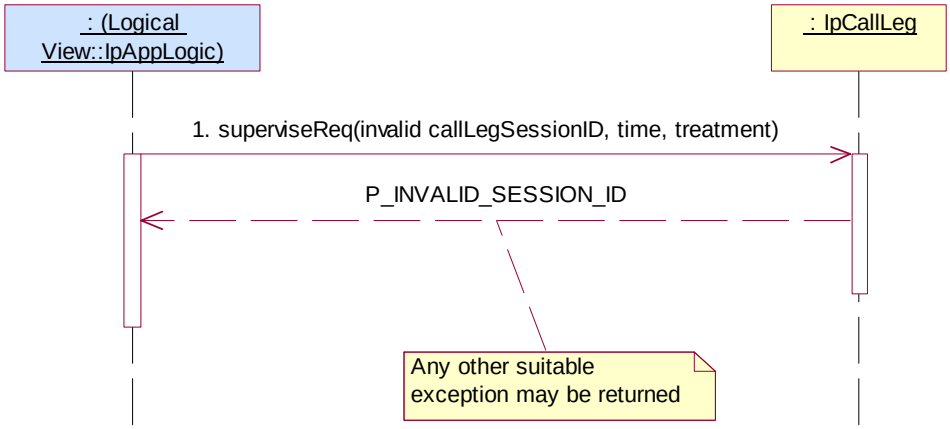
Reference: ES 201 915-4 [1], clause 7.3.5

Preamble: Same as MPCC_ IpMultiPartyCall _14

Condition: createCallLeg and superviseReq methods are supported.

Test Sequence:

1. Method call **superviseReq()**
   Parameters: invalid callLegSessionID, valid time, valid treatment
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MPCC_ IpCallLeg _29

Summary: IpCallLeg, getCall**,** P_INVALID_SESSION_ID

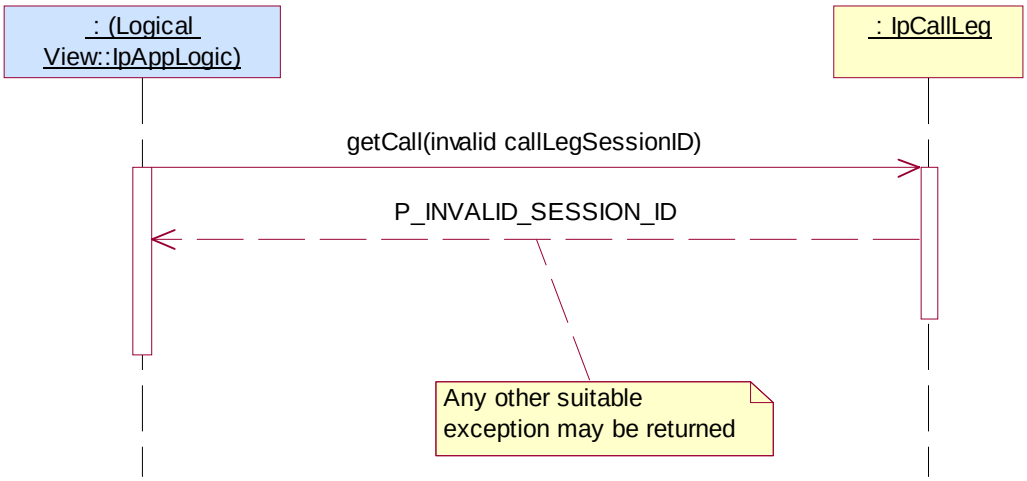
Reference: ES 201 915-4 [1], clause 7.3.5

Preamble: Same as MPCC_ IpMultiPartyCall _03

Condition: getCall method is supported.

Test Sequence:

1. Method call **getCall()**
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



5.2.3 MultiMedia Call Control Service (MMCC)

5.2.3.1 IpMultiMediaCallControlManager

5.2.3.1.1 Mandatory, valid behaviour

Test MMCC_ IpMultiMediaCallControlManager _01

Summary: IpMultiMediaCallControlManager , all mandatory methods, successful

Reference: ES 201 915-4 [1], clauses 7.3.1 and 8.3.1

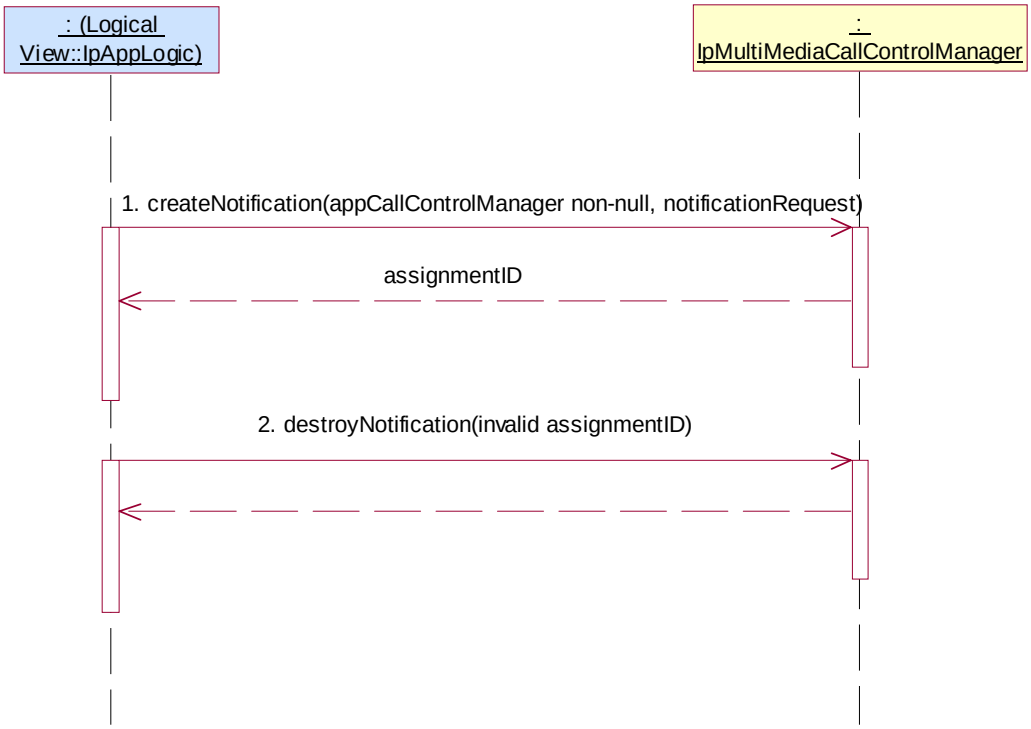
Condition: createNotification method is supported.

Test Sequence:

1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, non-null, value, valid notificationRequest
   Check: valid value of TpAssignmentID is returned

2. Triggered action: cause IUT to call Method **reportNotification()** method on the tester's (application) **IpAppMultiMediaCallControlManager** interface.
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID

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



Test MMCC_ IpMultiMediaCallControlManager _02

Summary: IpMultiMediaCallControlManager, all mandatory methods, successful

Reference: ES 201 915-4 [1], clauses 7.3.1 and 8.3.1

Condition: createNotification method is supported.

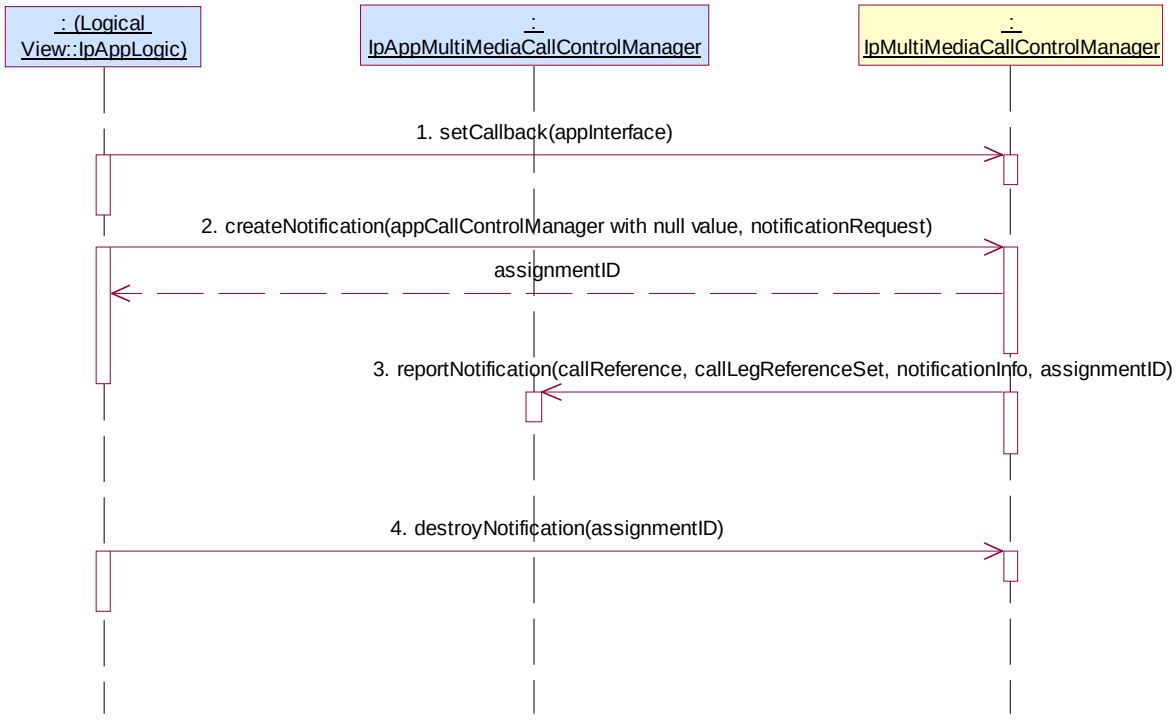
Test Sequence:

1. Method call **setCallback()** on IpCallControlManager
   Parameters: valid, non-null, value of appInterface parameter
   Check: no exception is returned

2. Method call **createNotification()**
   Parameters: appCallControlManager with null, value, valid notificationRequest
   Check: valid value of TpAssignmentID is returned

3. Triggered action: cause IUT to call Method **reportNotification()** method on the tester's (application) **IpAppMultiMediaCallControlManager** interface.
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID

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



Test MMCC_ IpMultiMediaCallControlManager _03

Summary: IpMultiMediaCallControlManager, all mandatory methods, successful

Reference: ES 201 915-4 [1], clauses 7.3.1 and 8.3.1

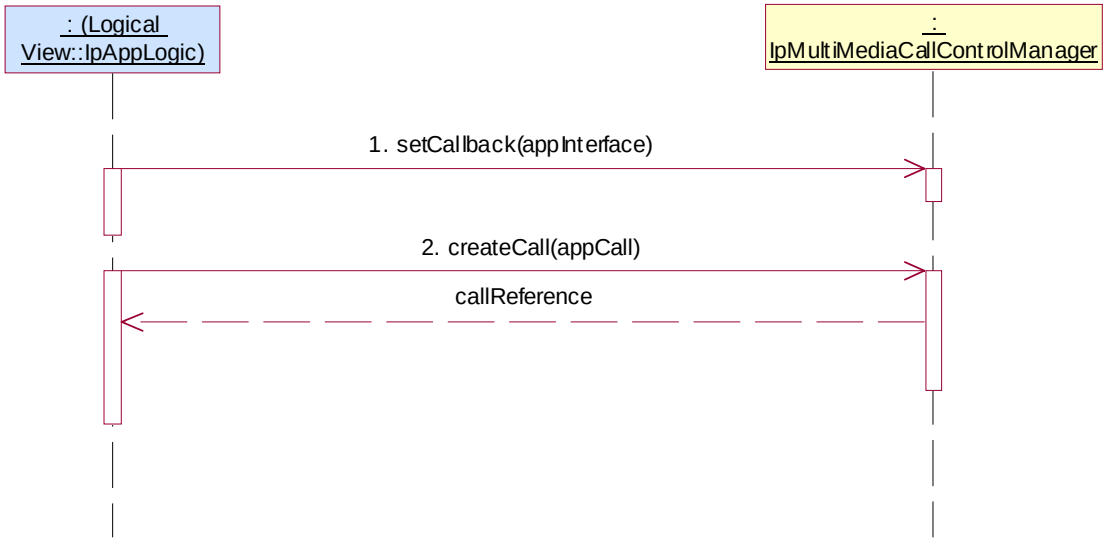
Preamble: Application has a reference interface used for callbacks.

Condition: createCall method is supported.

Test Sequence:

1. Method call **setCallback()** on IpMultiMediaCallControlManager
   Parameters: valid, non-null, value of appInterface parameter
   Check: no exception is returned

2. Method call **createCall()**
   Parameters: valid appCall
   Check: valid value of TpMultiMediaCallIdentifier is returned



Test MMCC_ IpMultiMediaCallControlManager _04

Summary: IpMultiMediaCallControlManager, all methods mandatory, successful

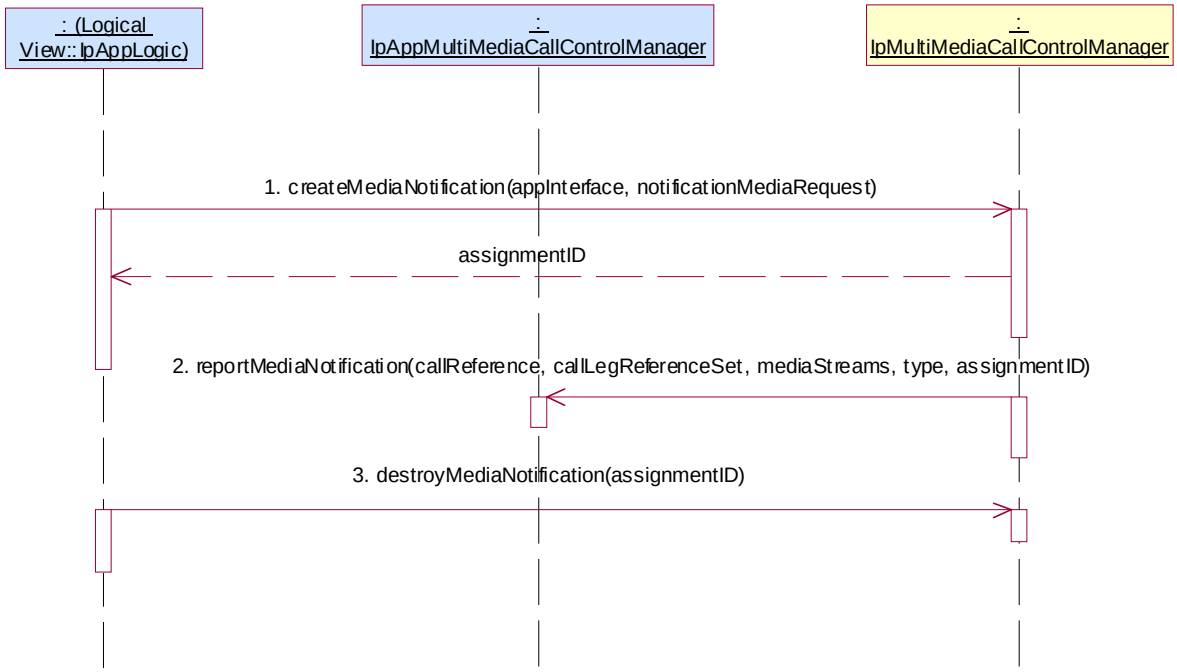
Reference: ES 201 915-4 [1], clause 8.3.1

Test Sequence:

1. Method call **createMediaNotification()**
   Parameters: valid appInterface, valid notificationMediaRequest
   Check: valid value of TpAssignmentID is returned

2. Triggered action: cause IUT to call Method **reportMediaNotification()** method on the tester's (application) **IpAppMultiMediaCallControlManager** interface.
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID

3. Method call **destroyMediaNotification()**
   Parameters: valid assignmentID returned in 1.
   Check: no exception is returned



5.2.3.1.2 Mandatory, invalid behaviour

Test MMCC_ IpMultiMediaCallControlManager _05

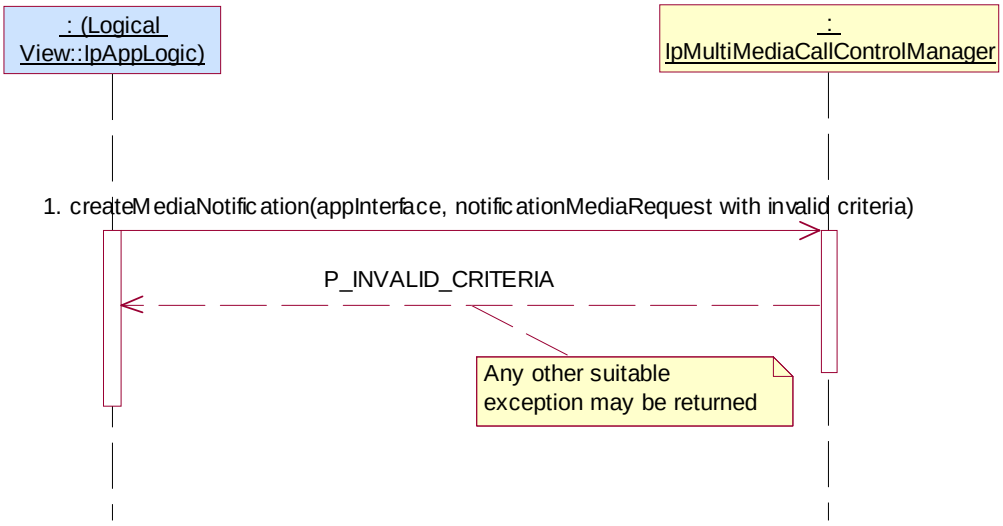
Summary: IpMultiMediaCallControlManager, createMediaNotification, P_INVALID_CRITERIA

Reference: ES 201 915-4 [1], clause 8.3.1

Preamble: Application has a reference interface used for callbacks.

Test Sequence:

1. Method call **createMediaNotification()**
   Parameters: valid appInterface, valid notificationMediaRequest with invalid criteria
   Check: P_INVALID_CRITERIA, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCallControlManager _06

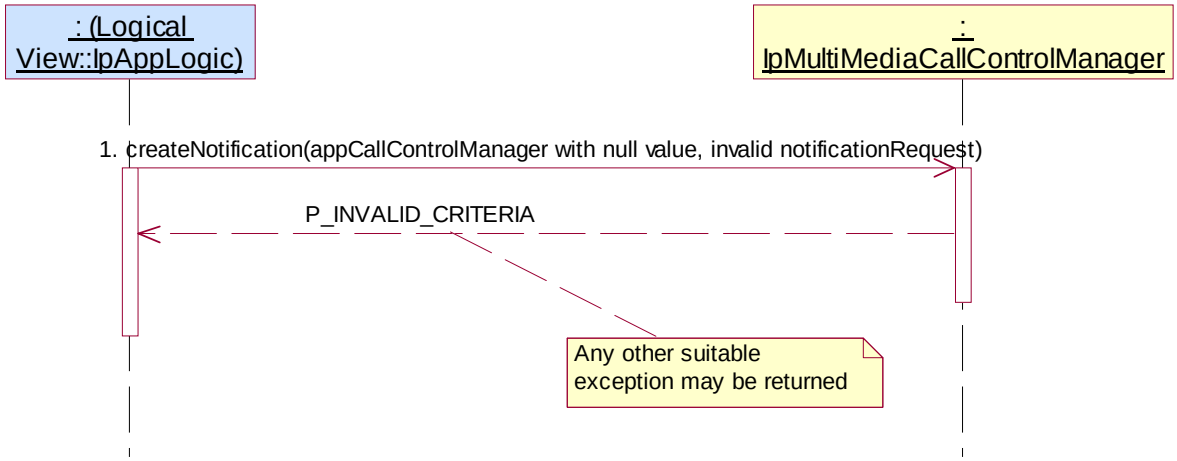
Summary: IpMultiMediaCallControlManager, createNotification, P_INVALID_CRITERIA

Reference: ES 201 915-4 [1], clause 7.3.1

Condition: createNotification method is supported.

Test Sequence:

1. Method call **createNotification()**
   Parameters: appCallControlManager with null value, invalid notificationRequest
   Check: P_INVALID_CRITERIA, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCallControlManager _07

Summary: IpMultiMediaCallControlManager, destroyNotification, P_INVALID_ASSIGNMENT_ID

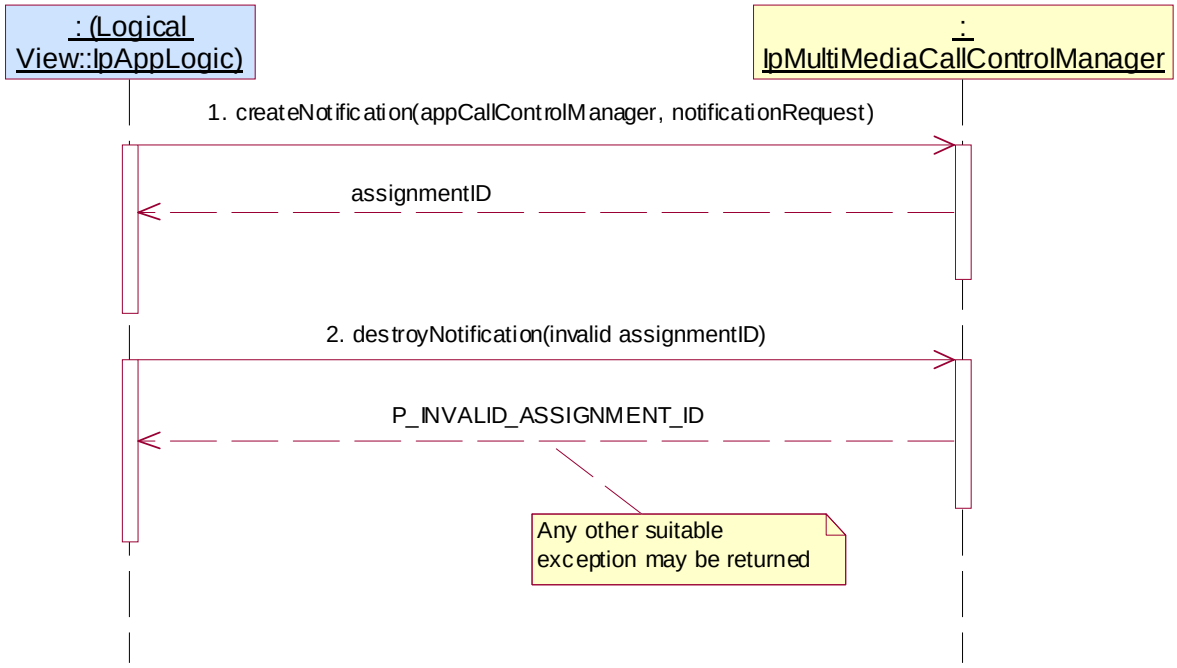
Reference: ES 201 915-4 [1], clause 7.3.1

Condition: createNotification, destroyNotification methods are supported.

Test Sequence:

1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, non-null, value, valid notificationRequest
   Check: valid value of TpAssignmentID is returned

2. Method call **destroyNotification**()
   Parameters: invalid assignmentID
   Check: P_INVALID_ASSIGNMENT_ID, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCallControlManager _08

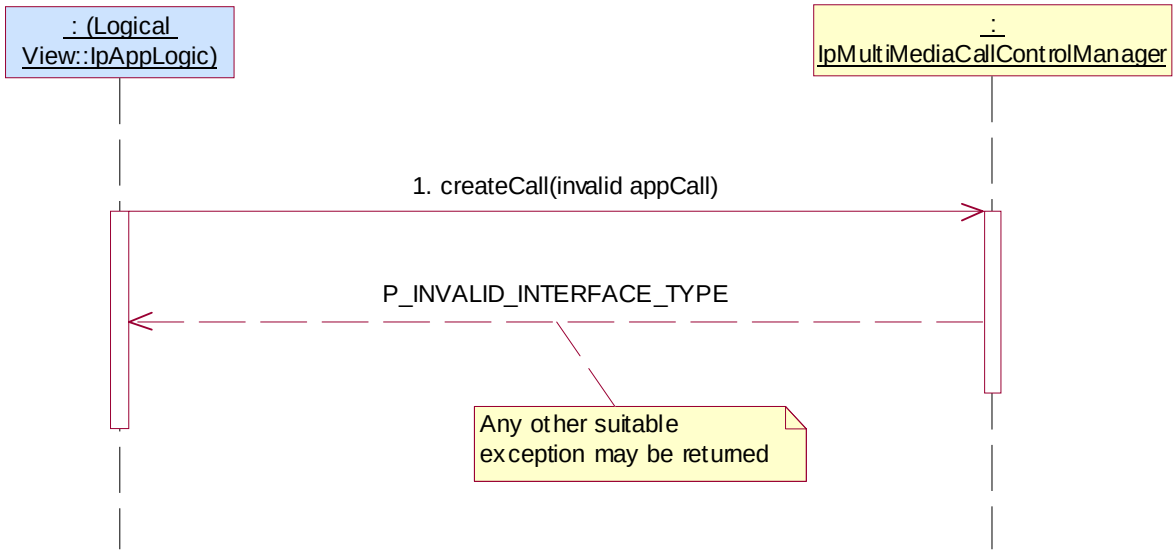
Summary: IpMultiMediaCallControlManager, createCall , P_INVALID_INTERFACE_TYPE

Reference: ES 201 915-4 [1], clause 7.3.1

Condition: createCall method is supported.

Test Sequence:

1. Method call **createCall()**
   Parameters: invalid value of appCall
   Check: P_INVALID_INTERFACE_TYPE, or another suitable exception, is returned



5.2.3.1.3 Optional, valid behaviour

Test MMCC_ IpMultiMediaCallControlManager _09

Summary: IpMultiMediaCallControlManager, getNotification, successful

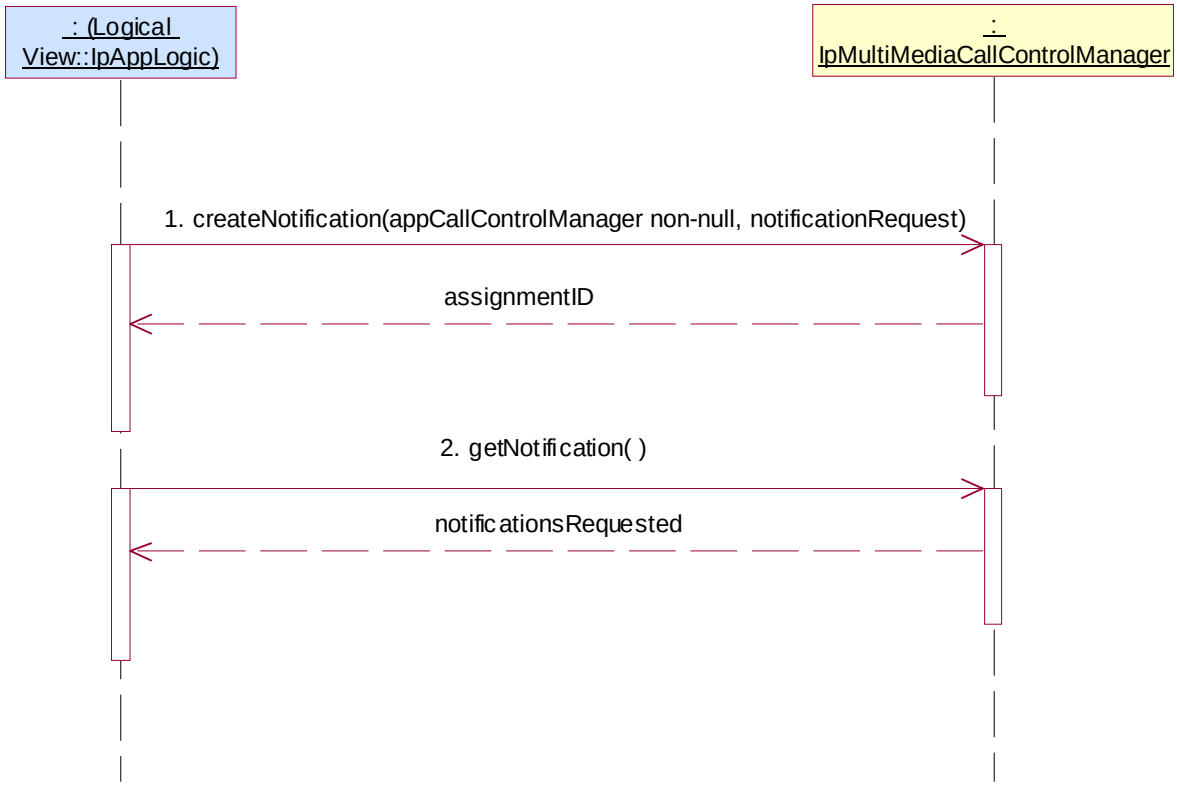
Reference: ES 201 915-4 [1], clause 7.3.1

Condition: createNotification and getNotification methods are supported.

Test Sequence:

1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, non-null, value, valid notificationRequest
   Check: valid value of TpAssignmentID is returned

2. Method call **getNotification()**
   Parameters: None
   Check: valid value of TpNotificationRequestedSet is returned where notificationRequest given in 1. is included as a value of this TpCallEventCriteriaResult



Test MMCC_ IpMultiMediaCallControlManager _10

Summary: IpMultiMediaCallControlManager, getMediaNotification, successful

Reference: ES 201 915-4 [1], clause 8.3.1

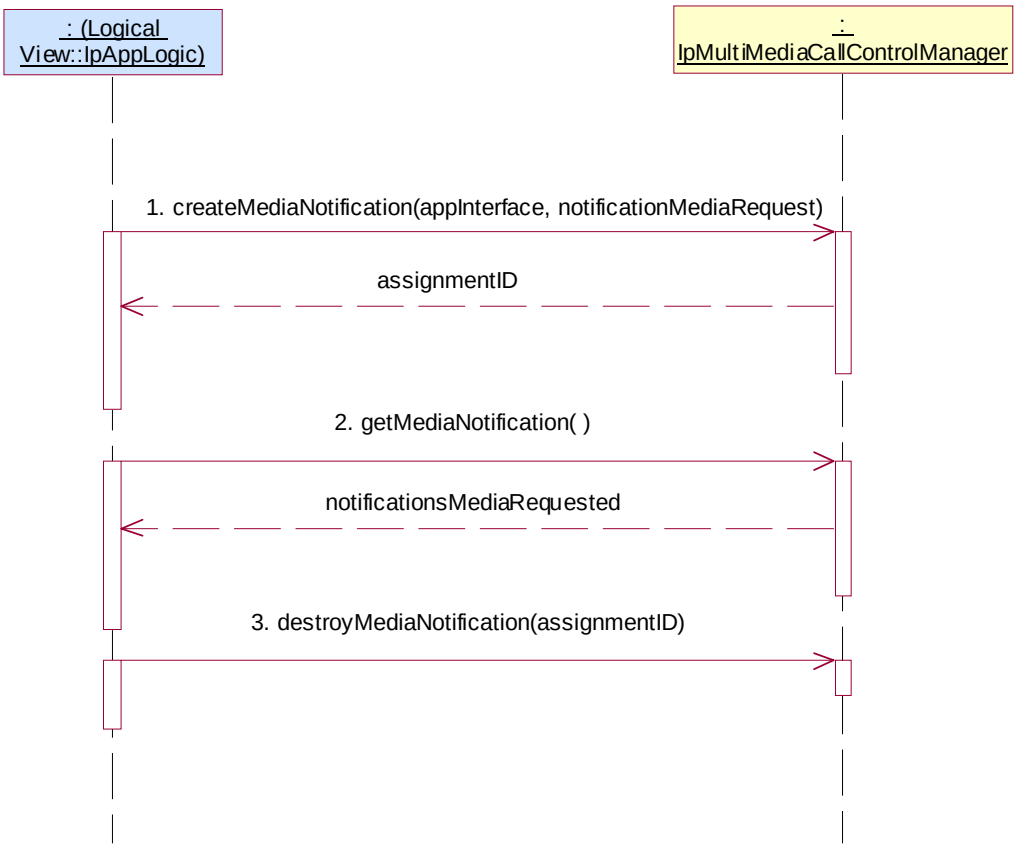
Condition: getMediaNotification method is supported.

Test Sequence:

1. Method call **createMediaNotification()**
   Parameters: valid appInterface, valid notificationMediaRequest
   Check: valid value of TpAssignmentID is returned

2. Method call **getMediaNotification()**
   Parameters: None
   Check: valid value of TpMediaNotificatioRequestedSet is returned with values of notificationMediaRequest given in 1.

3. Method call **destroyMediaNotification()**
   Parameters: valid assignmentID returned in 1.
   Check: no exception is returned



Test MMCC_ IpMultiMediaCallControlManager _11

Summary: IpMultiMediaCallControlManager, changeMediaNotification, successful

Reference: ES 201 915-4 [1], clause 8.3.1

Condition: getMediaNotification and changeMediaNotification methods are supported.

Test Sequence:

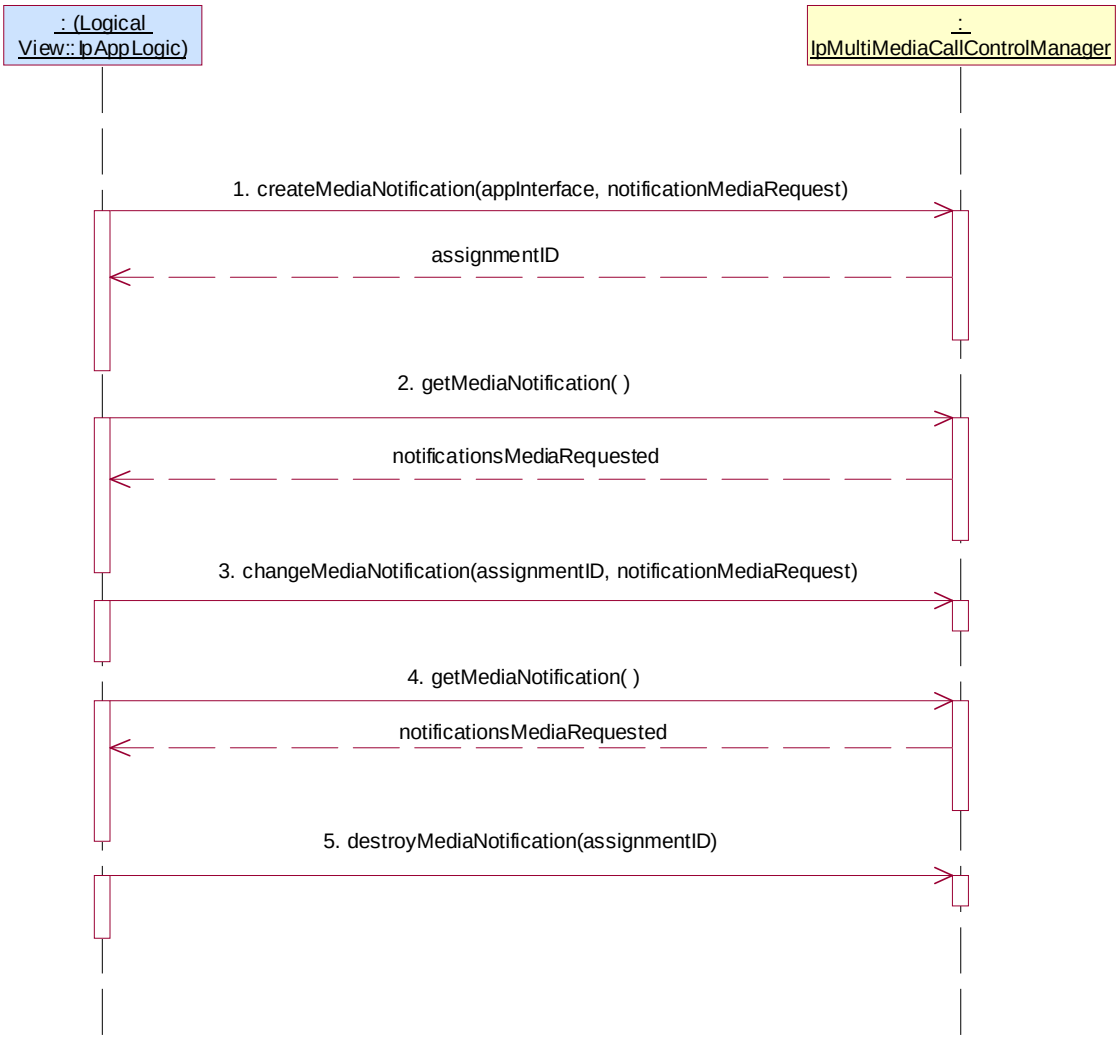
1. Method call **createMediaNotification()**
   Parameters: valid appInterface, valid notificationMediaRequest
   Check: valid value of TpAssignmentID is returned

2. Method call **getMediaNotification()**
   Parameters: None
   Check: valid value of TpMediaNotificatioRequestedSet is returned with values of notificationMediaRequest given in 1.

3. Method call **changeMediaNotification()**
   Parameters: valid assignmentID returned in 1., valid notificationMediaRequest with different values from notificationMediaRequest given in 1.
   Check: no exception is returned

4. Method call **getMediaNotification()**
   Parameters: None
   Check: valid value of TpMediaNotificatioRequestedSet is returned with values of notificationMediaRequest given in 3.

5. Method call **destroyMediaNotification()**
   Parameters: valid assignmentID returned in 1.
   Check: no exception is returned


: (Logical View:: IpAppLogic)
: IpMultiMediaCallControlManager
1. createMediaNotification(appInterface, notificationMediaRequest)
assignmentID
2. getMediaNotification( )
notificationsMediaRequested
3. changeMediaNotification(assignmentID, notificationMediaRequest)
4. getMediaNotification( )
notificationsMediaRequested
5. destroyMediaNotification(assignmentID)

Test MMCC_ IpMultiMediaCallControlManager _12

Summary: IpMultiMediaCallControlManager, changeNotification, successful

Reference: ES 201 915-4 [1], clause 7.3.1

Condition: createNotification, getNotification and changeNotification methods are supported.

Test Sequence:

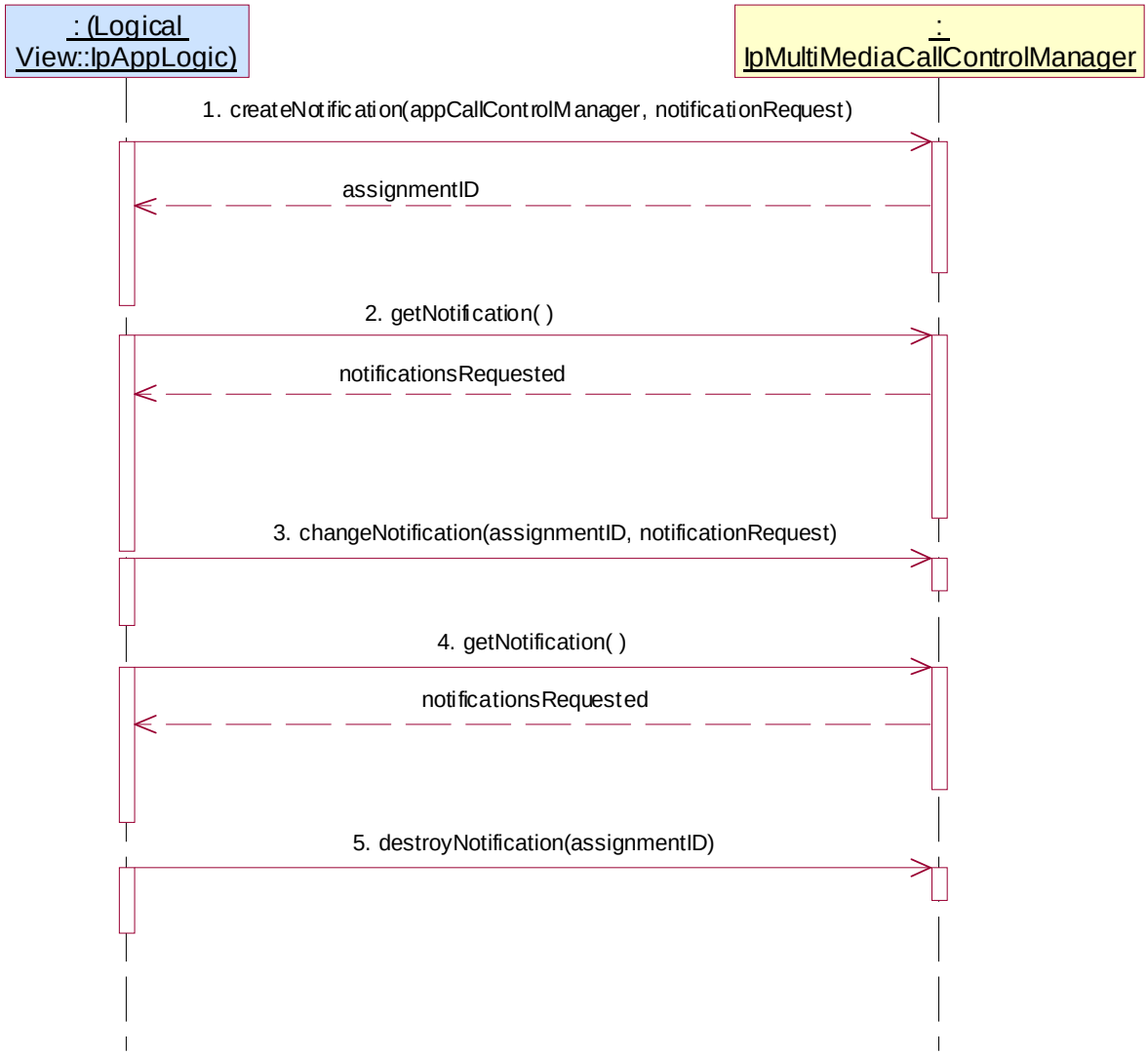
1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, non-null, value, valid notificationRequest
   Check: valid value of TpAssignmentID is returned

2. Method call **getNotification()**
   Parameters: None
   Check: valid value of TpNotificationRequestedSet is returned where notificationRequest given in 1. is included as a value of this TpCallEventCriteriaResult

3. Method call **changeNotification**()
   Parameters: assignmentID returned in 1., valid notificationRequest different from this given in 1.
   Check: no exception is returned

4. Method call **getNotification()**
   Parameters: None
   Check: valid value of TpNotificationRequestedSet is returned where notificationRequest given in 1. is included as a value of this TpCallEventCriteriaResult

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


: (Logical View::IpAppLogic)
: IpMultiMediaCallControlManager
1. createNotification(appCallControlManager, notificationRequest)
assignmentID
2. getNotification( )
notificationsRequested
3. changeNotification(assignmentID, notificationRequest)
4. getNotification( )
notificationsRequested
5. destroyNotification(assignmentID)

Test MMCC_ IpMultiMediaCallControlManager _13

Summary: IpMultiMediaCallControlManager, all methods, successful

Reference: ES 201 915-4 [1], clauses 7.3.1 and 8.3.1

Condition: createNotification, callOverLoadEncountered and callOverLoadCeased methods are supported.

Test Sequence:

1. Method call createNotification()
   Parameters: appCallControlManager with valid, non-null, value, valid notificationRequest
   Check: valid value of TpAssignmentID is returned

2. Method call setCallLoadControl()
   Parameters: valid duration, valid mechanism, valid treatment, valid addressRange
   Check: valid value of TpAssignmentID is returned

3. Triggered action: cause IUT to call reportNotification() numerous times during the following sequence, and attempt to provoke an overload condition and then remove it.

4. Triggered action: cause IUT to call callOverLoadEncountered() method on the tester's (Application) IpAppMultiMediaCallControlManager interface.
   Parameters: valid assignmentID returned in 2.

5. Triggered action: cause IUT to call **callOverLoadCeased()** method on the tester's (Application) **IpAppMultiMediaCallControlManager** interface.
   Parameters: valid assignmentID returned in 2.


: (Logical View::IpAppLogic)
: IpAppMultiMediaCallControlManager
: IpMultiMediaCallControlManager
1. createNotification(appCallControlManager, notificationRequest)
assignmentID
2. setCallLoadControl(duration, mechanism, treatment, addressRange)
assignmentID
3. reportNotification(callReference, callLegReferenceSet, notificationInfo, assignmentID)
appCallBack
4. callOverloadEncountered(assignmentID)
5. callOverloadCeased(assignmentID)

Test MMCC_ IpMultiMediaCallControlManager _14

Summary: IpMultiMediaCallControlManager, all methods, successful

Reference: ES 201 915-4 [1], clause 7.3.1 and 8.3.1

Condition: createNotification, setCallLoadControl methods are supported.

Test Sequence:

1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, not null, value, valid eventCriteria
   Check: valid value of TpAssignmentID is returned

2. Method call **setCallLoadControl()**
   Parameters: valid duration, valid mechanism, valid treatment, valid addressRange
   Check: valid value of TpAssignmentID is returned

3. Triggered action: cause IUT to call **reportNotification**()method on the tester's (Application) **IpAppMultiMediaCallControlManager** interface.

4 Triggered action: cause IUT to call **managerInterrupted()** method on the tester's (Application) **IpAppMultiMediaCallControlManager** interface.
   Parameters: None

5 Triggered action: cause IUT to call **managerResumed()** method on the tester's (Application) **IpAppMultiMediaCallControlManager** interface.
   Parameters: None

Test MMCC_ IpMultiMediaCallControlManager _15

Summary: IpMultiMediaCallControlManager, all methods, successful

Reference: ES 201 915-4 [1], clauses 7.3.1 and 8.3.1

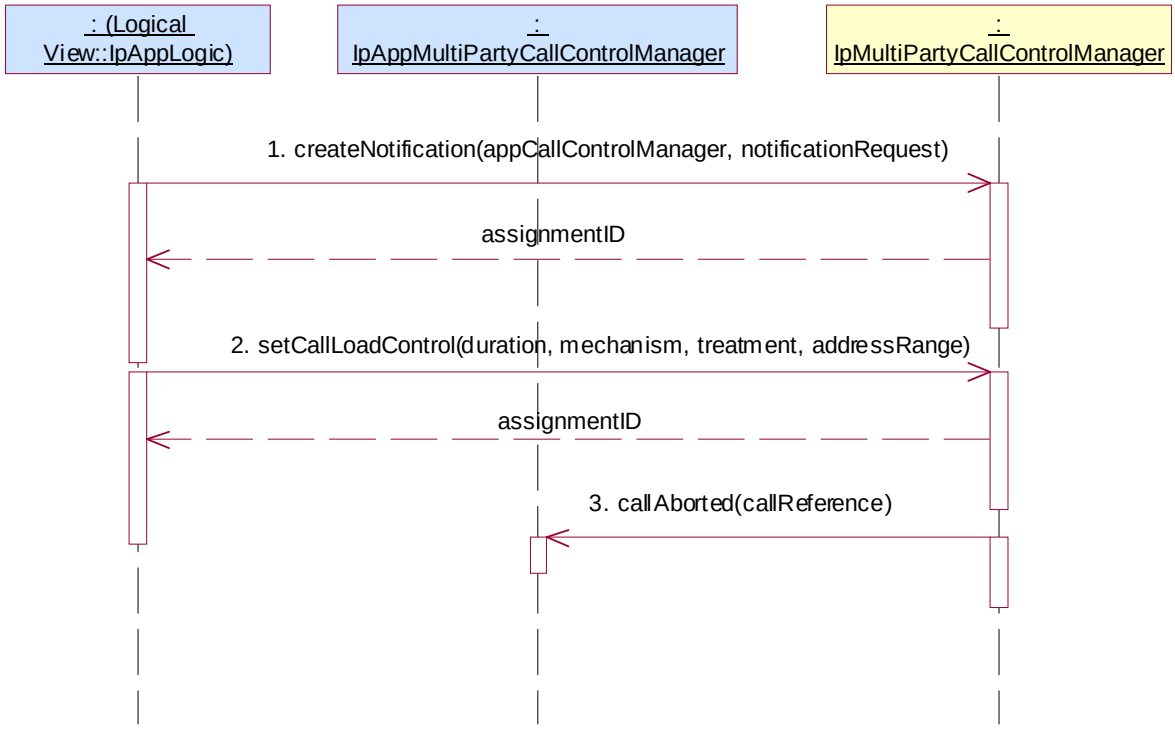
Condition: createNotificationand callAborted methods are supported.

Test Sequence:

1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, not null, value, valid eventCriteria
   Check: valid value of TpAssignmentID is returned

2. Triggered action: cause IUT to call **reportNotification**() method on the tester's (Application) **IpAppMultiMediaCallControlManager** interface.

3 Triggered action: cause IUT to call **callAborted()** method on the tester's (Application) **IpAppMultiMediaCallControlManager** interface.
   Parameters: valid assignmentID as reported in reportNotification.



5.2.3.1.4 Optional, invalid behaviour

Test MMCC_ IpMultiMediaCallControlManager _16

Summary: IpMultiMediaCallControlManager, changeMediaNotification, P_INVALID_CRITERIA

Reference: ES 201 915-4 [1], clause 8.3.1

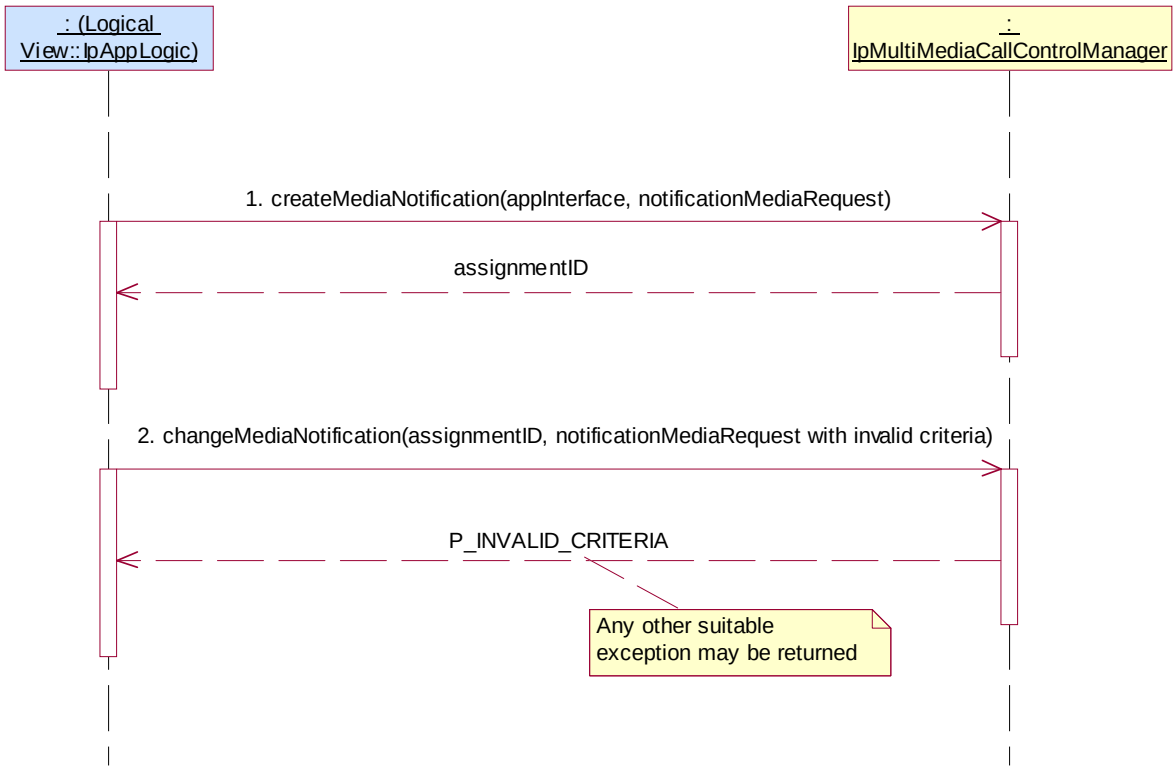
Preamble: Application has a reference interface used for callbacks.

Condition: changeMediaNotification method is supported.

Test Sequence:

1. Method call **createMediaNotification()**
   Parameters: valid appInterface, valid notificationMediaRequest
   Check: valid value of TpAssignmentID is returned

2. Method call **changeMediaNotification()**
   Parameters: valid assignmentID returned in 1., valid notificationMediaRequest with invalid criteria
   Check: P_INVALID_CRITERIA, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCallControlManager _17

Summary: IpMultiMediaCallControlManager, changeMediaNotification, P_INVALID_ASSIGNMENT_ID

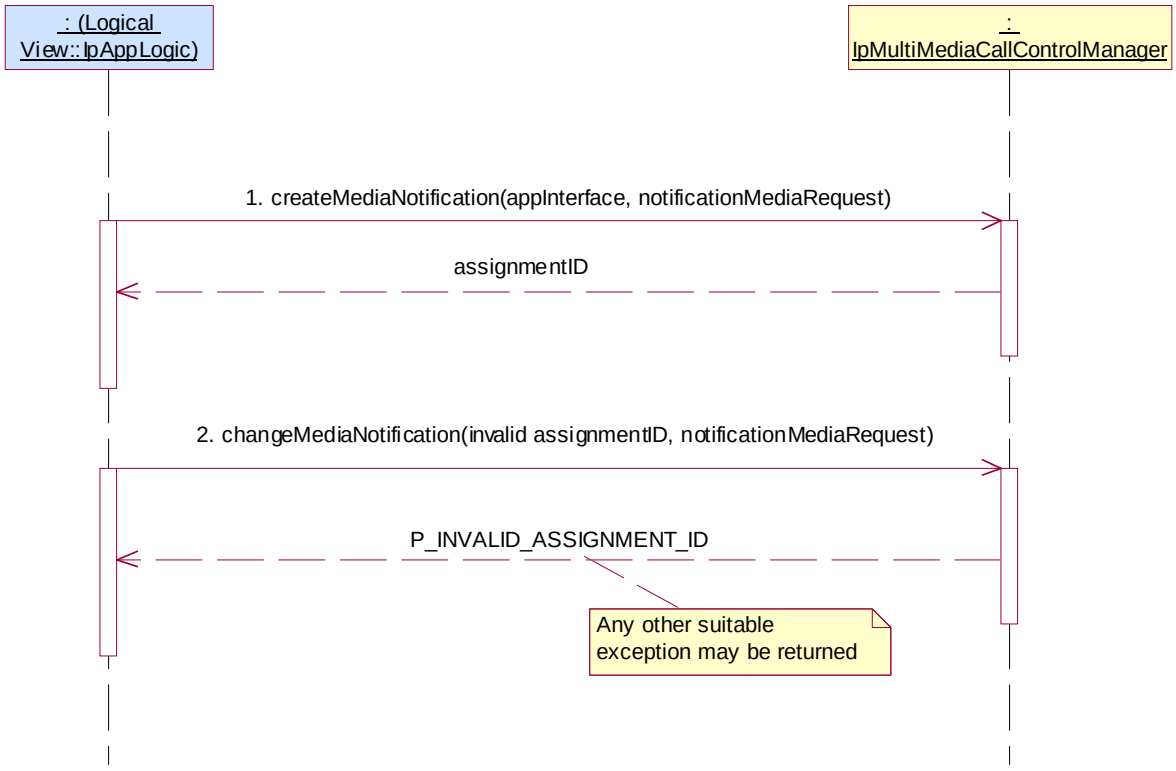
Reference: ES 201 915-4 [1], clause 8.3.1

Condition: createNotification, changeMediaNotification methods are supported.

Test Sequence:

1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, non-null, value, valid notificationRequest
   Check: valid value of TpAssignmentID is returned

2. Method call **changeMediaNotification()**
   Parameters: invalid assignmentID, valid notificationMediaRequest
   Check: P_INVALID_ASSIGNMENT_ID, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCallControlManager _18

Summary: IpMultiMediaCallControlManager, changeNotification, P_INVALID_ASSIGNMENT_ID

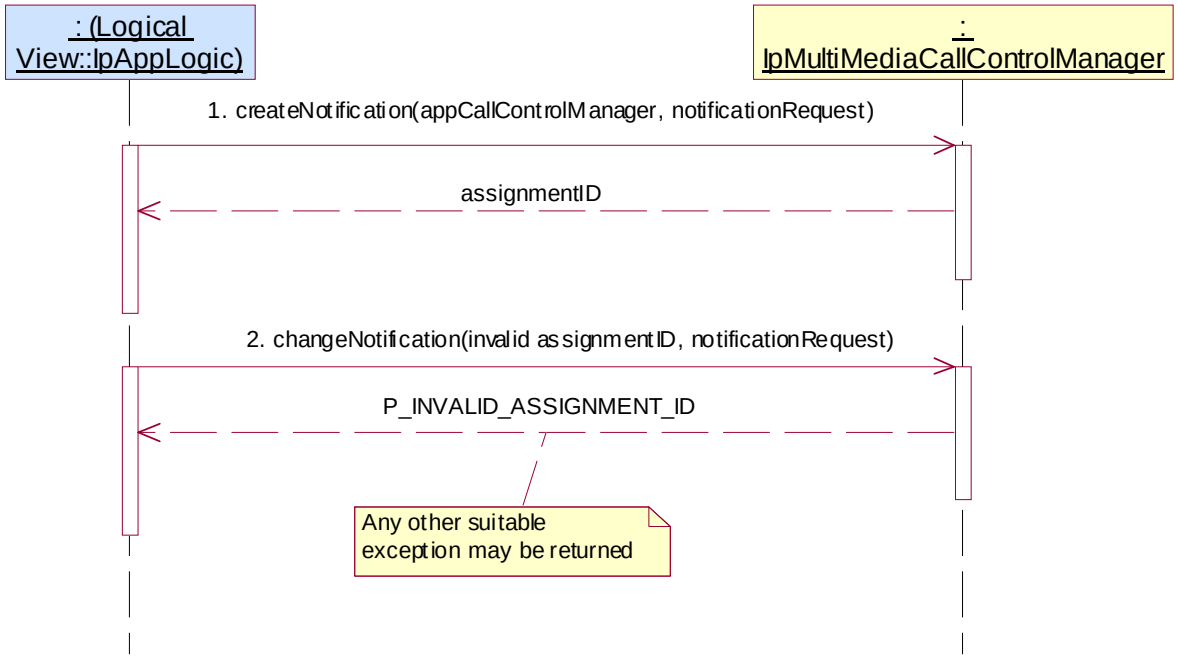
Reference: ES 201 915-4 [1], clause 7.3.1

Condition: createNotification, changeNotification methods are supported.

Test Sequence:

1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, non-null, value, valid notificationRequest
   Check: valid value of TpAssignmentID is returned

2. Method call **changeNotification**()
   Parameters: invalid assignmentID, valid notificationRequest
   Check: P_INVALID_ASSIGNMENT_ID, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCallControlManager _19

Summary: IpMultiMediaCallControlManager, changeNotification, P_INVALID_CRITERIA

Reference: ES 201 915-4 [1], clause 7.3.1

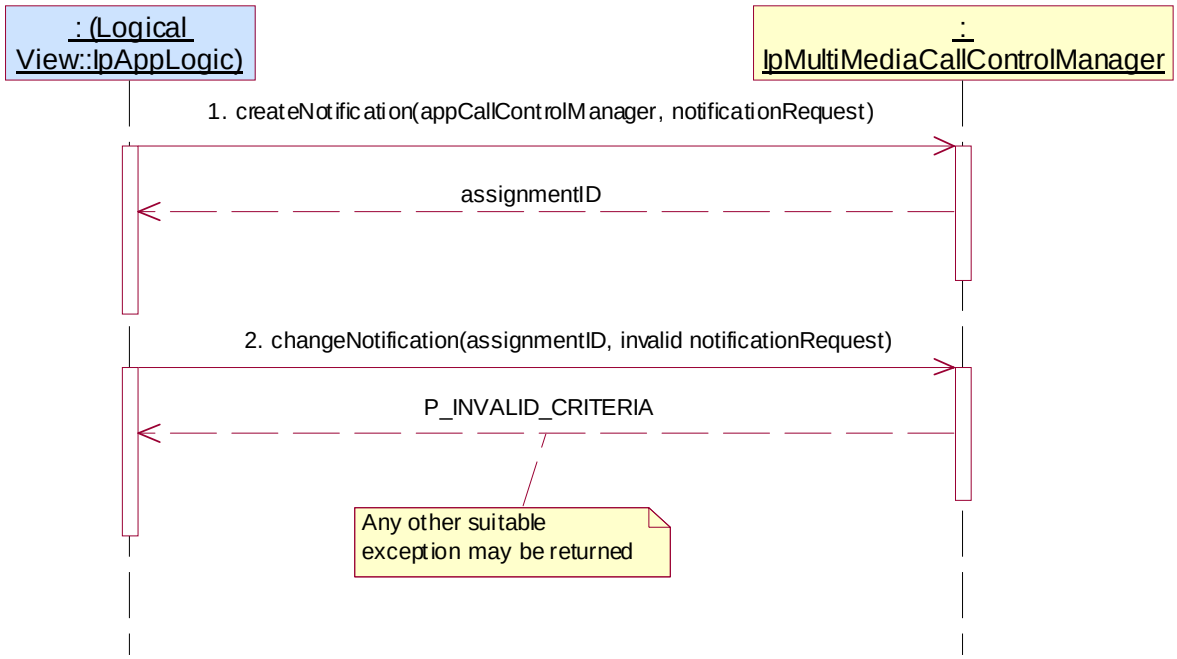
Preamble: Application has a reference interface used for callbacks.

Condition: createNotification and changeNotification methods are supported.

Test Sequence:

1. Method call **createNotification()**
   Parameters: appCallControlManager with null value, valid notificationRequest
   Check: valid value of TpAssignmentID is returned

2. Method call **changeNotification**()
   Parameters: assignmentID returned in 1., invalid notificationRequest
   Check: P_INVALID_CRITERIA, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCallControlManager _20

Summary: IpMultiMediaCallControlManager, setCallLoadControl, , P_INVALID_ADDRESS

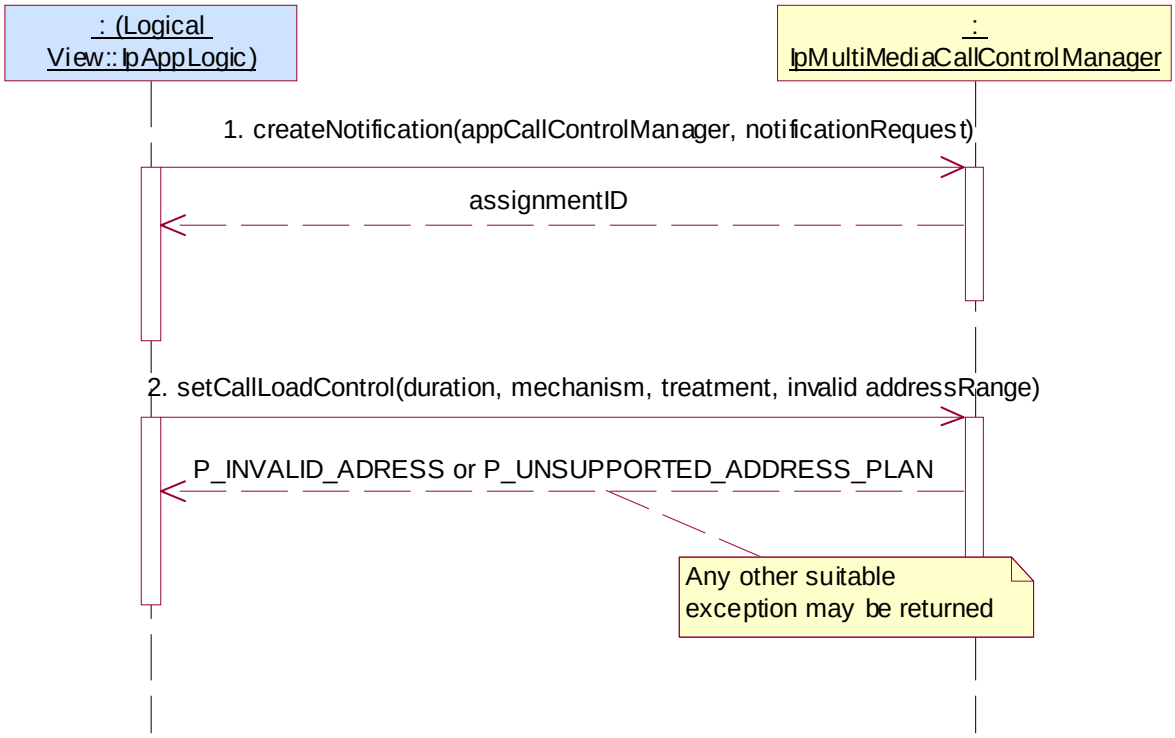
Reference: ES 201 915-4 [1], clause 7.3.1

Condition: createNotification and setCallLoadControl are supported.

Test Sequence:

1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, not null, value, valid eventCriteria
   Check: valid value of TpAssignmentID is returned

2. Method call **setCallLoadControl()**
   Parameters: valid duration, valid mechanism, valid treatment, invalid addressRange
   Check: P_INVALID_ADDRESS, P_UNSUPPORTED_ADDRESS_PLAN or another suitable exception, is returned



5.2.3.2 IpMultimediaCall

5.2.3.2.1 Mandatory, valid behaviour

According Call Control SCF specification, at least one of the two following test sequence is mandatory.

Test MMCC_ IpMultiMediaCall _01

Summary: IpMultiMediaCall, all methods mandatory, successful

Reference: ES 201 915-4 [1], clause 7.3.1 and 7.3.3.

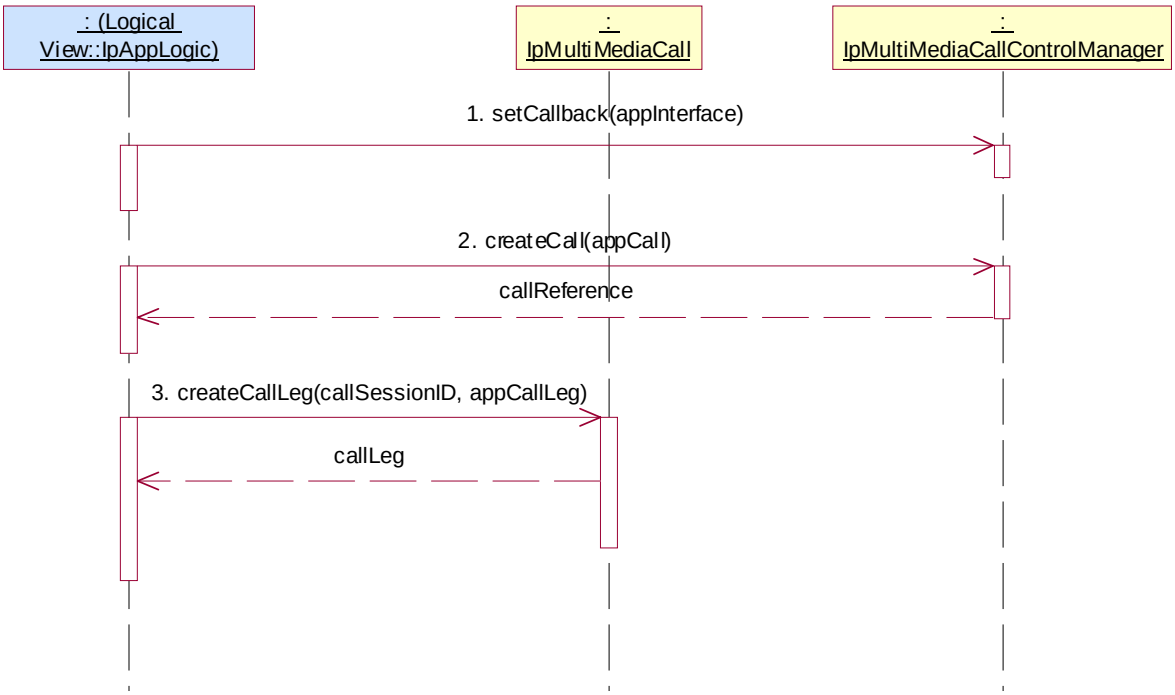
Condition: createCall, createCallLeg methods are supported.

Test Sequence:

1. Method call **setCallback()** on IpMultiMediaCallControlManager
   Parameters: valid, non-null, value of appInterface parameter
   Check: no exception is returned

2. Method call **createCall()**
   Parameters: valid appCall
   Check: valid value of TpMultiMediaCallIdentifier is returned

3. Method call **createCallLeg()** on IpMultiMediaCall
   Parameters: valid callSessionID returned in 2., valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned



Test MMCC_ IpMultiMediaCall _02

Summary: IpMultiMediaCall, all methods mandatory, successful

Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

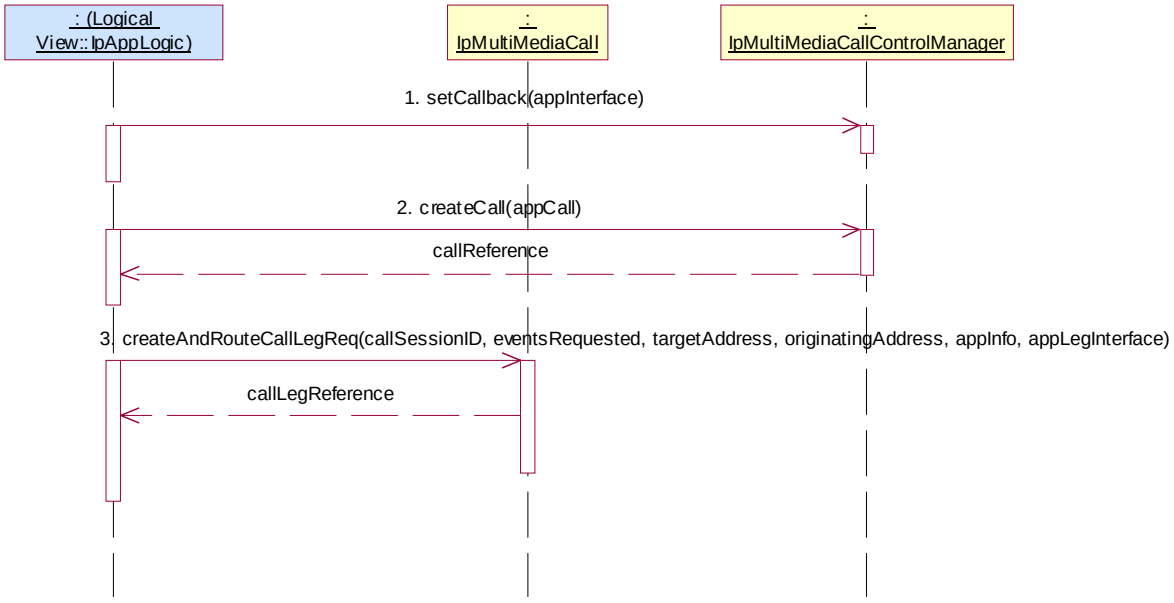
Condition: createCall, createAndRouteCallLeg method is supported.

Test Sequence:

1. Method call **setCallback()** on IpMultiMediaCallControlManager
   Parameters: valid, non-null, value of appInterface parameter
   Check: no exception is returned

2. Method call **createCall()**
   Parameters: valid appCall
   Check: valid value of TpMultiMediaCallIdentifier is returned

3. Method call **createAndRouteCallLegReq()** on IpMultiMediaCall
   Parameters: valid callSessionID returned in 2., valid eventsRequested, valid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: valid value of TpCallLegIdentifier



Test MMCC_ IpMultiMediaCall _03

Summary: IpMultiMediaCall, all methods mandatory, successful

Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Application has a valid callSessionID returned by one of the three following sequence:

1. Method call **setCallback()** on IpMultiMediaCallControlManager
   Parameters: valid, non-null, value of appInterface parameter
   Check: no exception is returned

2. Method call **createCall()**
   Parameters: valid appCall
   Check: valid value of TpMultiMediaCallIdentifier is returned

either

3. Method call **createCallLeg()** on IpMultiMediaCall
   Parameters: valid callSessionID returned in 2., valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

4. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 3., valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

or

3. Method call **createAndRouteCallLegReq()** on IpMultiMediaCall
   Parameters: valid callSessionID returned in 2., valid eventsRequested, valid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: valid value of TpCallLegIdentifier

or

1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, non-null, value, valid notificationRequest
   Check: valid value of TpAssignmentID is returned

2. Triggered action: cause IUT to call Method **reportNotification()** method on the tester's (application's) **IpMultiMediaCallControlManager** interface
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID

either

3. Method call **createCallLeg()** on IpMultiMediaCall
   Parameters: valid callSessionID reported in 2., valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

4. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 3., valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

or

3. Method call **createAndRouteCallLegReq()** on IpMultiMediaCall
   Parameters: valid callSessionID reported in 2., valid eventsRequested, valid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: valid value of TpCallLegIdentifier

or

1. Method call **createMediaNotification()**
   Parameters: valid appInterface, valid notificationMediaRequest
   Check: valid value of TpAssignmentID is returned

2. Triggered action: cause IUT to call Method **reportMediaNotification()** method on the tester's (application's) **IpMultiMediaCallControlManager** interface.
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID

either

3. Method call **createCallLeg()** on IpMultiMediaCall
   Parameters: valid callSessionID reported in 2., valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

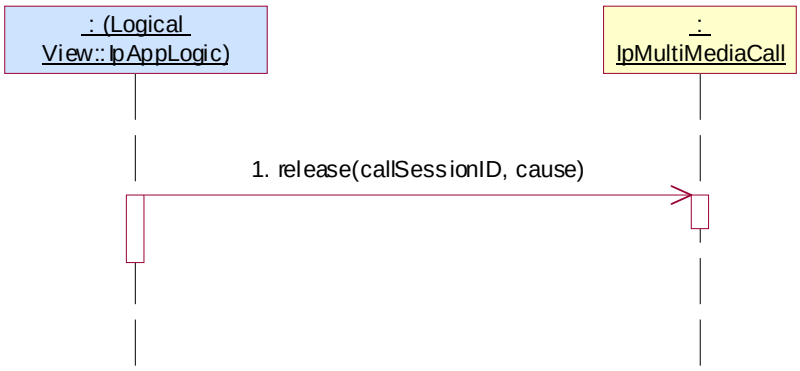
4. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 3., valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

or

3. Method call **createAndRouteCallLegReq()** on IpMultiMediaCall
   Parameters: valid callSessionID reported in 2., valid eventsRequested, valid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: valid value of TpCallLegIdentifier

Test Sequence:

1. Method call **release()** on IpMultiMediaCall
   Parameters: valid callSessionID returned in preamble, valid cause
   Check: no exception is returned



Test MMCC_ IpMultiMediaCall _04

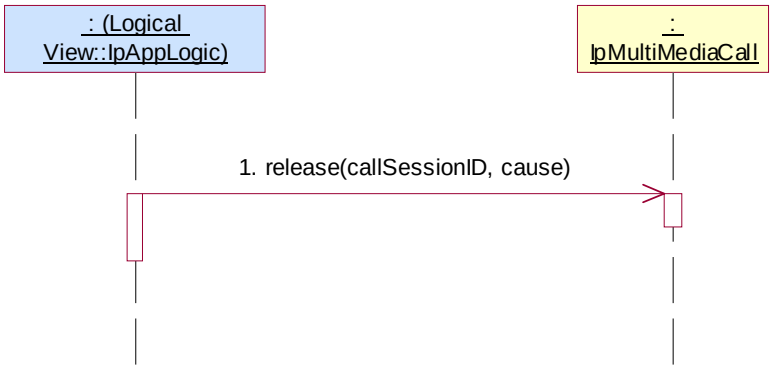
Summary: IpMultiMediaCall, all methods mandatory, successful

Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as MMCC_ IpMultiMediaCall _03

Test Sequence:

1. Method call **deassignCall()** on IpMultiMediaCall
   Parameters: valid callSessionID returned in preamble.
   Check: no exception is returned



5.2.3.2.2 Mandatory, invalid behaviour

Test MMCC_ IpMultiMediaCall _05

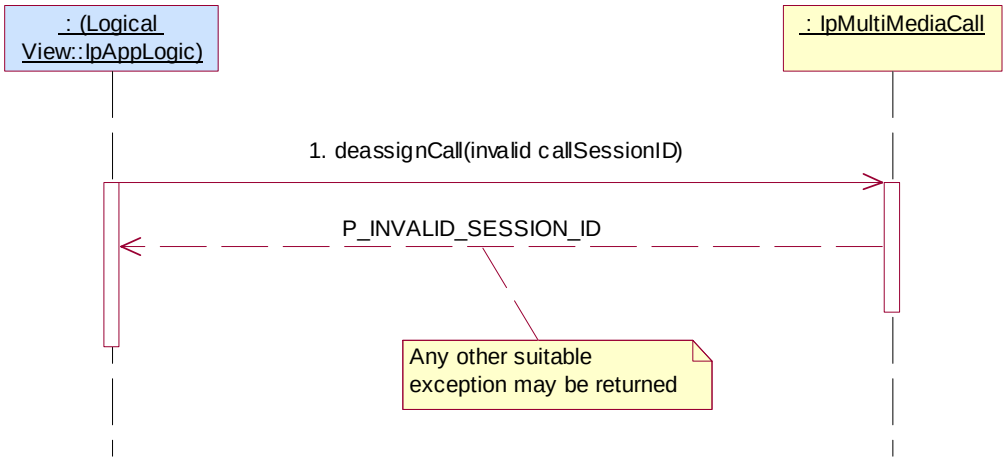
Summary: IpMultiMediaCall, deassignCall, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Same as MMCC_ IpMultiMediaCall _03

Test Sequence:

1. Method call **deassignCall()** on IpMultiMediaCall
   Parameters: invalid callSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCall _06

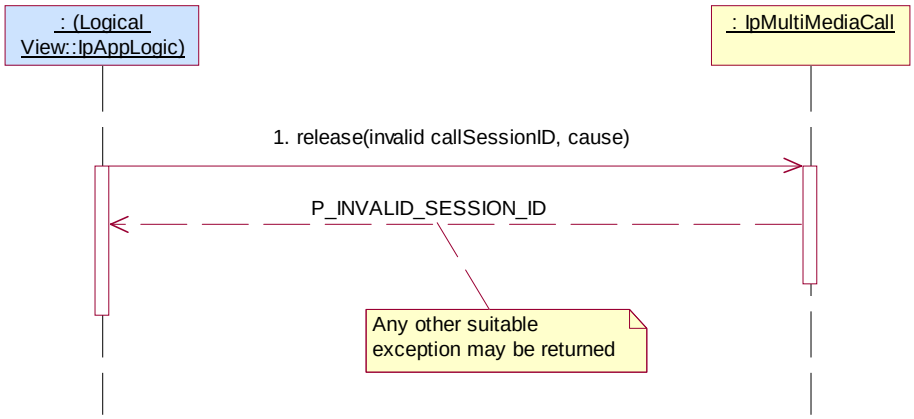
Summary: IpMultiMediaCall, release, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.3

Preamble: Same as MMCC_ IpMultiMediaCall _03

Test Sequence:

1. Method call **release()** on IpMultiMediaCall
   Parameters: invalid callSessionID, valid cause
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCall _07

Summary: IpMultiMediaCall, createCallLeg, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Application has a valid callSessionID returned by one of the two following sequence:

1. Method call **setCallback()** on IpMultiMediaCallControlManager
   Parameters: valid, non-null, value of appInterface parameter
   Check: no exception is returned

2. Method call **createCall()**
   Parameters: valid appCall
   Check: valid value of TpMultiPartyCallIdentifier is returned

or

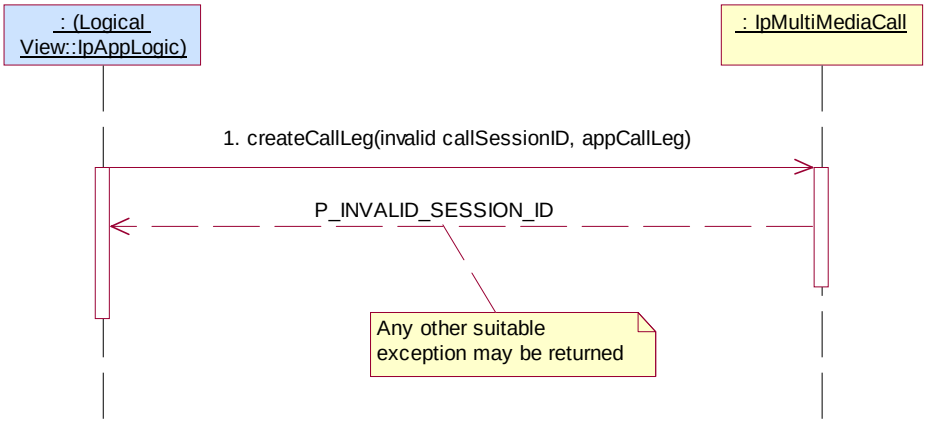
1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, non-null, value, valid notificationRequest
   Check: valid value of TpAssignmentID is returned

2. Triggered action: cause IUT to call Method **reportNotification()** method on the tester's (application)
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID

Condition: createCallLeg method is supported.

Test Sequence:

1. Method call **createCallLeg()** on IpMultiMediaCall
   Parameters: invalid callSessionID, valid appCallLeg
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCall _08

Summary: IpMultiMediaCall, createCallLeg, P_INVALID_INTERFACE_TYPE

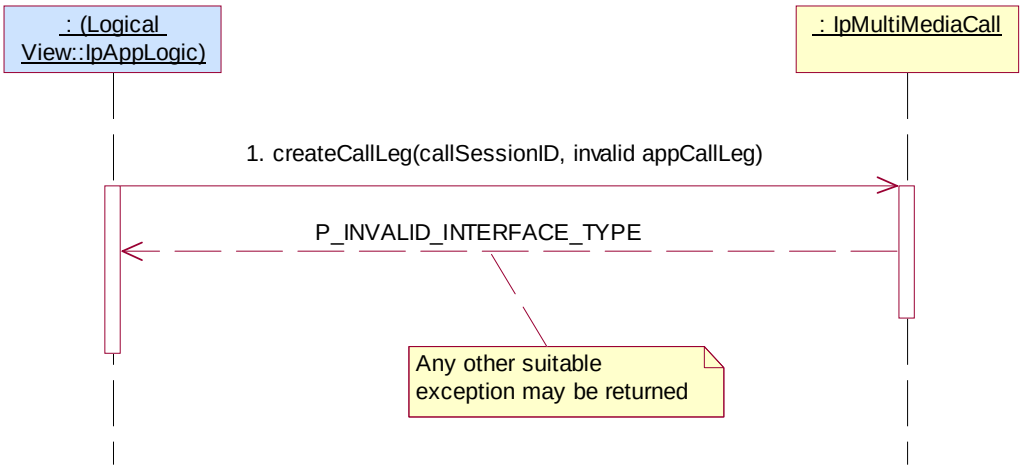
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3

Preamble: Same as MMCC_ IpMultiMediaCall _07

Condition: CreateCallLeg method is supported.

Test Sequence:

1. Method call **createCallLeg()** on IpMultiMediaCall
   Parameters: valid callSessionID returned in preamble, invalid appCallLeg
   Check: P_INVALID_INTERFACE_TYPE, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCall _09

Summary: IpMultiMediaCall, createAndRouteCallLegReq, P_INVALID_SESSION_ID

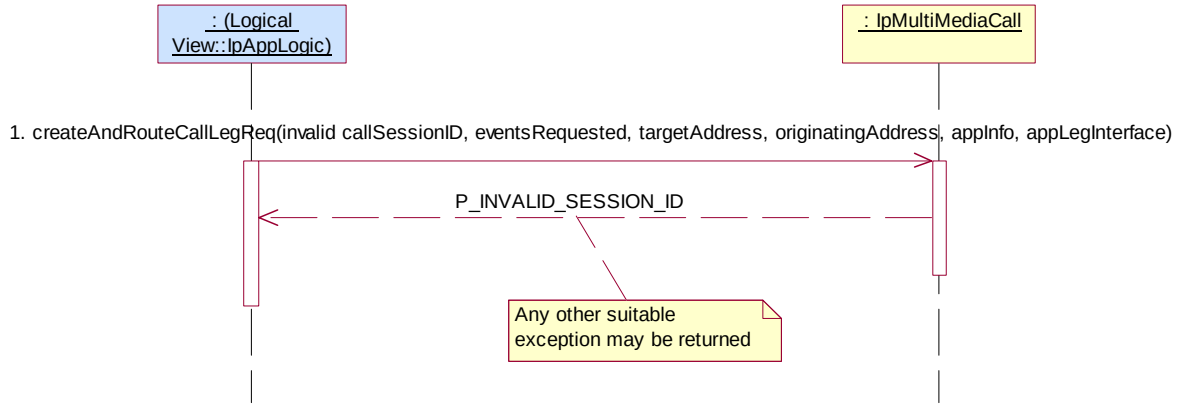
Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Same as MMCC_ IpMultiMediaCall _07

Condition: CreateAndRouteCallLeg method is supported.

Test Sequence:

1. Method call **createAndRouteCallLegReq()** on IpMultiMediaCall
   Parameters: invalid callSessionID, valid eventsRequested, valid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCall _10

Summary: IpMultiMediaCall, createAndRouteCallLegReq, P_INVALID_INTERFACE_TYPE

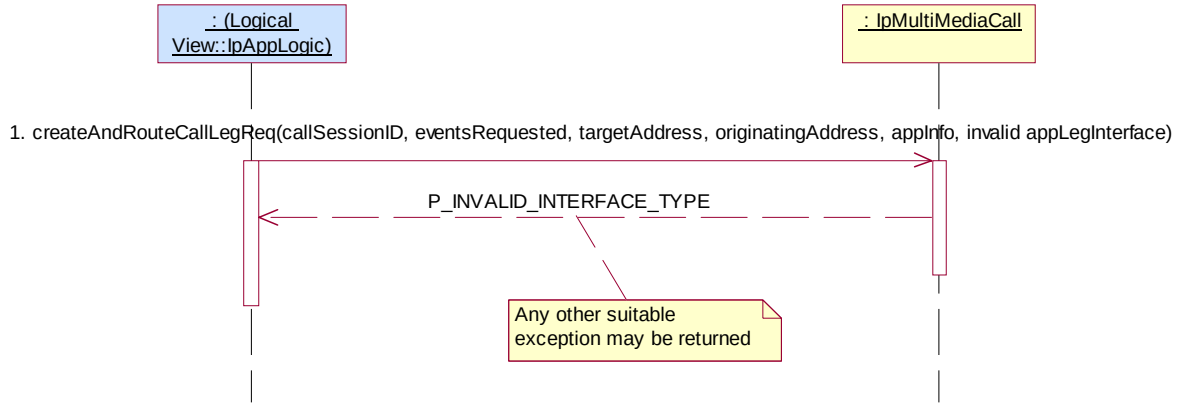
Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Same as MMCC_ IpMultiMediaCall _07

Condition: CreateAndRouteCallLeg method is supported.

Test Sequence:

1. Method call **createAndRouteCallLegReq()** on IpMultiMediaCall
   Parameters: valid callSessionID, valid eventsRequested, valid targetAddress, valid originatingAddress, valid appInfo, invalid appLegInterface
   Check: P_INVALID_INTERFACE_TYPE, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCall _11

Summary: IpMultiMediaCall, createAndRouteCallLegReq, P_INVALID_ADDRESS

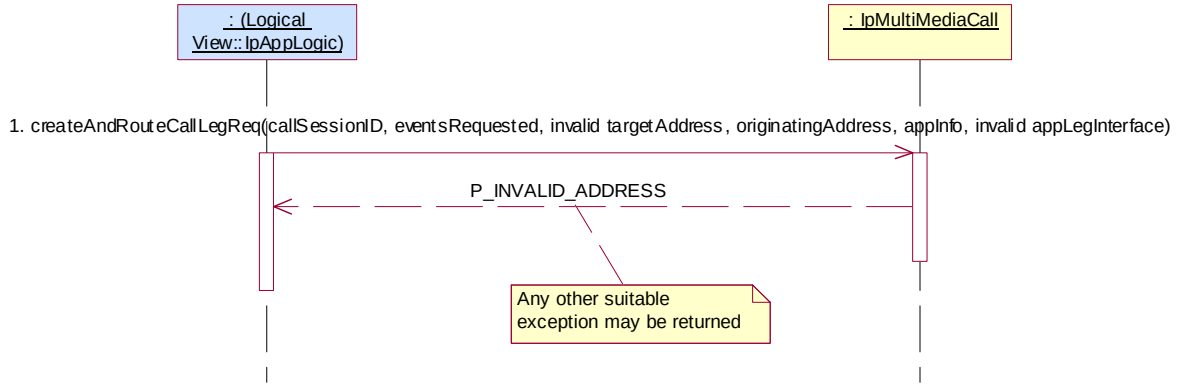
Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Same as MMCC_ IpMultiMediaCall _07

Condition: CreateAndRouteCallLeg method is supported.

Test Sequence:

1. Method call **createAndRouteCallLegReq()** on IpMultiMediaCall
   Parameters: valid callSessionID, valid eventsRequested, invalid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: P_INVALID_ADDRESS, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCall _12

Summary: IpMultiMediaCall, createAndRouteCallLegReq, P_INVALID_ADDRESS

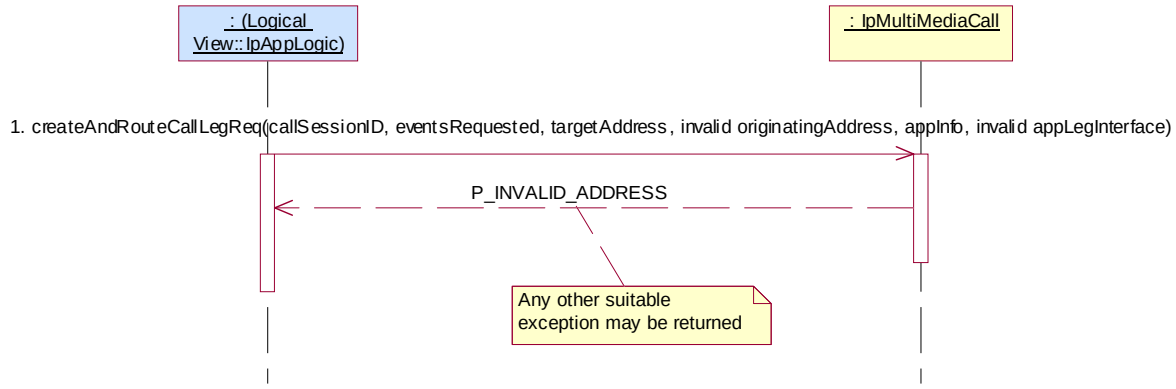
Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Same as MMCC_ IpMultiMediaCall _07

Condition: CreateAndRouteCallLeg method is supported.

Test Sequence:

1. Method call **createAndRouteCallLegReq()** on IpMultiMediaCall
   Parameters: valid callSessionID, valid eventsRequested, valid targetAddress, invalid originatingAddress, valid appInfo, valid appLegInterface
   Check: P_INVALID_ADDRESS, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCall _13

Summary: IpMultiMediaCall, createAndRouteCallLegReq, P_INVALID_CRITERIA

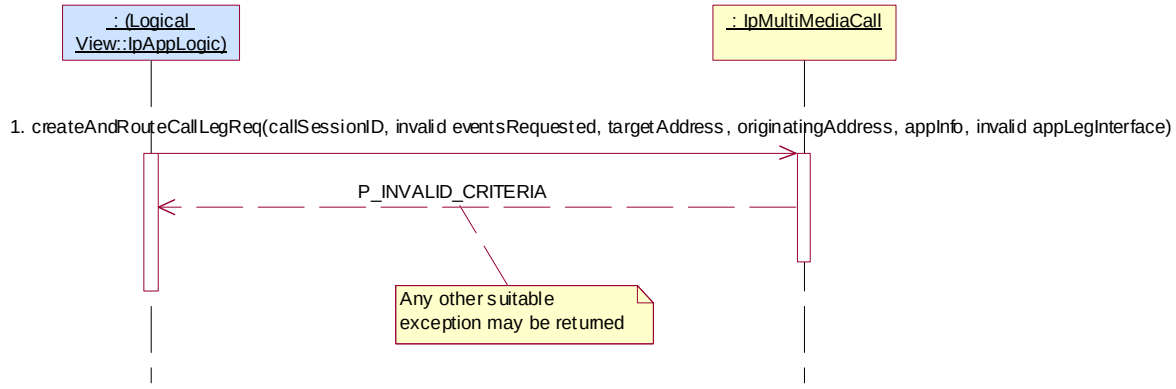
Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Same as MMCC_ IpMultiMediaCall _07

Condition: CreateAndRouteCallLeg method is supported.

Test Sequence:

1. Method call **createAndRouteCallLegReq()** on IpMultiMediaCall
   Parameters: valid callSessionID, invalid eventsRequested, valid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: P_INVALID_CRITERIA, or another suitable exception, is returned



5.2.3.2.3 Optional, valid behaviour

Test MMCC_ IpMultiMediaCall _14

Summary: IpMultiMediaCall, all methods, successful

Reference: ES 201 915-4 [1], clauses 7.3.1 and 8.3.1

Preamble: Application has a valid callSessionID returned by one of the two following sequence:

1. Method call **setCallback()** on IpMultiMediaCallControlManager
   Parameters: valid, non-null, value of appInterface parameter
   Check: no exception is returned

2. Method call **createCall()**
   Parameters: valid appCall
   Check: valid value of TpMultiPartyCallIdentifier is returned

3. Method call **createCallLeg()** on IpMultiMediaCall
   Parameters: valid callSessionID returned in 2., valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

or

1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, non-null, value, valid notificationRequest
   Check: valid value of TpAssignmentID is returned

2. Triggered action: cause IUT to call Method **reportNotification()** method on the tester's (application) **IpAppMultiMediaCallControlManager interface**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID

3. Method call **createCallLeg()** on IpMultiMediaCall
   Parameters: valid callSessionID reported in 2., valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

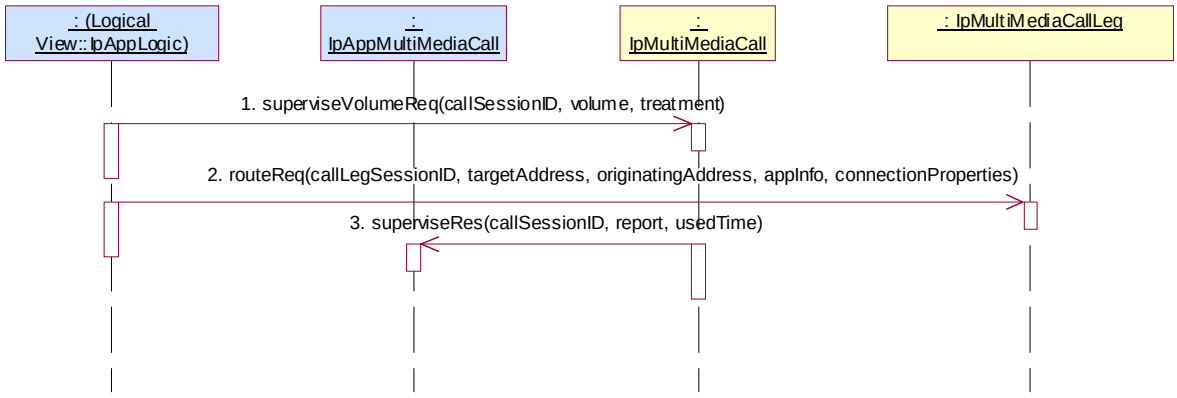
Condition: superviseVolumeReq method is supported.

Test Sequence:

1. Method call **superviseVolumeReq()** on IpMultiMediaCall
   Parameters: valid callSessionID returned in preamble, valid volume, valid treatment
   Check: no exception is returned

2. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

3. Triggered action: cause IUT to call Method **superviseVolumeRes()** method on the tester's (application) **IpMultiMediaCall** interface.
   Parameters: callSessionID, report, usedVolume


: (Logical View::IpAppLogic)
: IpAppMultiMediaCall
: IpMultiMediaCall
: IpMultiMediaCallLeg
1. superviseVolumeReq(callSessionID, volume, treatment)
2. routeReq(callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties)
3. superviseRes(callSessionID, report, usedTime)

Test MMCC_ IpMultiMediaCall _15

Summary: IpMultiMediaCall, getInfoReq, successful

Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as MMCC_ IpMultiMediaCall _14

Condition: createCallLeg and getInfoReq methods are supported.

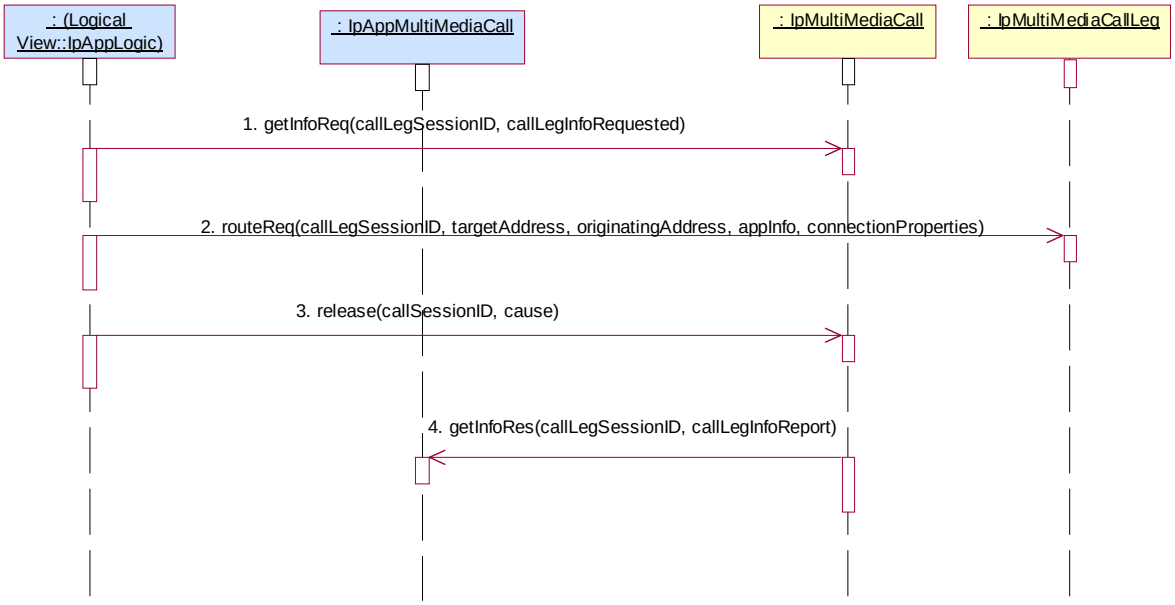
Test Sequence:

1. Method call **getInfoReq()** on IpMultiMediaCall
   Parameters: valid callSessionID returned in preamble, valid callInfoRequested
   Check: no exception is returned

2. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

3. Method call **release()** on IpMultiMediaCall
   Parameters: valid callSessionID returned in preamble, valid cause
   Check: no exception is returned

4. Triggered action: cause IUT to call **getInfoRes()** method on the tester's (Application) **IpAppMultiMediaCall** interface.
   Parameters: callSessionID given in 1., valid callInfoReport.



Test MMCC_ IpMultiMediaCall _16

Summary: IpMultiMediaCall, setChargePlan, successful

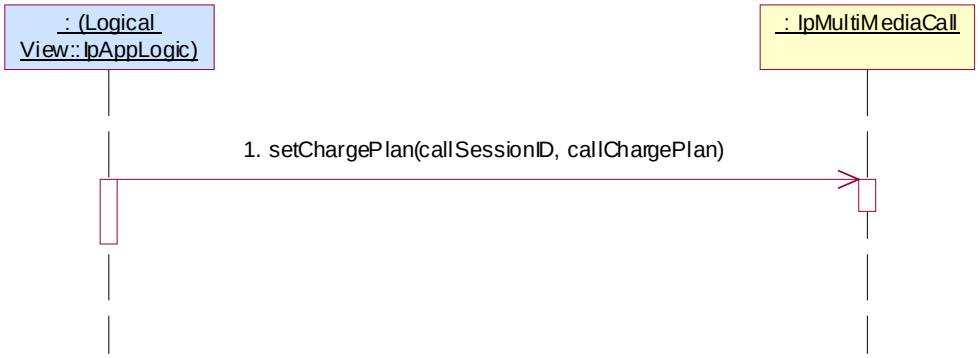
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as MMCC_ IpMultiMediaCall _14

Condition: createCallLeg and setChargePlan methods are supported.

Test Sequence:

1. Method call **setChargePlan()** on IpMultiMediaCall
   Parameters: valid callSessionID returned in preamble, valid callChargePlan
   Check: no exception is returned



Test MMCC_ IpMultiMediaCall _17

Summary: IpMultiMediaCall, setAdviceOfCharge, successful

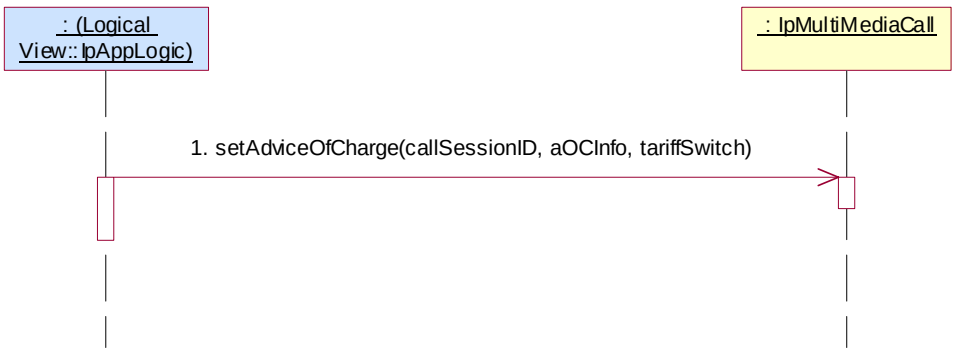
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as MMCC_ IpMultiMediaCall _14

Condition: createCallLeg and setAdviceOfCharge methods are supported.

Test Sequence:

1. Method call **setAdviceOfCharge()** on IpMultiMediaCall
   Parameters: valid callSessionID returned in 1., valid aOCInfo, valid tariffSwitch
   Check: no exception is returned



Test MMCC_ IpMultiMediaCall _18

Summary: IpMultiMediaCall, superviseReq, successful

Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as MMCC_ IpMultiMediaCall _14

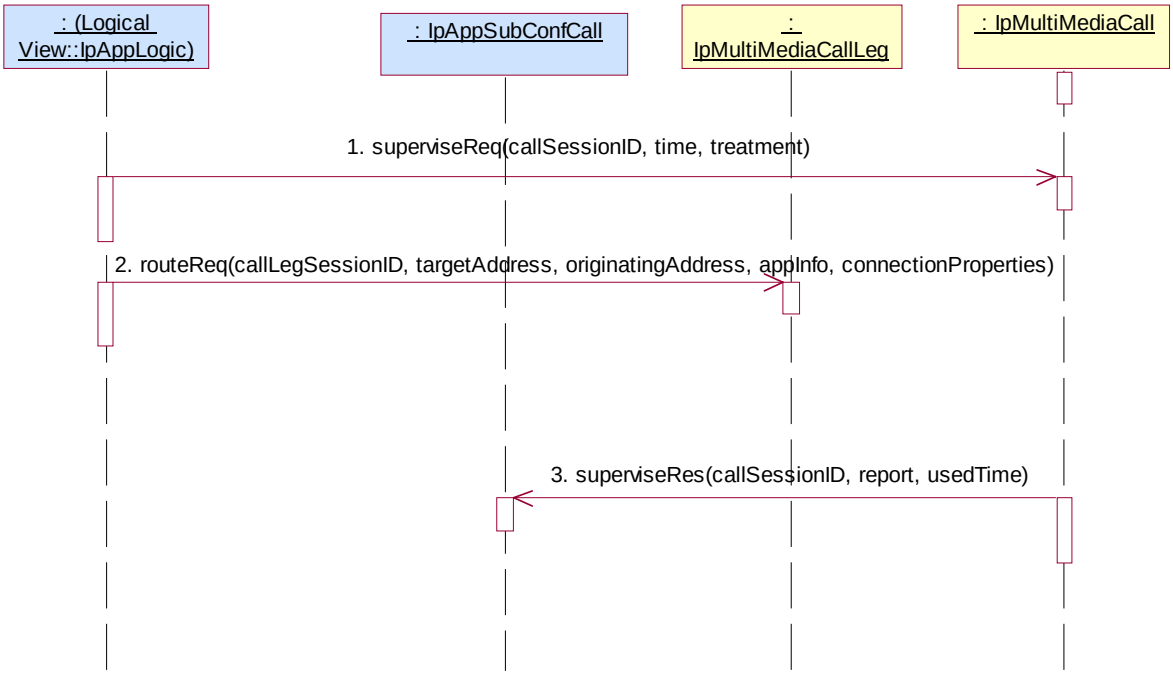
Condition: createCallLeg and superviseReq methods are supported.

Test Sequence:

1. Method call **superviseReq()** on IpMultiMediaCall
   Parameters: valid callSessionID returned in preamble, valid time, valid treatment
   Check: no exception is returned

2. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

3 Triggered action: cause IUT to call **superviseRes()** method on the tester's (Application) **IpAppMultiMediaCall** interface.
   Parameters: callSessionID given in 1., valid report, valid usedTime.



Test MMCC_ IpMultiMediaCall _19

Summary: IpMultiMediaCall, all methods, successful

Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as MMCC_ IpMultiMediaCall _03

Condition: getCallLegs method is supported.

Test Sequence:

1. Method call **getCallLegs()** on IpMultiMediaCall
   Parameters: valid callSessionID returned in preamble.
   Check: valid value of TpCallLegIdentifierSet which contains CallLegIdentifier returned in preamble.

5.2.3.2.4 Optional, invalid behaviour

Test MMCC_ IpMultiMediaCall _20

Summary: IpMultiMediaCall, superviseVolumeReq, P_INVALID_SESSION_ID

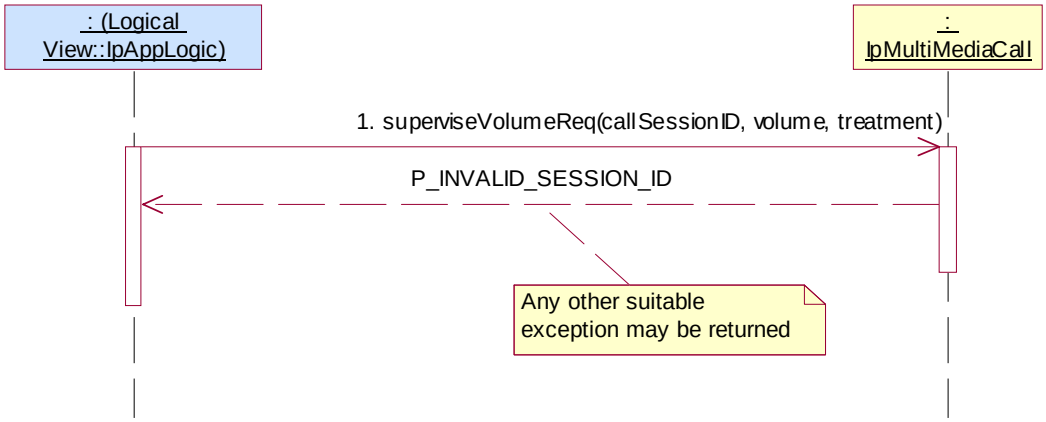
Reference: ES 201 915-4 [1], clause 8.3.1

Preamble: Same as MMCC_ IpMultiMediaCall _14

Condition: superviseVolumeReq method is supported.

Test Sequence:

1. Method call **superviseVolumeReq()** on IpMultiMediaCall
   Parameters: invalid callSessionID, valid volume, valid treatment
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCall _21

Summary: IpMultiMediaCall, getInfoReq, P_INVALID_SESSION_ID

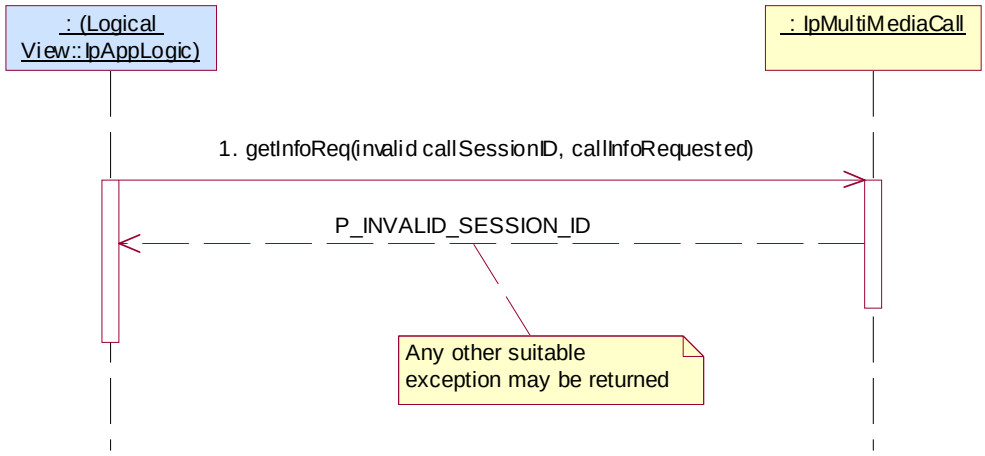
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as MMCC_ IpMultiMediaCall _14

Condition: createCallLeg and getInfoReq methods are supported.

Test Sequence:

1. Method call **getInfoReq()** on IpMultiMediaCall
   Parameters: invalid callSessionID, valid callInfoRequested
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCall _22

Summary: IpMultiMediaCall, setChargePlan, P_INVALID_SESSION_ID

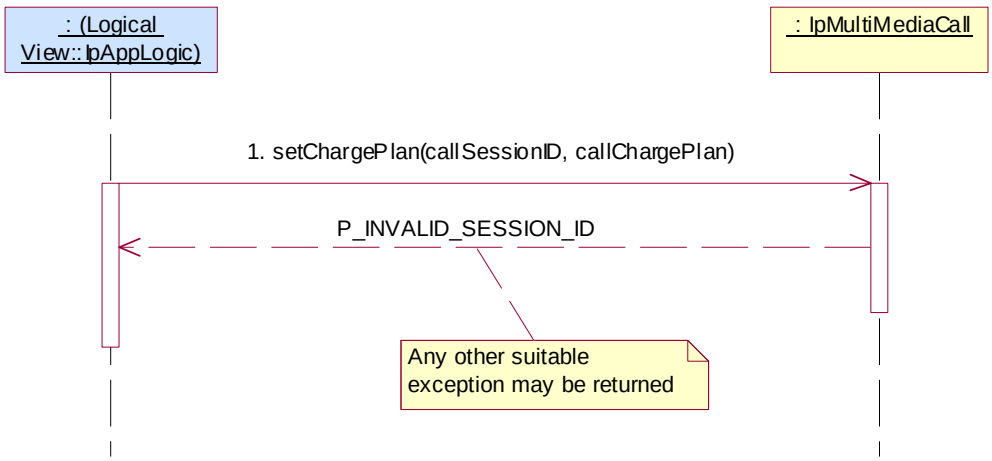
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as MMCC_ IpMultiMediaCall _14

Condition: createCallLeg and setChargePlan methods are supported.

Test Sequence:

1. Method call **setChargePlan()** on IpMultiMediaCall
   Parameters: invalid callSessionID, valid callChargePlan
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCall _23

Summary: IpMultiMediaCall, setAdviceOfCharge, P_INVALID_SESSION_ID

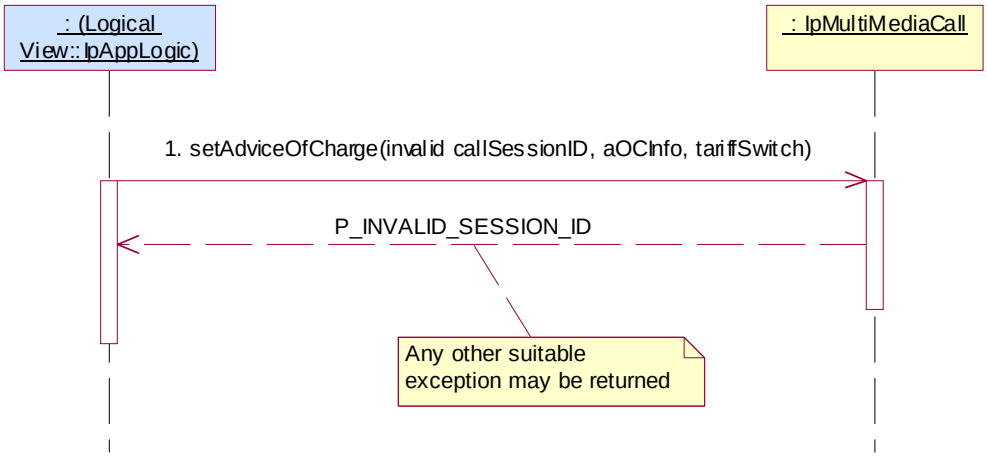
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as MMCC_ IpMultiMediaCall _14

Condition: createCallLeg and setAdviceOfCharge methods are supported.

Test Sequence:

1. Method call **setAdviceOfCharge()** on IpMultiMediaCall
   Parameters: invalid callSessionID, valid aOCInfo, valid tariffSwitch
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCall _24

Summary: IpMultiMediaCall, setAdviceOfCharge, P_INVALID_CURRENCY

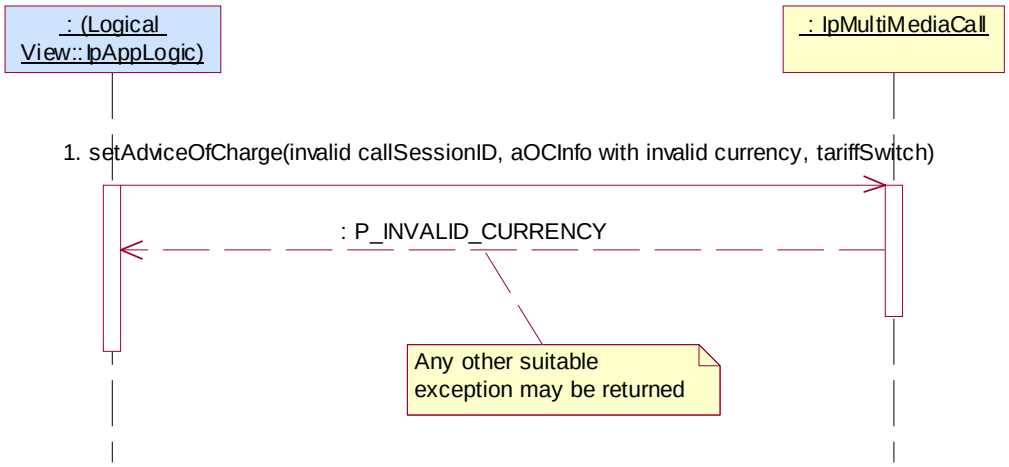
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as MMCC_ IpMultiMediaCall _14

Condition: createCallLeg and setAdviceOfCharge methods are supported.

Test Sequence:

1. Method call **setAdviceOfCharge()** on IpMultiMediaCall
   Parameters: valid callSessionID returned in 1., aOCInfo with invalid currency, valid tariffSwitch
   Check: P_INVALID_CURRENCY, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCall _25

Summary: IpMultiMediaCall, setAdviceOfCharge, P_INVALID_AMOUNT

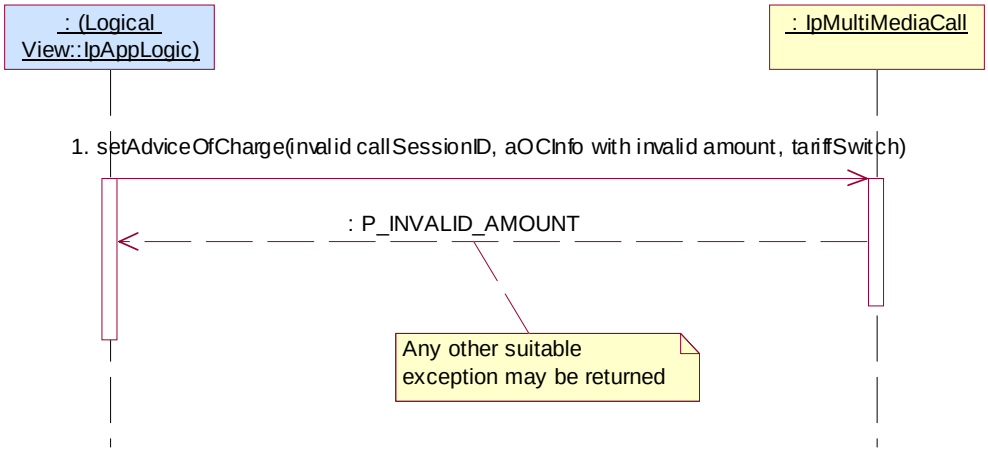
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as MMCC_ IpMultiMediaCall _14

Condition: createCallLeg and setAdviceOfCharge methods are supported.

Test Sequence:

1. Method call **setAdviceOfCharge()** on IpMultiMediaCall
   Parameters: valid callSessionID returned in 1., aOCInfo, with invalid amount, valid tariffSwitch
   Check: P_INVALID_AMOUNT, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCall _26

Summary: IpMultiMediaCall, superviseReq, P_INVALID_SESSION_ID

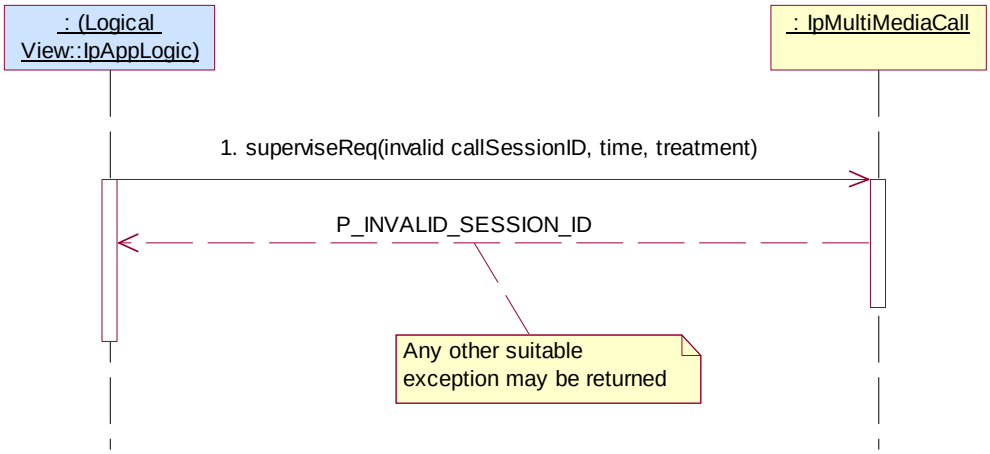
Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Same as MMCC_ IpMultiMediaCall _14

Condition: createCallLeg and superviseReq methods are supported.

Test Sequence:

1. Method call **superviseReq()** on IpMultiMediaCall
   Parameters: invalid callSessionID, valid time, valid treatment
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCall _27

Summary: IpMultiMediaCall, getCallLegs, P_INVALID_SESSION_ID

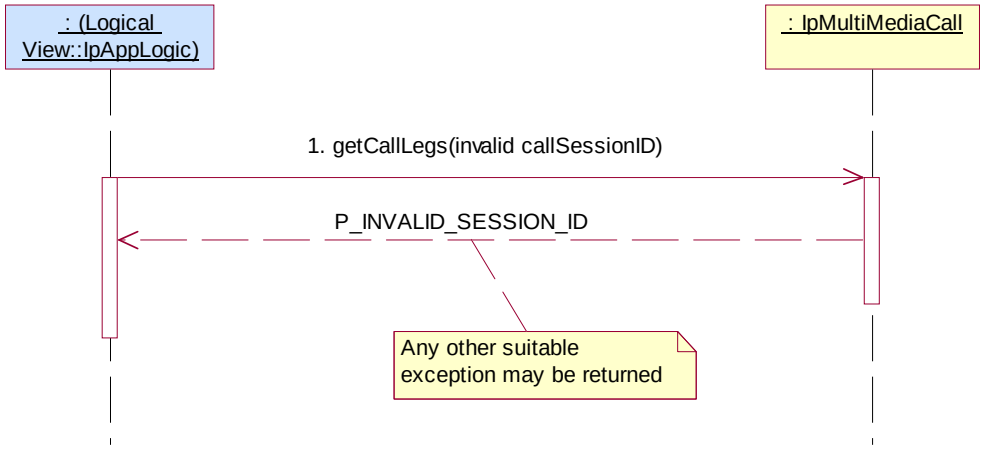
Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Same as MMCC_ IpMultiMediaCall _03

Condition: CreateCallLeg method is supported.

Test Sequence:

1. Method call **getCallLegs()** on IpMultiMediaCall
   Parameters: invalid callSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



5.2.3.3 IpMultiMediaCallLeg

5.2.3.3.1 Mandatory, valid behaviour

Test MMCC_ IpMultiMediaCallLeg _01

Summary: IpMultiMediaCallLeg, all methods mandatory, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 8.3.5.

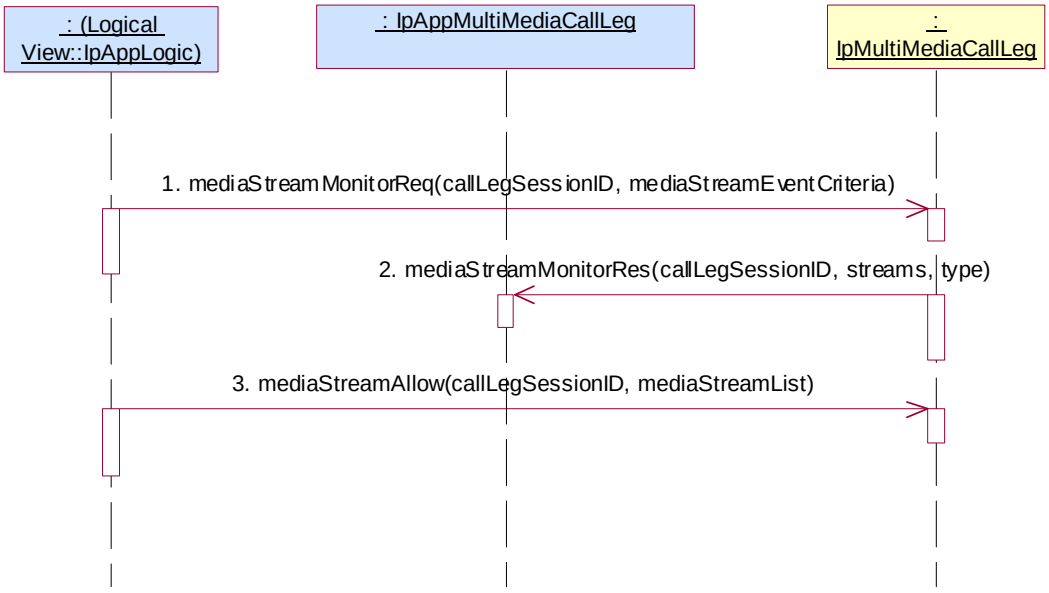
Preamble: Same as MMCC_ IpMultiMediaCall _03

Test Sequence:

1. Method call **mediaStreamMonitorReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid mediaStreamEventCriteria
   Check: no exception is returned

2. Triggered action: cause IUT to call Method **mediaStreamMonitorRes()** method on the tester's (application)
   Parameters: callLegSessionID, streams, type

3. Method call **mediaStreamAllow()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid mediaStreamList
   Check: no exception is returned



Test MMCC_ IpMultiMediaCallLeg _02

Summary: IpMultiMediaCallLeg, all mandatory methods, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as MMCC_ IpMultiMediaCall _07

Condition: createCallLeg method is supported

Test Sequence:

1. Method call **createCallLeg()** on IpMultiMediaCall
   Parameters: valid callSessionID returned in preamble, valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

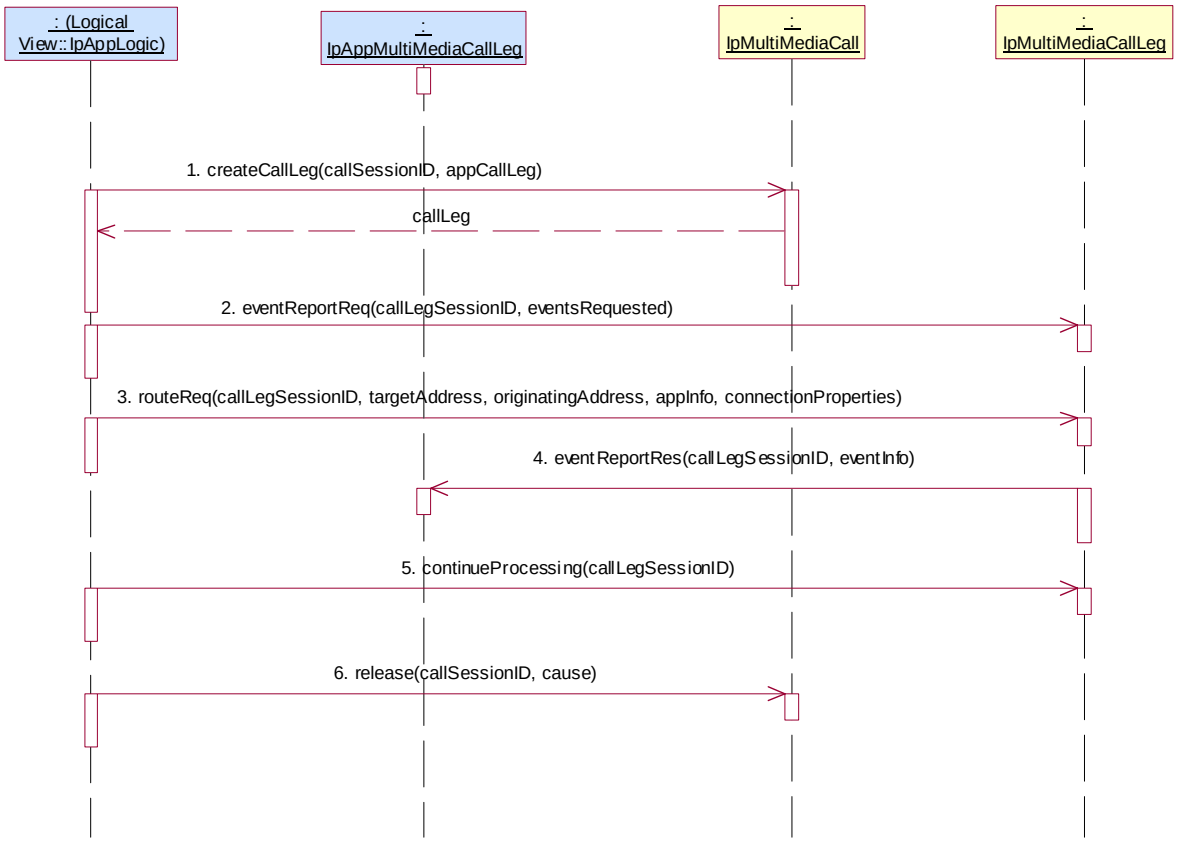
2. Method call **eventReportReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 1, valid eventsRequested
   Check: no exception is returned

3. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 1., valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

4 Triggered action: cause IUT to interrupt call leg processing with a notification or an event: cause IUT to call **eventReportRes()** method on the tester's (Application) **IpAppCallLeg** interface.
   Parameters: callLegSessionID, errorIndication

5. Method call **continueProcessing()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 1.
   Check: no exception is returned

6. Method call **release()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 1, valid cause
   Check: no exception is returned


: (Logical View:: IpAppLogic)
: IpAppMultiMediaCallLeg
: IpMultiMediaCall
: IpMultiMediaCallLeg
1. createCallLeg(callSessionID, appCallLeg)
callLeg
2. eventReportReq(callLegSessionID, eventsRequested)
3. routeReq(callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties)
4. eventReportRes(callLegSessionID, eventInfo)
5. continueProcessing(callLegSessionID)
6. release(callSessionID, cause)

Test MMCC_ IpMultiMediaCallLeg _03

Summary: IpMultiMediaCallLeg, all mandatory methods, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as MMCC_ IpMultiMediaCall _07

Condition: createAndRouteCallLeg method is supported

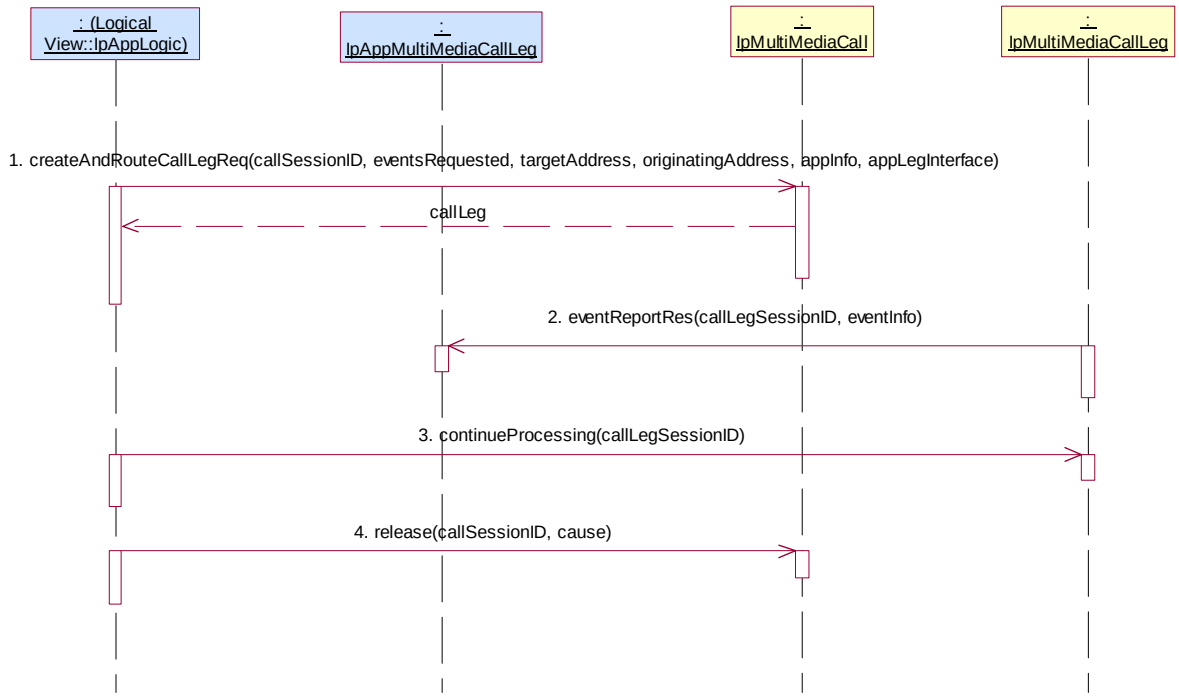
Test Sequence:

1. Method call **createAndRouteCallLeg()** on IpMultiMediaCall
   Parameters: valid callSessionID returned in preamble, valid eventsRequested, valid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: valid value of TpCallLegIdentifier is returned

2 Triggered action: cause IUT to interrupt call leg processing with a notification or an event: cause IUT to call **eventReportRes()** method on the tester's (Application) **IpAppCallLeg** interface.
   Parameters: callLegSessionID returned in 1, errorIndication

3. Method call **continueProcessing()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble.
   Check: no exception is returned

4. Method call **release()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid cause
   Check: no exception is returned



Test MMCC_ IpMultiMediaCallLeg _04

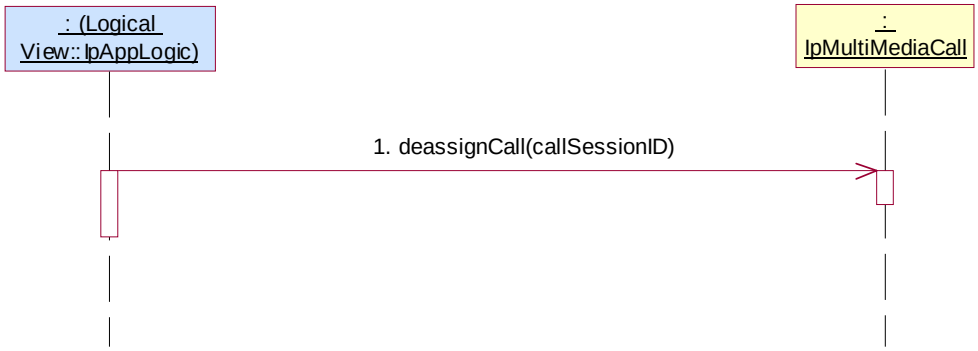
Summary: IpMultiMediaCallLeg, all mandatory methods, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as MMCC_ IpMultiMediaCall _03

Test Sequence:

1. Method call **deassign()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble.
   Check: no exception is returned



5.2.3.3.2 Mandatory, invalid behaviour

Test MMCC_ IpMultiMediaCallLeg _05

Summary: IpMultiMediaCallLeg, continueProcessing, : P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.5

Preamble: Same as MMCC_ IpMultiMediaCall _07

Test Sequence:

1. Method call **createCallLeg()** on IpMultiMediaCall
   Parameters: valid callSessionID returned in preamble, valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

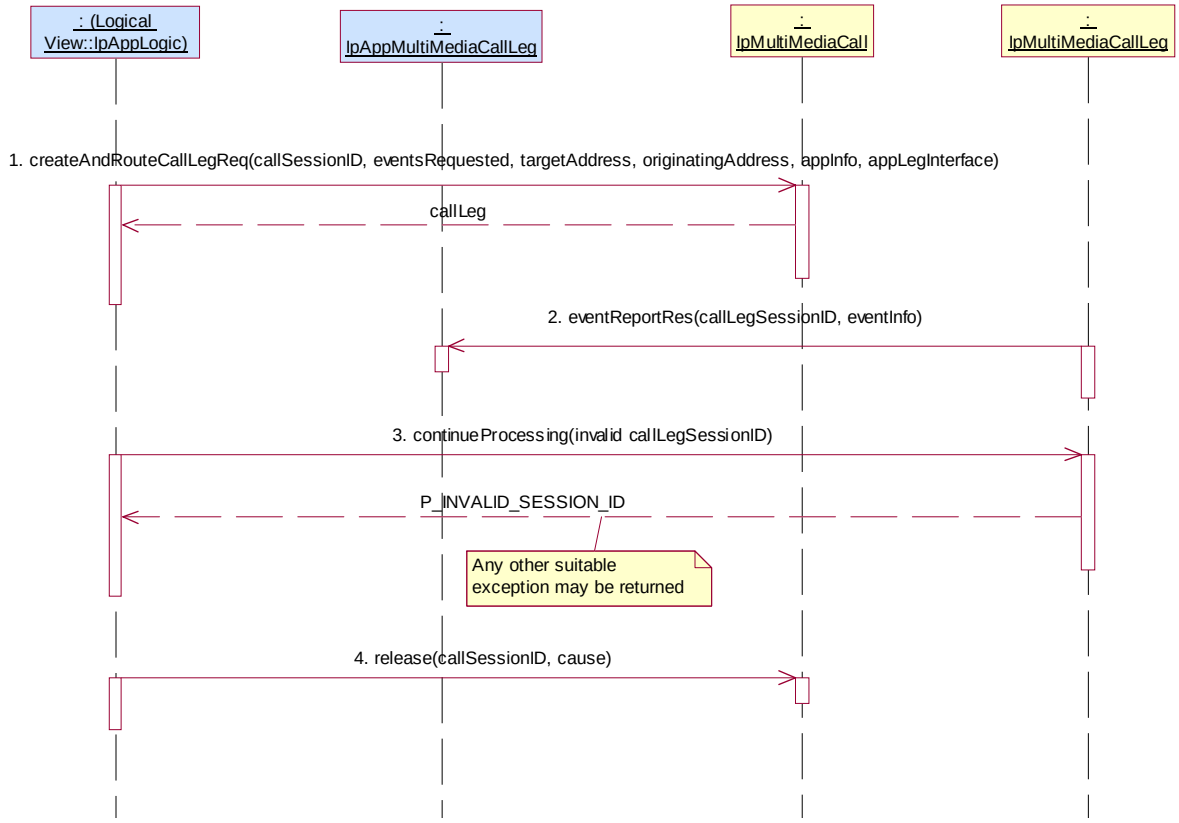
2. Method call **eventReportReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 1, valid eventsRequested
   Check: no exception is returned

3. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 1., valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

4. Triggered action: cause IUT to interrupt call leg processing with a notification or an event: cause IUT to call **eventReportRes()** method on the tester's (Application) **IpAppCallLeg** interface.
   Parameters: callLegSessionID, errorIndication

5. Method call **continueProcessing()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned

6. Method call **release()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 1, valid cause
   Check: no exception is returned



Test MMCC_ IpMultiMediaCallLeg _06

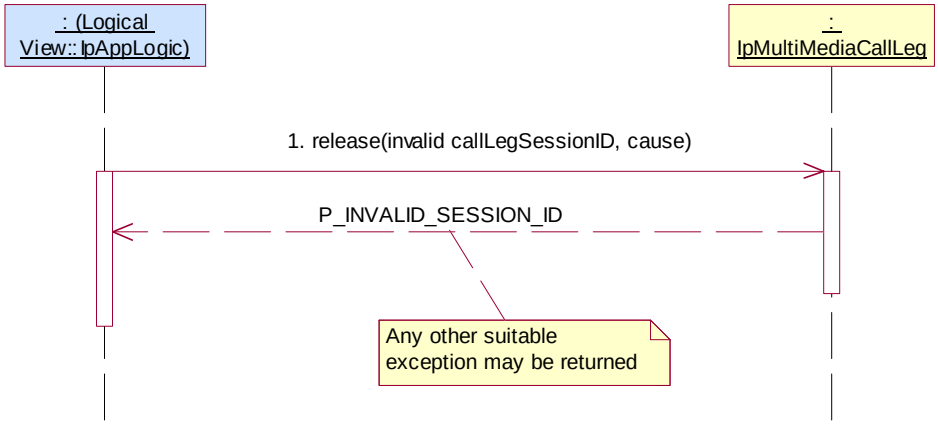
Summary: IpMultiMediaCallLeg, release**,** P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.5

Preamble: Same as MMCC_ IpMultiMediaCall _03

Test Sequence:

1. Method call **release()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID, valid cause
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCallLeg _07

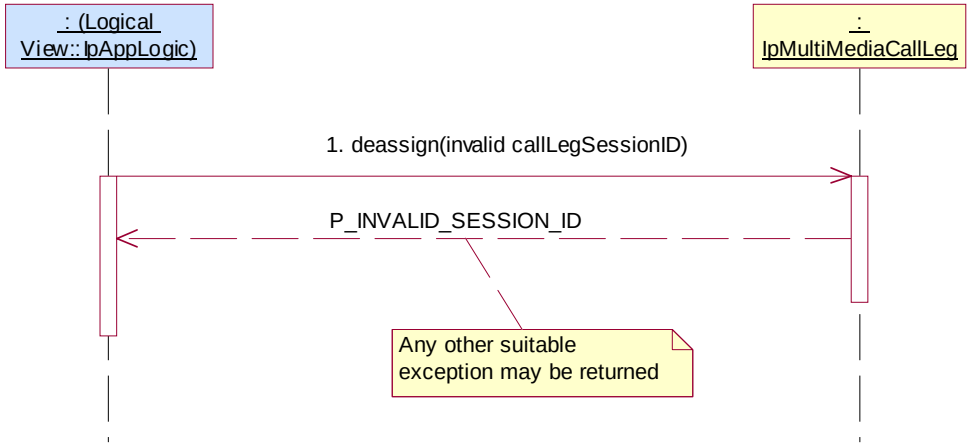
Summary: IpMultiMediaCallLeg, deassign**,** P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.5

Preamble: Same as MMCC_ IpMultiMediaCall _03

Test Sequence:

1. Method call **deassign()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCallLeg _08

Summary: IpMultiMediaCallLeg, mediaStreamAllow, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 8.3.5.

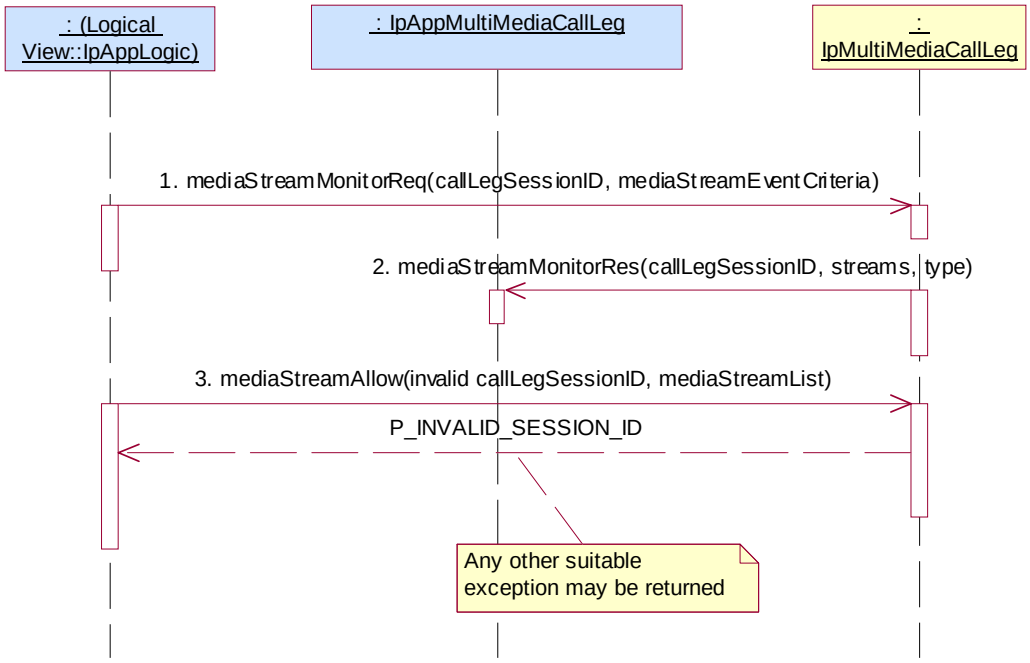
Preamble: Same as MMCC_ IpMultiMediaCall _03

Test Sequence:

1. Method call **mediaStreamMonitorReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid mediaStreamEventCriteria
   Check: no exception is returned

2. Triggered action: cause IUT to call Method **mediaStreamMonitorRes()** method on the tester's (application)
   Parameters: callLegSessionID, streams, type

3. Method call **mediaStreamAllow()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID, valid mediaStreamList
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCallLeg _09

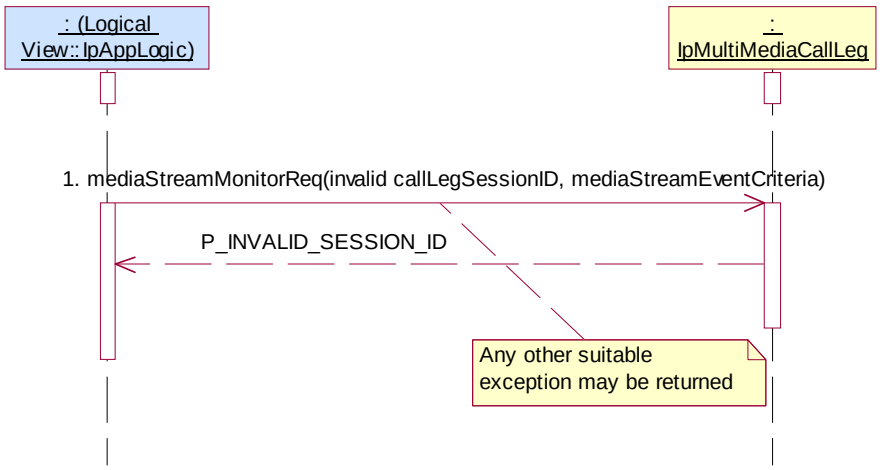
Summary: IpMultiMediaCallLeg, mediaStreamMonitorReq, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 8.3.5.

Preamble: Same as MMCC_ IpMultiMediaCall _03

Test Sequence:

1. Method call **mediaStreamMonitorReq()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID, valid mediaStreamEventCriteria
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCallLeg _10

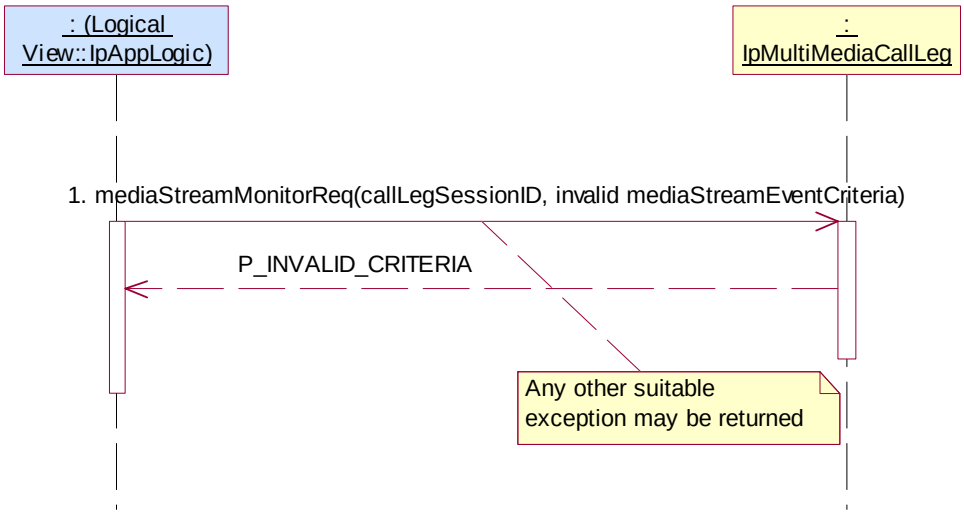
Summary: IpMultiMediaCallLeg, mediaStreamMonitorReq, P_INVALID_CRITERIA

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 8.3.5.

Preamble: Same as MMCC_ IpMultiMediaCall _03

Test Sequence:

1. Method call **mediaStreamMonitorReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 2., valid mediaStreamEventCriteria with invalid criteria
   Check: P_INVALID_CRITERIA, or another suitable exception, is returned



5.2.3.3.3 Optional, valid behaviour

Test MMCC_ IpMultiMediaCallLeg _11

Summary: IpMultiMediaCallLeg, getInfoReq, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

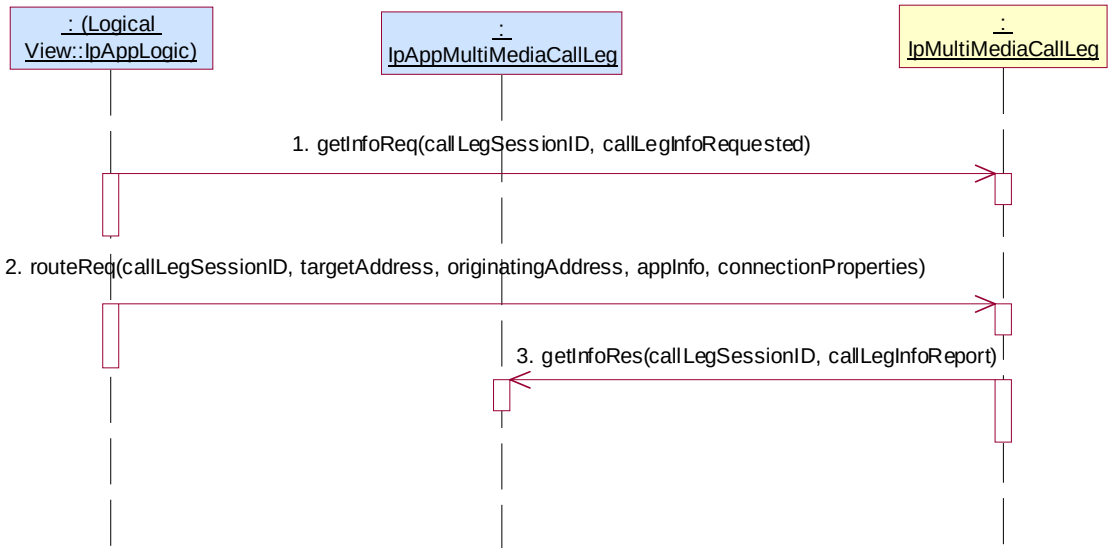
Preamble: Same as MMCC_ IpMultiMediaCall _14

Test Sequence:

1. Method call **getInfoReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid callLegInfoRequested
   Check: no exception is returned

2. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

3 Triggered action: cause IUT to call **getInfoRes()** method on the tester's (Application) **IpAppCallLeg** interface.
   Parameters: callLegSessionID given in 1., valid callLegInfoReport.



Test MMCC_ IpMultiMediaCallLeg _12

Summary: IpMultiMediaCallLeg, attachMediaReq, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Application has a valid callSessionID returned by one of the three following sequence:

1. Method call **setCallback()** on IpMultiMediaCallControlManager
   Parameters: valid, non-null, value of appInterface parameter
   Check: no exception is returned

2. Method call **createCall()**
   Parameters: valid appCall
   Check: valid value of TpMultiMediaCallIdentifier is returned

3. Method call **createCallLeg()** on IpMultiMediaCall
   Parameters: valid callSessionID returned in 2., valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

4. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 3., valid targetAddress, valid appInfo, valid connectionProperties set to have explicit media management
   Check: no exception is returned

or

1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, non-null, value, valid notificationRequest
   Check: valid value of TpAssignmentID is returned

2. Triggered action: cause IUT to call Method **reportNotification()** method on the tester's (application's) **IpMultiMediaCallControlManager** interface
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID

3. Method call **createCallLeg()** on IpMultiMediaCall
   Parameters: valid callSessionID reported in 2., valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

4. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 3., valid targetAddress, valid appInfo, valid connectionProperties set to have explicit media management
   Check: no exception is returned

or

1. Method call **createMediaNotification()**
   Parameters: valid appInterface, valid notificationMediaRequest
   Check: valid value of TpAssignmentID is returned

2. Triggered action: cause IUT to call Method **reportMediaNotification()** method on the tester's (application's) **IpMultiMediaCallControlManager** interface.
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID

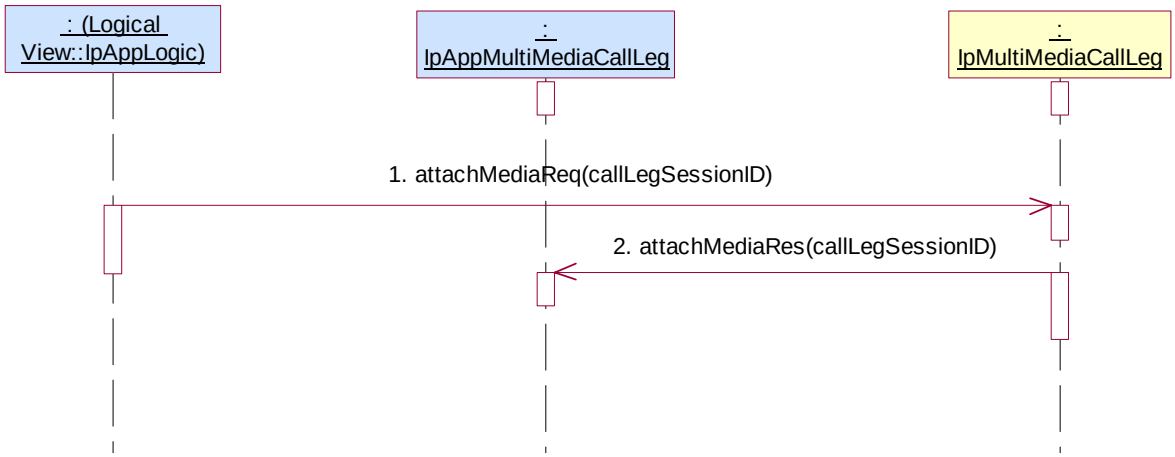
3. Method call **createCallLeg()** on IpMultiMediaCall
   Parameters: valid callSessionID reported in 2., valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

4. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 3., valid targetAddress, valid appInfo, valid connectionProperties set to have explicit media management
   Check: no exception is returned

Test Sequence:

1. Method call **attachMediaReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble.
   Check: no exception is returned

2 Triggered action: cause IUT to call **attachMediaRes()** method on the tester's (Application) **IpAppCallLeg** interface.
   Parameters: callLegSessionID



Test MMCC_ IpMultiMediaCallLeg _13

Summary: IpMultiMediaCallLeg, detachMediaReq, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as MMCC_ IpMultiMediaCallleg _12

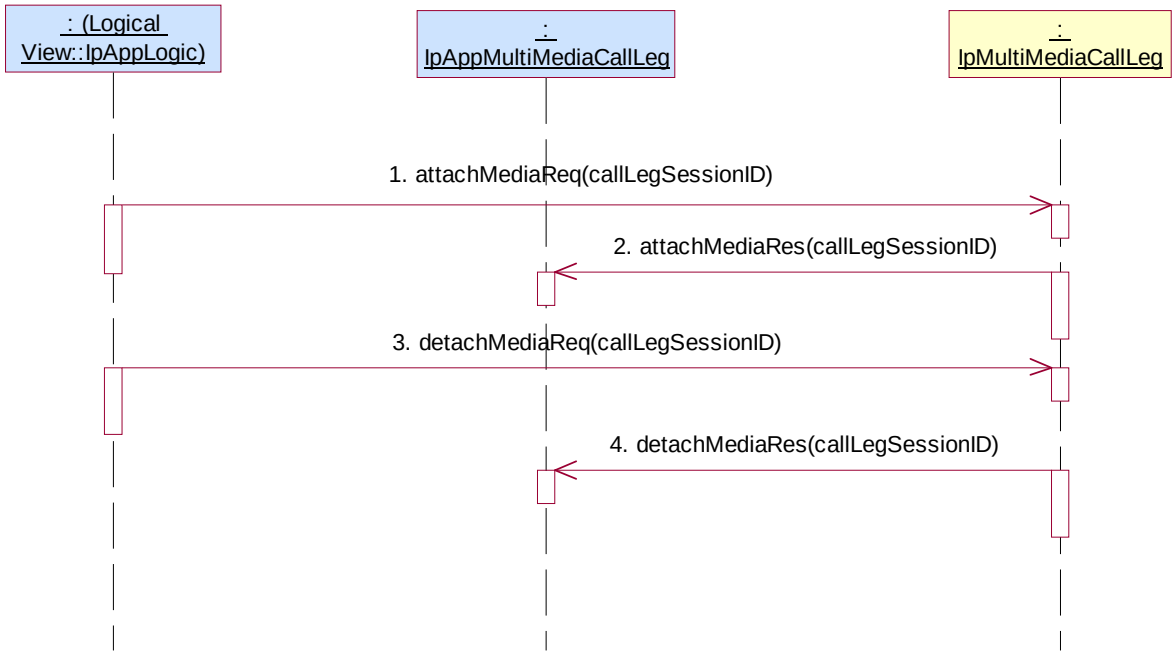
Test Sequence:

1. Method call **attachMediaReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble.
   Check: no exception is returned

2 Triggered action: cause IUT to call **attachMediaRes()** method on the tester's (Application) **IpAppCallLeg** interface.
   Parameters: callLegSessionID

3. Method call **detachMediaReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble.
   Check: no exception is returned

4. Triggered action: cause IUT to call **detachMediaRes()** method on the tester's (Application) **IpAppCallLeg** interface.
   Parameters: callLegSessionID



Test MMCC_ IpMultiMediaCallLeg _14

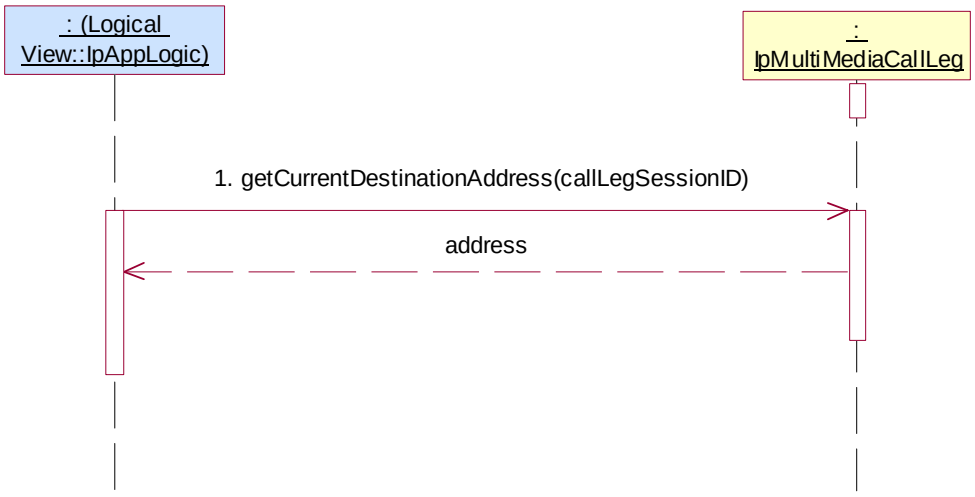
Summary: IpMultiMediaCallLeg, getCurrentDestinationAddress, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as MMCC_ IpMultiMediaCall _03

Test Sequence:

1. Method call **getCurrentDestinationAddress()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble.
   Check: valid value of TpAddress is returned



Test MMCC_ IpMultiMediaCallLeg _15

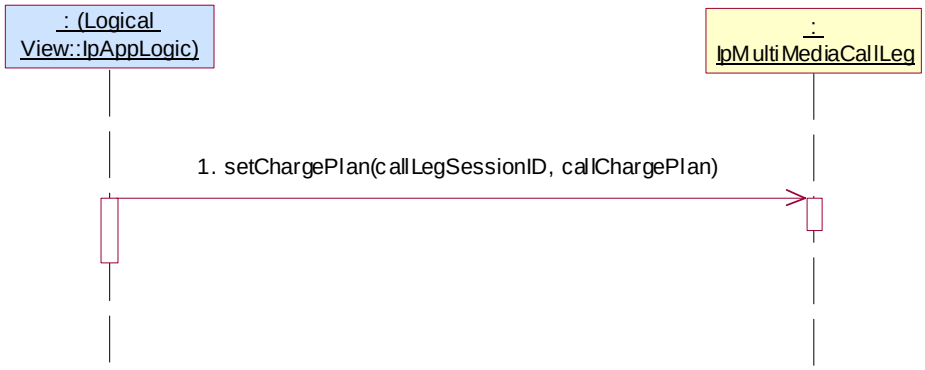
Summary: IpMultiMediaCallLeg, setChargePlan, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as MMCC_ IpMultiMediaCall _14

Test Sequence:

1. Method call **setChargePlan()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid callChargePlan
   Check: no exception is returned



Test MMCC_ IpMultiMediaCallLeg _16

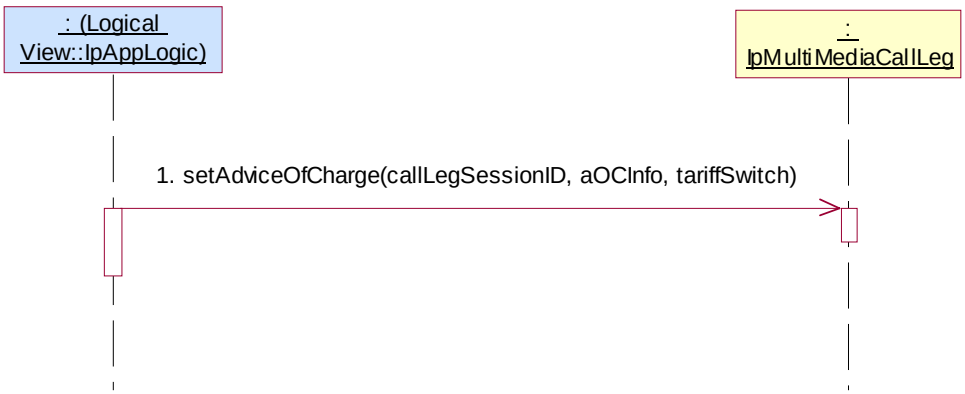
Summary: IpMultiMediaCallLeg, setAdviceOfCharge, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as MMCC_ IpMultiMediaCall _14

Test Sequence:

1. Method call **setAdviceOfCharge()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid aOCInfo, valid tariffSwitch
   Check: no exception is returned



Test MMCC_ IpMultiMediaCallLeg _17

Summary: IpMultiMediaCallLeg, superviseReq, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

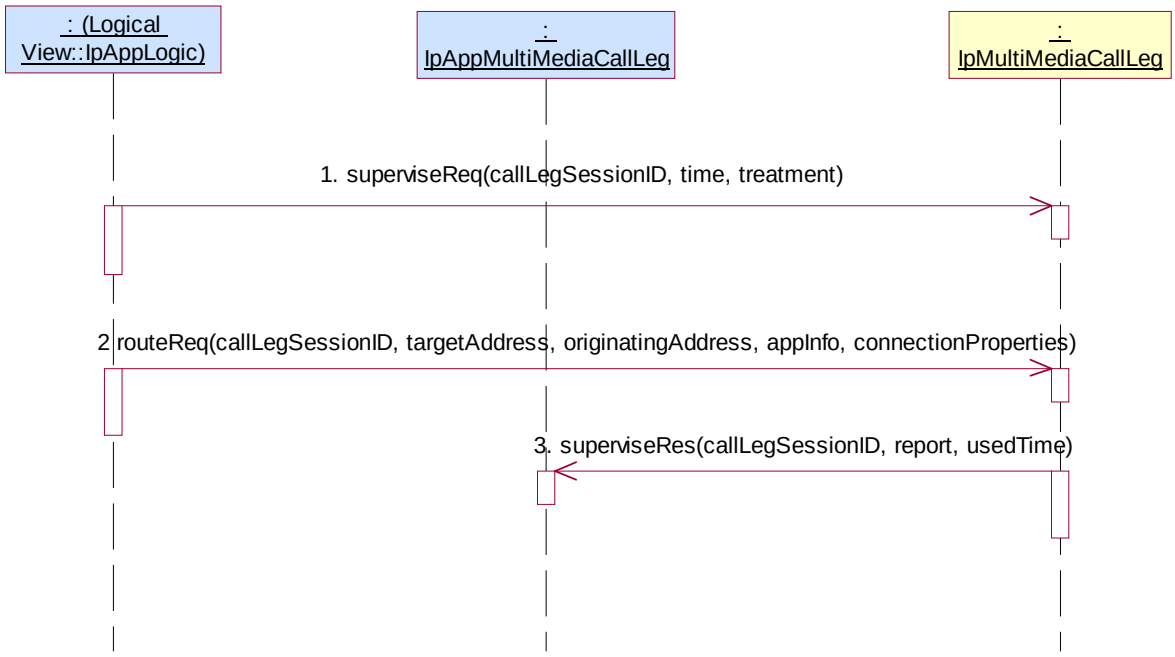
Preamble: Same as MMCC_ IpMultiMediaCall _14

Test Sequence:

1. Method call **superviseReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid time, valid treatment
   Check: no exception is returned

2. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

3. Triggered action: cause IUT to call **superviseRes()** method on the tester's (Application) **IpAppCallLeg** interface.
   Parameters: callLegSessionID, report, usedTime



Test MMCC_ IpMultiMediaCallLeg _18

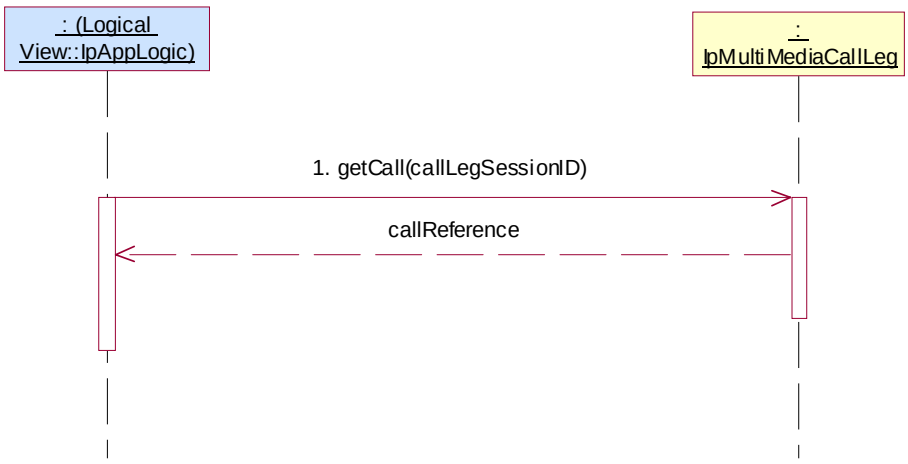
Summary: IpMultiMediaCallLeg, getCall, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as MMCC_ IpMultiMediaCall _14

Test Sequence:

1. Method call **getCall()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble.
   Check: valid TpMultiPartyCallIdentifier is returned



Test MMCC_ IpMultiMediaCallLeg _19

Summary: IpMultiMediaCallLeg, getCallEnded, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

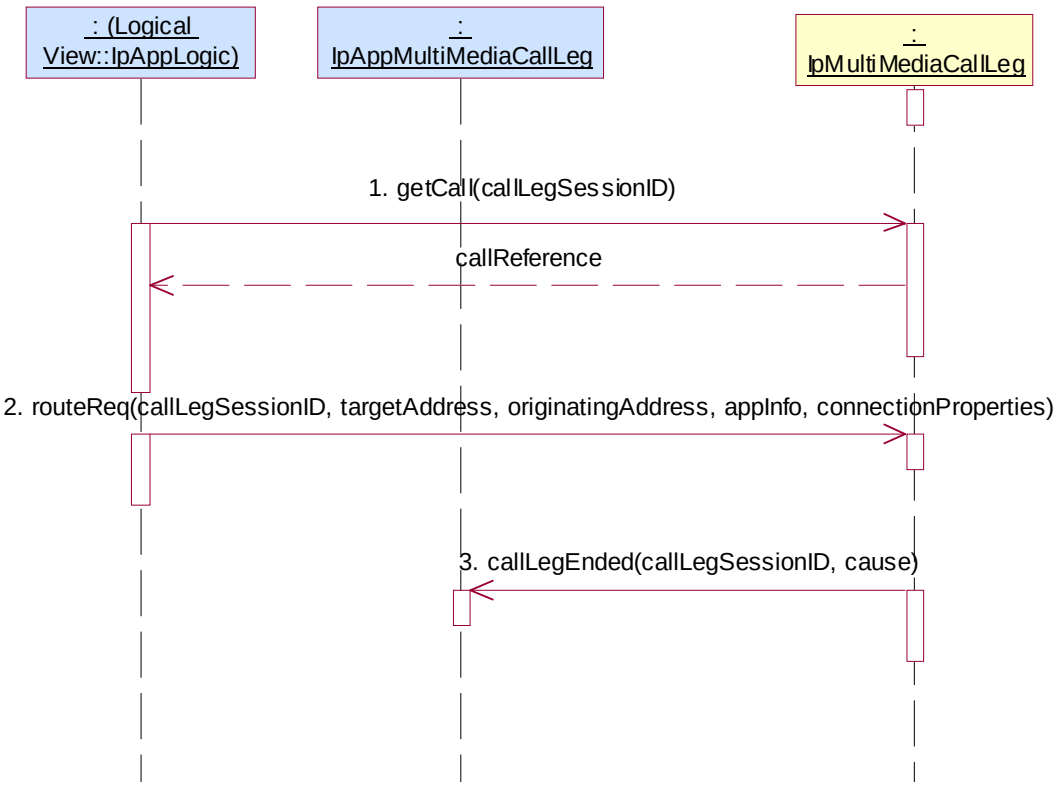
Preamble: Same as MMCC_ IpMultiMediaCall _14

Test Sequence:

1. Method call **getCall()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble.
   Check: valid TpMultiPartyCallIdentifier is returned

2. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

3 Triggered action: cause IUT to call **callLegEnded()** method on the tester's (Application) **IpAppCallLeg** interface.
   Parameters: callLegSessionID, cause



5.2.3.3.4 Optional, invalid behaviour

Test MMCC_ IpMultiMediaCallLeg _20

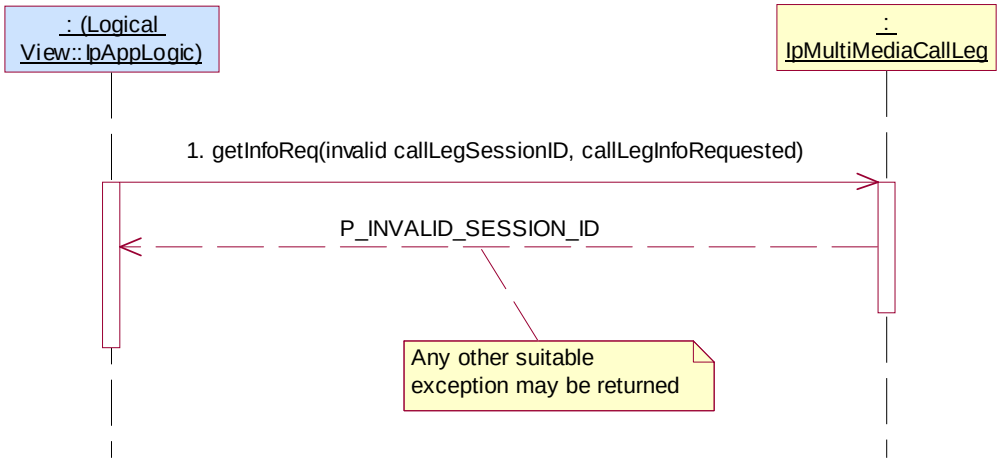
Summary: IpMultiMediaCallLeg, getInfoReq, : P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.5.

Preamble: Same as MMCC_ IpMultiMediaCall _14

Test Sequence:

1. Method call **getInfoReq()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID, valid callLegInfoRequested
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCallLeg _21

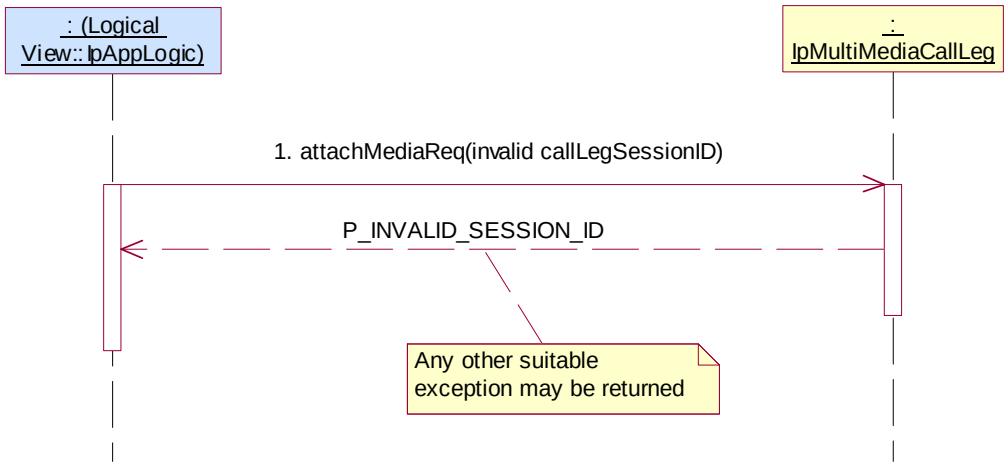
Summary: IpMultiMediaCallLeg, attachMediaReq: P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.5.

Preamble: Same as MMCC_ IpMultiMediaCallLeg _12

Test Sequence:

1. Method call **attachMediaReq()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCallLeg _22

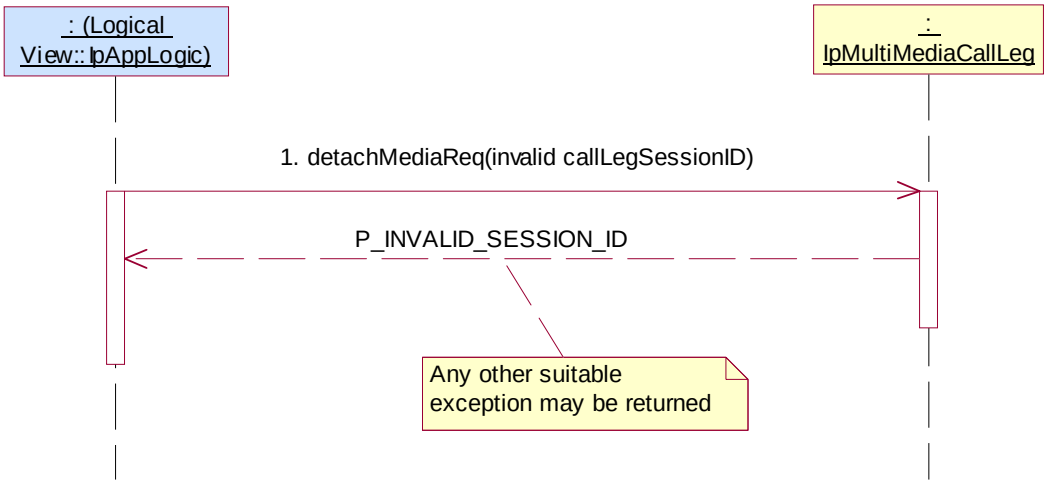
Summary: IpMultiMediaCallLeg, detachMediaReq, : P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.5.

Preamble: Same as MMCC_ IpMultiMediaCall _03

Test Sequence:

1. Method call **detachMediaReq()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCallLeg _23

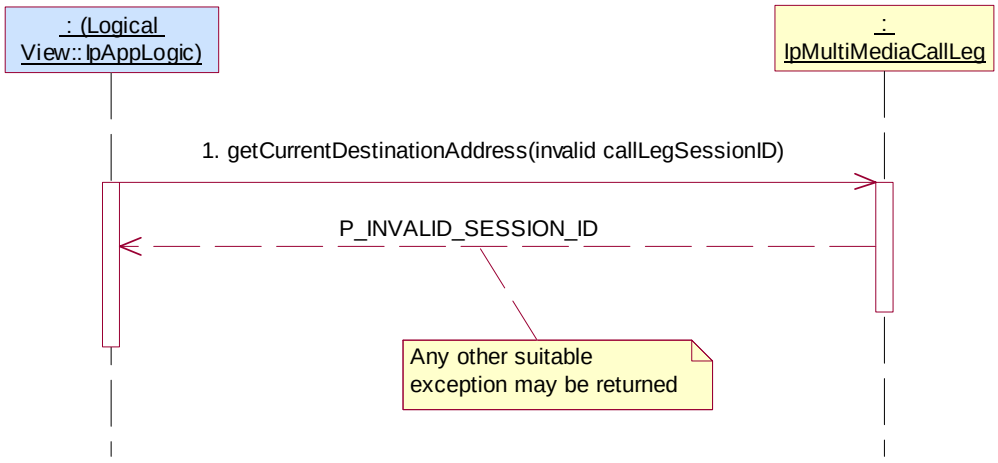
Summary: IpMultiMediaCallLeg, getCurrentDestinationAddress, : P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.5.

Preamble: Same as MMCC_ IpMultiMediaCall _03

Test Sequence:

1. Method call **getCurrentDestinationAddress()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCallLeg _24

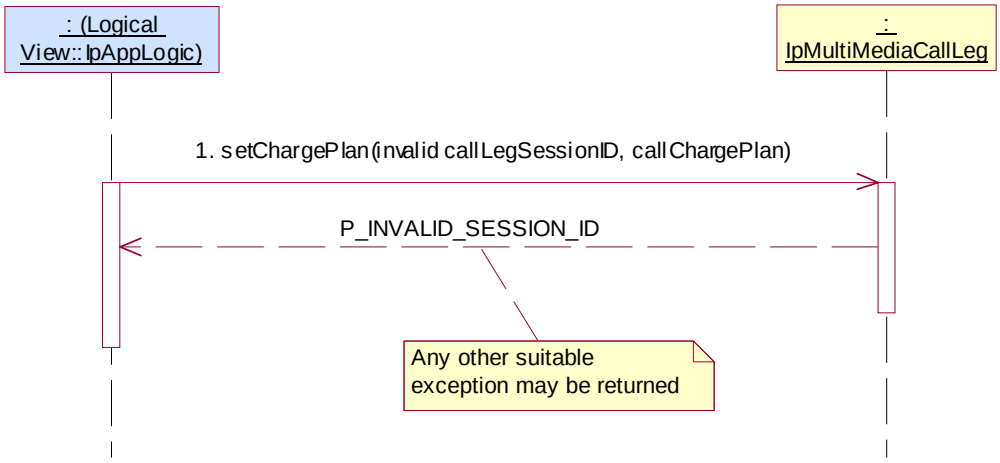
Summary: IpMultiMediaCallLeg, setChargePlan: P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.5

Preamble: Same as MMCC_ IpMultiMediaCall _14

Test Sequence:

1. Method call **setChargePlan()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID, valid callChargePlan
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCallLeg _25

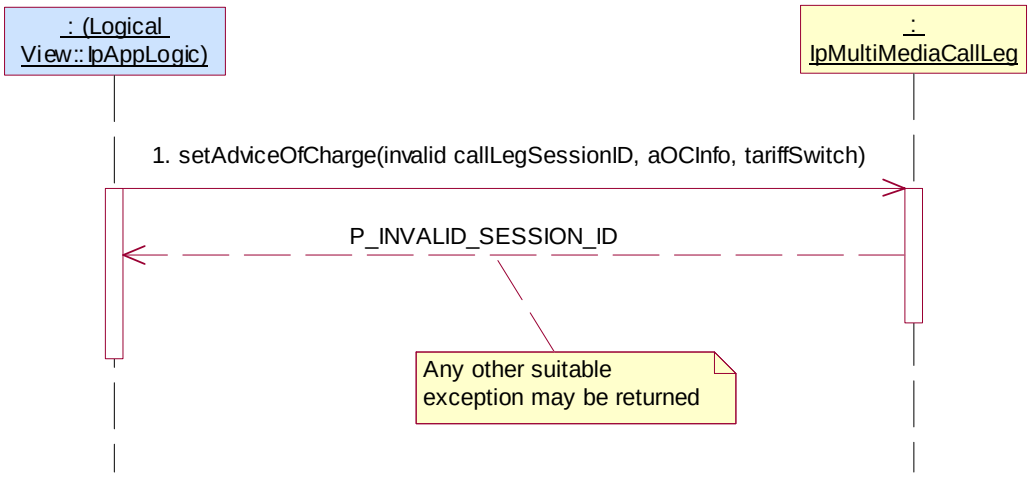
Summary: IpMultiMediaCallLeg, setAdviceOfCharge, : P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.5

Preamble: Same as MMCC_ IpMultiMediaCall _14

Test Sequence:

1. Method call **setAdviceOfCharge()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID, valid aOCInfo, valid tariffSwitch
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCallLeg _26

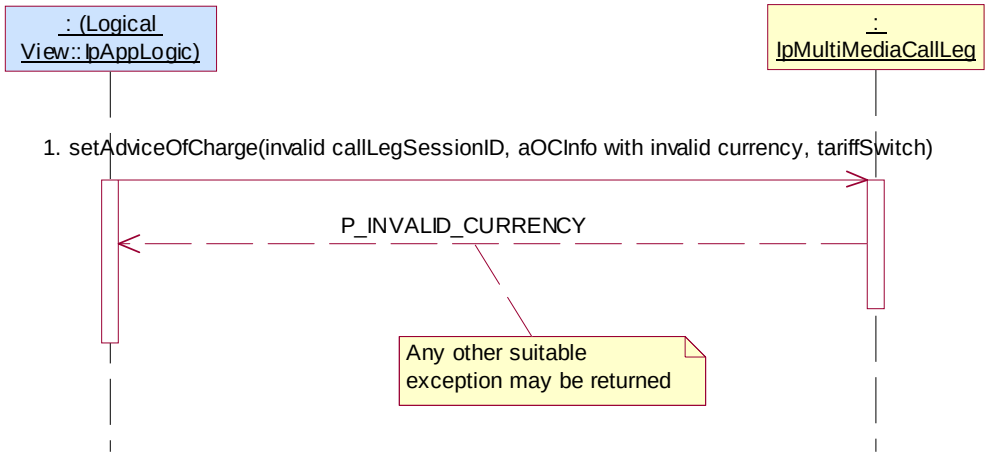
Summary: IpMultiMediaCallLeg, setAdviceOfCharge, P_INVALID_CURRENCY

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as MMCC_ IpMultiMediaCall _14

Test Sequence:

1. Method call **setAdviceOfCharge()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, aOCInfo with invalid currency, valid tariffSwitch
   Check: P_INVALID_CURRENCY, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCallLeg _27

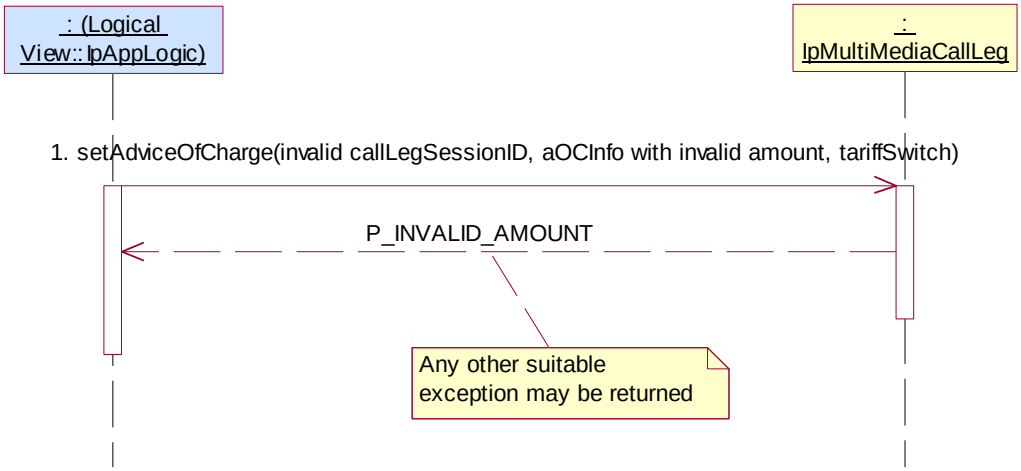
Summary: IpMultiMediaCallLeg, setAdviceOfCharge, P_INVALID_AMOUNT

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as MMCC_ IpMultiMediaCall _14

Test Sequence:

1. Method call **setAdviceOfCharge()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, aOCInfo with invalid amount, valid tariffSwitch
   Check: P_INVALID_AMOUNT, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCallLeg _28

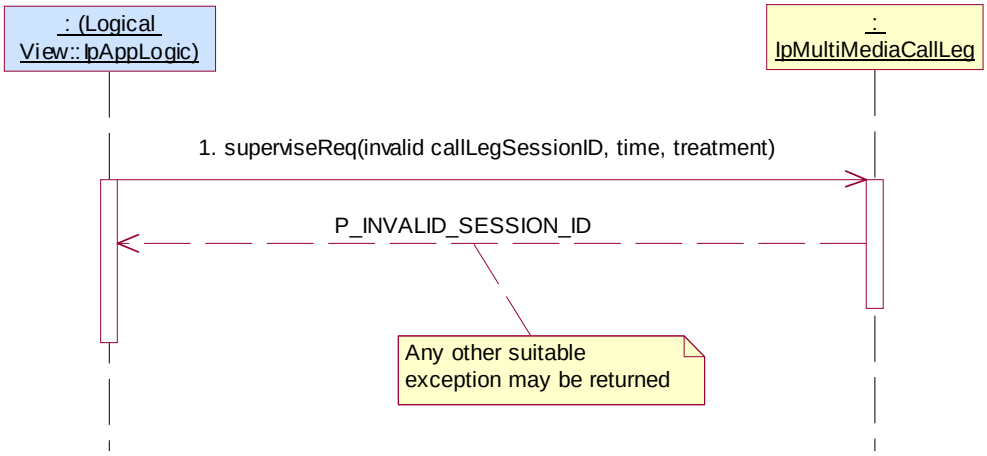
Summary: IpMultiMediaCallLeg, superviseReq, : P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.5

Preamble: Same as MMCC_ IpMultiMediaCall _14

Test Sequence:

1. Method call **superviseReq()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID, valid time, valid treatment
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCallLeg _29

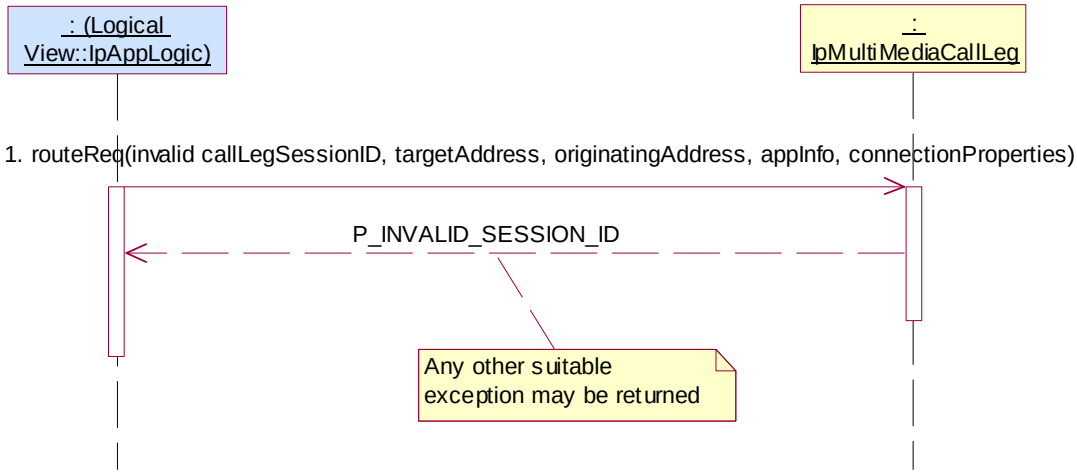
Summary: IpMultiMediaCallLeg, routeReq, : P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.5

Preamble: Same as MMCC_ IpMultiMediaCall _14

Test Sequence:

1. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID, valid targetAddress, valid originatingAddress, valid appInfo, valid connectionProperties
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCallLeg _30

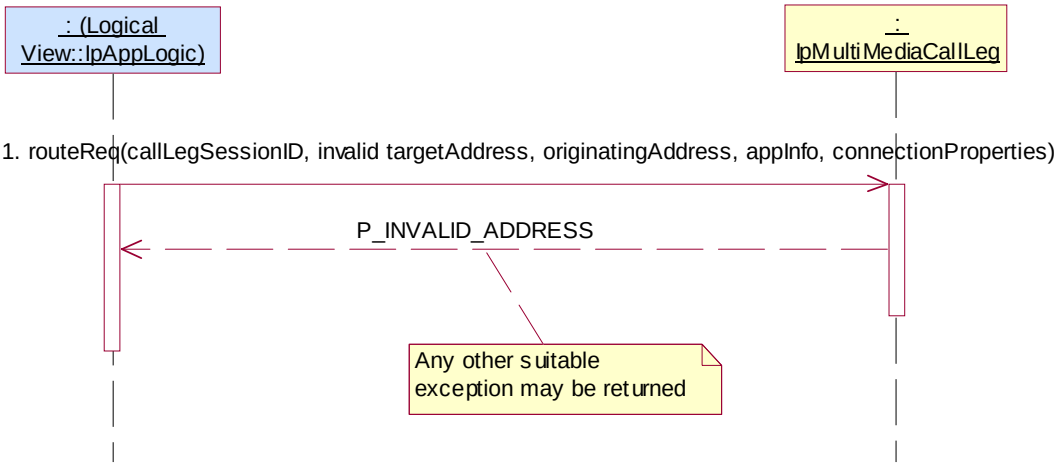
Summary: IpMultiMediaCallLeg, setAdviceOfCharge, P_INVALID_ADDRESS

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as MMCC_ IpMultiMediaCall _14

Test Sequence:

1. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, invalid targetAddress, valid originatingAddress, valid appInfo, valid connectionProperties
   Check: P_INVALID_ADDRESS, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCallLeg _31

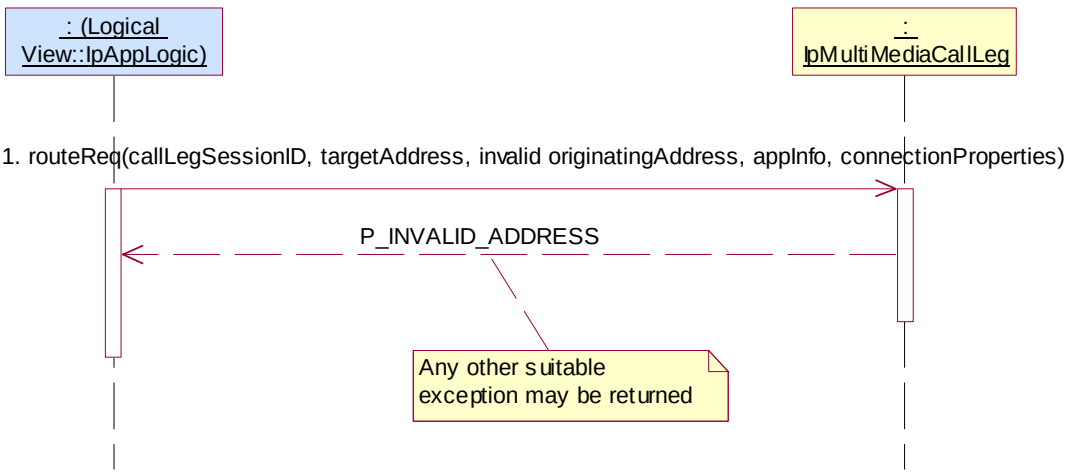
Summary: IpMultiMediaCallLeg, setAdviceOfCharge, P_INVALID_ADDRESS

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as MMCC_ IpMultiMediaCall _14

Test Sequence:

1. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid targetAddress, invalid originatingAddress, valid appInfo, valid connectionProperties
   Check: P_INVALID_ADDRESS, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCallLeg _32

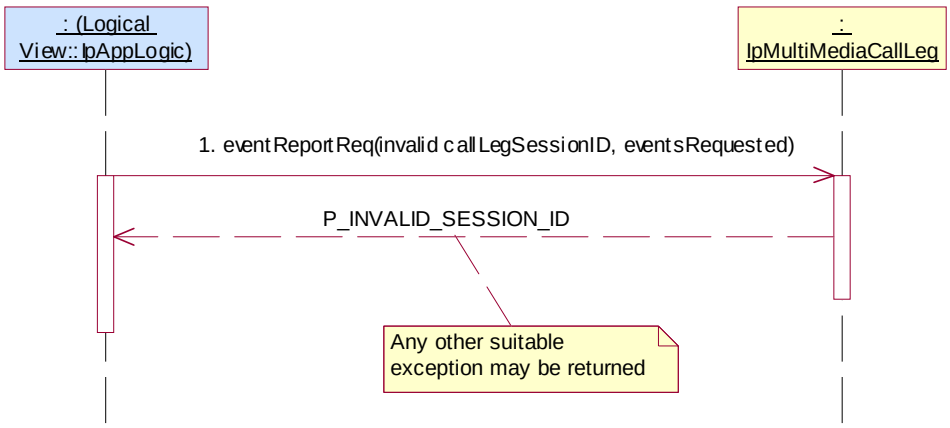
Summary: IpMultiMediaCallLeg, eventReportReq**,** P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.5

Preamble: Same as MMCC_ IpMultiMediaCall _14

Test Sequence:

1. Method call **eventReportReq()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID, valid eventsRequested
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCallLeg _33

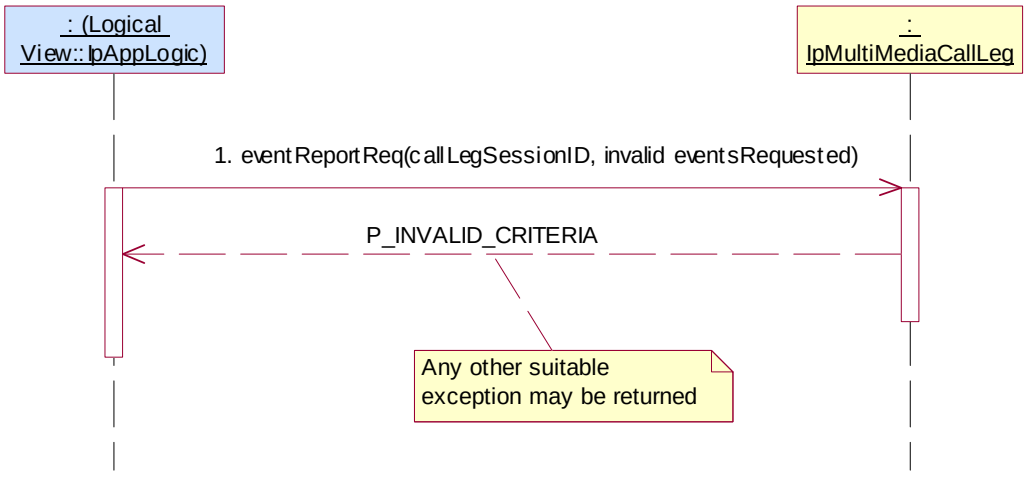
Summary: IpMultiMediaCallLeg, eventReportReq, P_INVALID_CRITERIA

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as MMCC_ IpMultiMediaCall _14

Test Sequence:

1. Method call **eventReportReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, invalid eventsRequested
   Check: P_INVALID_CRITERIA, or another suitable exception, is returned



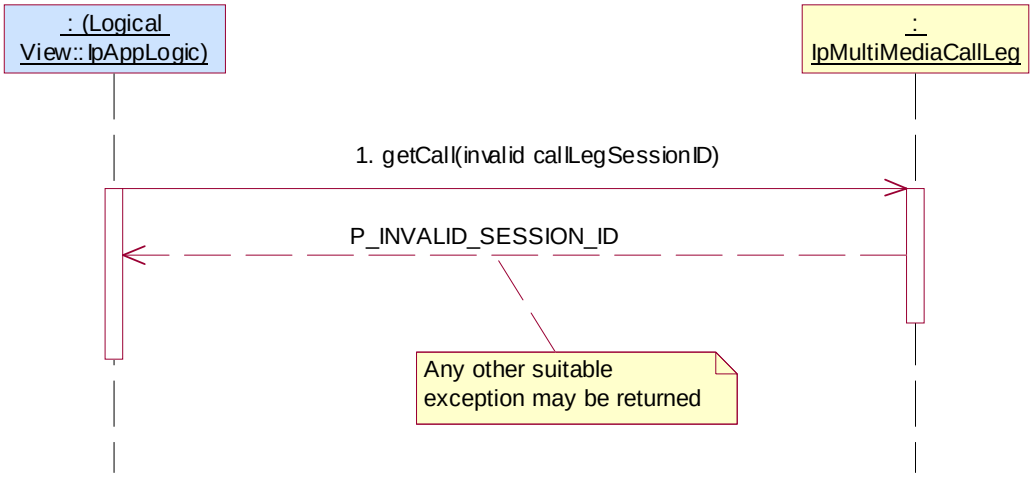
Test MMCC_ IpMultiMediaCallLeg _34

Summary: IpMultiMediaCallLeg, getCall**,** P_INVALID_SESSION_ID

Preamble: Same as MMCC_ IpMultiMediaCall _03

Reference: ES 201 915-4 [1], clause 7.3.5

1. Method call **getCall()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test MMCC_ IpMultiMediaCallLeg _35

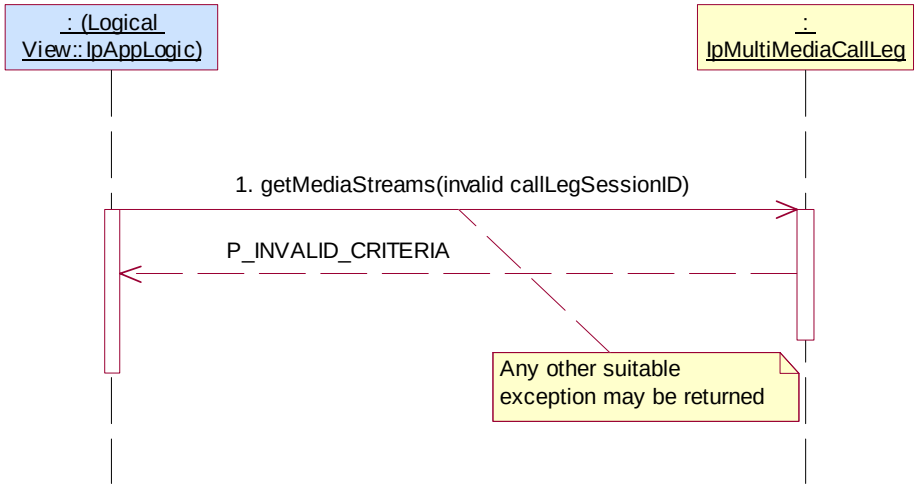
Summary: IpMultiMediaCallLeg, getMediaStreams, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 8.3.5.

Preamble: Same as MMCC_ IpMultiMediaCall _03

Test Sequence:

1. Method call **getMediaStreams()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



5.2.3.4 IpMultiMediaStream

5.2.3.4.1 Mandatory, valid behaviour

Test MMCC_ IpMultiMediaStream _01

Summary: IpMultiMediaStream, all methods mandatory, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3, 8.3.5 and 8.3.7.

Preamble: Application has a valid callSessionID returned by one of the three following sequence:

1. Method call **setCallback()** on IpMultiMediaCallControlManager
   Parameters: valid, non-null, value of appInterface parameter
   Check: no exception is returned

2. Method call **createCall()**
   Parameters: valid appCall
   Check: valid value of TpMultiMediaCallIdentifier is returned

3. Method call **createCallLeg()** on IpMultiMediaCall
   Parameters: valid callSessionID returned in 2., valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

4. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 2, valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

or

3. Method call **createAndRouteCallLegReq()** on IpMultiMediaCall
   Parameters: valid callSessionID returned in 2., valid eventsRequested, valid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: valid value of TpCallLegIdentifier

5. Method call **mediaStreamMonitorReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 3., valid mediaStreamEventCriteria
   Check: no exception is returned

6. Triggered action: cause IUT to call Method **mediaStreamMonitorRes()** method on the tester's (application)
   Parameters: callLegSessionID, streams, type

7. Method call **mediaStreamAllow()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 3., valid mediaStreamList
   Check: no exception is returned

or

1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, non-null, value, valid notificationRequest
   Check: valid value of TpAssignmentID is returned

2. Triggered action: cause IUT to call Method **reportNotification()** method on the tester's (application)
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID

3. Method call **createCallLeg()** on IpMultiMediaCall
   Parameters: valid callSessionID reported in 2., valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

4. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 2, valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

or

3. Method call **createAndRouteCallLegReq()** on IpMultiMediaCall
   Parameters: valid callSessionID returned in 2., valid eventsRequested, valid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: valid value of TpCallLegIdentifier

5. Method call **mediaStreamMonitorReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 3., valid mediaStreamEventCriteria
   Check: no exception is returned

6. Triggered action: cause IUT to call Method **mediaStreamMonitorRes()** method on the tester's (application)
   Parameters: callLegSessionID, streams, type

7. Method call **mediaStreamAllow()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 3., valid mediaStreamList
   Check: no exception is returned

or

1. Method call **createMediaNotification()**
   Parameters: valid appInterface, valid notificationMediaRequest
   Check: valid value of TpAssignmentID is returned

2. Triggered action: cause IUT to call Method **reportMediaNotification()** method on the tester's (application)
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID

3. Method call **createCallLeg()** on IpMultiMediaCall
   Parameters: valid callSessionID reported in 2., valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

4. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 2, valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

or

3. Method call **createAndRouteCallLegReq()** on IpMultiMediaCall
   Parameters: valid callSessionID returned in 2., valid eventsRequested, valid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: valid value of TpCallLegIdentifier

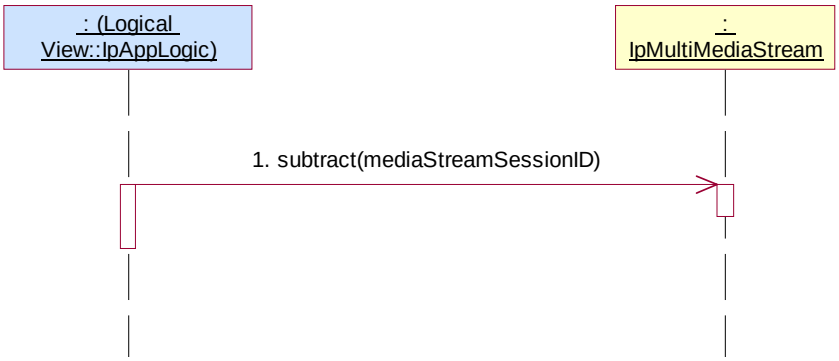
5. Method call **mediaStreamMonitorReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 3., valid mediaStreamEventCriteria
   Check: no exception is returned

6. Triggered action: cause IUT to call Method **mediaStreamMonitorRes()** method on the tester's (application)
   Parameters: callLegSessionID, streams, type

7. Method call **mediaStreamAllow()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 3., valid mediaStreamList
   Check: no exception is returned

Test Sequence:

1. Method call **subtract()** on IpMultiMediaStream
   Parameters: valid mediaStreamSessionID from TpMediaStreamSet returned in preamble.
   Check: no exception is returned



5.2.3.4.2 Mandatory, invalid behaviour

Test MMCC_ IpMultiMediaStream _02

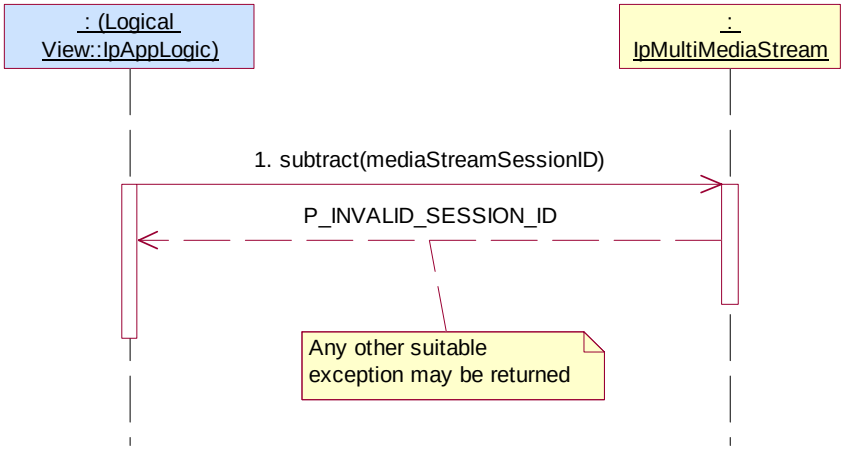
Summary: IpMultiMediaStream, subtract, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 8.3.7.

Preamble: Same as MMCC_ IpMultiMediaStream _01

Test Sequence:

1. Method call **subtract()** on IpMultiMediaStream
   Parameters: invalid mediaStreamSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



5.2.4 Conference Call Control Service (CCC)

5.2.4.1 IpConfCallControlManager

5.2.4.1.1 Mandatory, valid behaviour

According to Call Control SCF specification, at least one of the two following test sequence is mandatory:

Test CCC _ IpConfCallControlManager _01

Summary: IpConfCallControlManager, all mandatory methods, successful

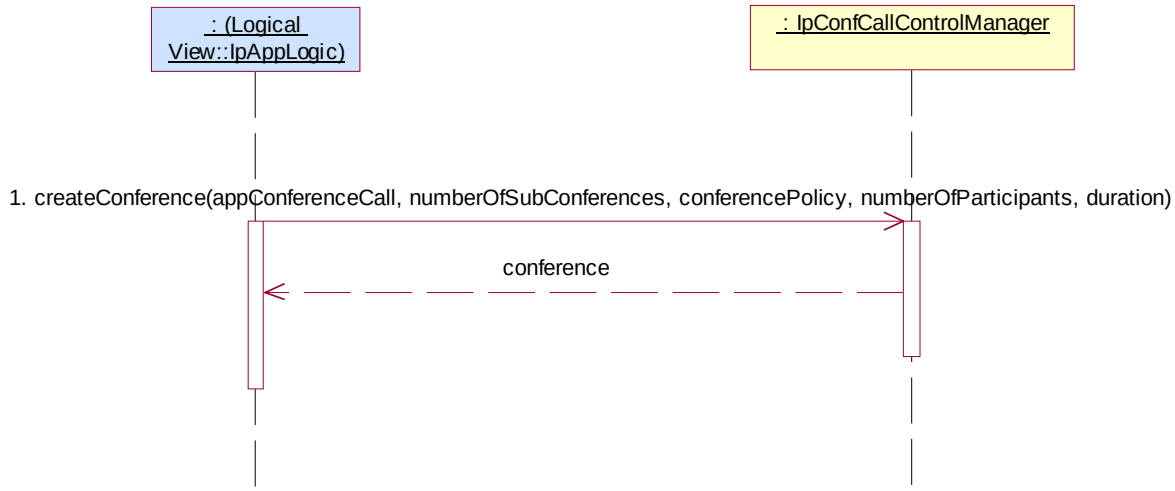
Reference: ES 201 915-4 [1], clause 9.3.1

Preamble: Application has a reference interface used for callbacks.

Condition: createConference method is supported.

Test Sequence:

1. Method call **createConference()**
   Parameters: valid appConferenceCall, valid numberOfSubConferences, valid conferencePolicy, valid numberOfParticipants, valid duration
   Check: valid value of TpConfCallIdentifier is returned



Test CCC _ IpConfCallControlManager _02

Summary: IpConfCallControlManager, all mandatory methods, successful

Reference: ES 201 915-4 [1], clause 9.3.1

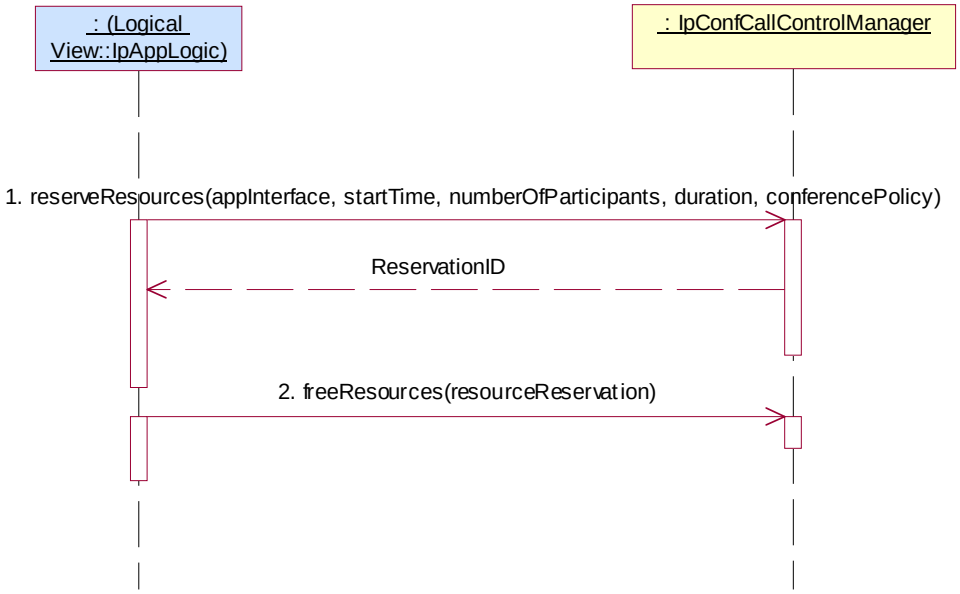
Preamble: Application has a reference interface used for callbacks.

Condition: reserveResources method is supported

Test Sequence:

1. Method call **reserveResources()**
   Parameters: valid appInterface, valid startTime, valid numberOfParticipants, valid duration, valid conferencePolicy
   Check: valid value of TpResourceReservation is returned

2. Method call **freeResources()**
   Parameters: valid resourceReservation returned in 1.
   Check: no exception is returned



Test CCC _ IpConfCallControlManager _03

Summary: IpConfCallControlManager, all mandatory methods, successful

Reference: ES 201 915-4 [1], clause 9.3.1

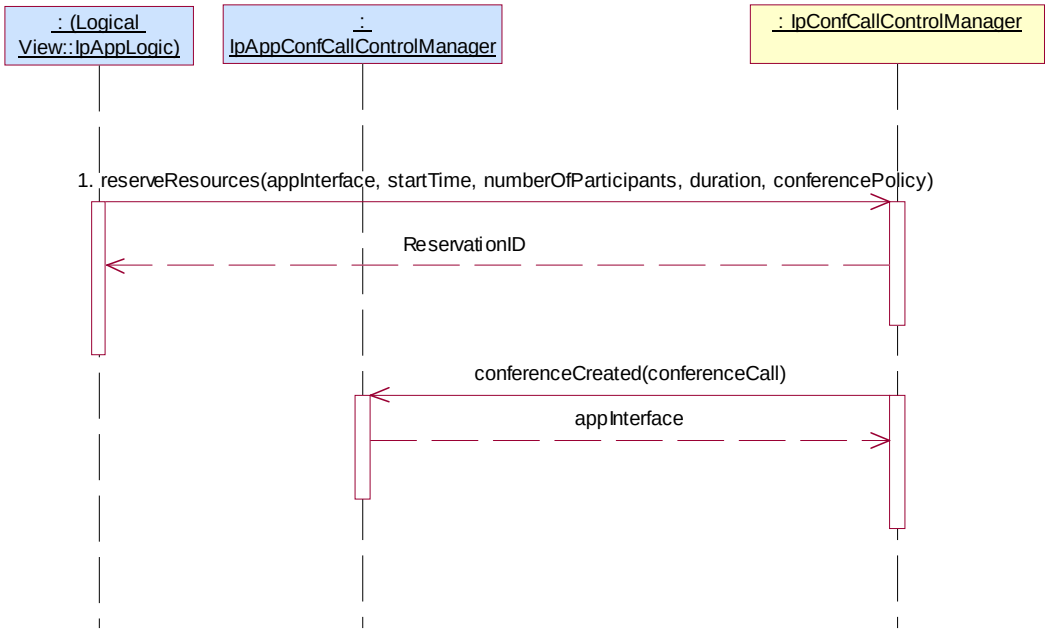
Preamble: Application has a reference interface used for callbacks.

Condition: reserveResources method is supported

Test Sequence:

1. Method call **reserveResources()**
   Parameters: valid appInterface, valid startTime, valid numberOfParticipants, valid duration, valid conferencePolicy
   Check: valid value of TpResourceReservation is returned

2. Trigger IUT to call **conferenceCreated()** on Tester's (application's) IpAppConfCallControlManager interface
   Parameters: valid conferenceCall.



5.2.4.1.2 Mandatory, invalid behaviour

For further study.

5.2.4.1.3 Optional, valid behaviour

Test CCC _ IpConfCallControlManager _04

Summary: IpConfCallControlManager, checkResources, successful

Reference: ES 201 915-4 [1], clause 9.3.1

Condition: reserveResources, checkResources methods are supported

Preamble: Application has a reference interface used for callbacks.

Test Sequence:

1. Method call **reserveResources()**
   Parameters: valid appInterface, valid startTime, valid numberOfParticipants, valid duration, valid conferencePolicy
   Check: valid value of TpResourceReservation is returned

2. Method call **checkResources()**
   Parameters: valid searchCriteria regarding reservation made in 1.
   Check: valid value of TpConfSearchResult is returned with values of reservation made in 1.

3. Method call **freeResources()**
   Parameters: valid resourceReservation returned in 2.
   Check: no exception is returned

Test CCC _ IpConfCallControlManager _05

Summary: IpConfCallControlManager, all methods, successful

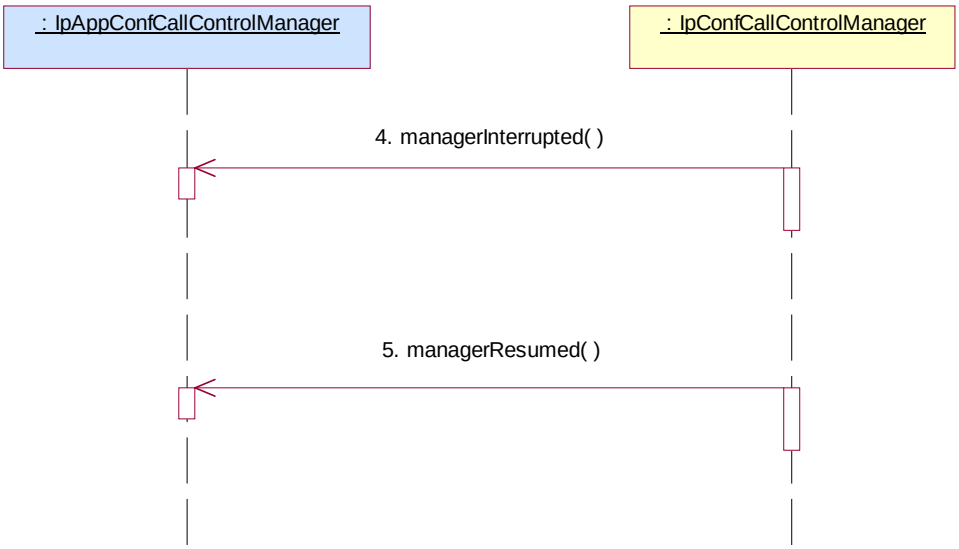
Reference: ES 201 915-4 [1], clauses 7.3.1 and 8.3.1

Condition: managerInterrupted, managerResumed methods are supported.

Test Sequence:

1. Triggered action: cause IUT to call **managerInterrupted()** method on the tester's (Application) **IpAppConfCallControlManager** interface.
   Parameters: None

2. Triggered action: cause IUT to call **managerResumed()** method on the tester's (Application) **IpAppConfCallControlManager** interface.
   Parameters: None



5.2.4.1.4 Optional, invalid behaviour

For further study.

5.2.4.2 IpConfCall

5.2.4.2.1 Mandatory, valid behaviour

Test CCC _ IpConfCall _01

Summary: IpConfCall, all methods mandatory, successful

Reference: ES 201 915-4 [1], clauses 9.3.1 and 9.3.3.

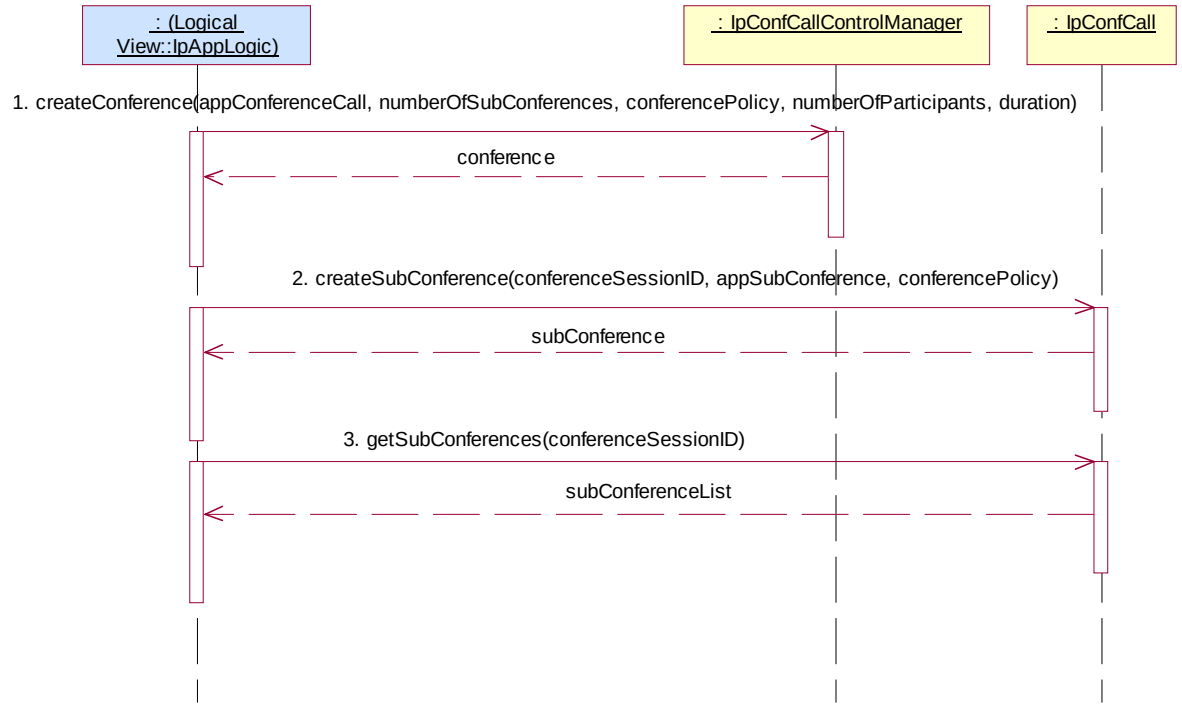
Preamble: Application has a reference interface used for callbacks.

Test Sequence:

1. Method call **createConference()** on IpConfCallControlManager
   Parameters: valid appConferenceCall, valid numberOfSubConferences, valid conferencePolicy, valid numberOfParticipants, valid duration
   Check: valid value of TpConfCallIdentifier is returned

2. Method call **createSubConference()** on IpConfCall
   Parameters: valid conferenceSessionID returned in 1., valid appSubConference, valid conferencePolicy
   Check: valid value of TpSubConfCallIdentifier is returned

3. Method call **getSubConferences()** on IpConfCall
   Parameters: valid conferenceSessionID returned in 1.
   Check: valid value of TpSubConfCallIdentifierSet is returned which contains TpSubConfCallIdentifier returned in 2.



Test CCC _ IpConfCall _02

Summary: IpConfCall, all methods mandatory, successful

Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Application has a valid callSessionID returned by one of the following sequences:

1. Method call **createConference()** on IpConfCallControlManager
   Parameters: valid appConferenceCall, valid numberOfSubConferences, valid conferencePolicy, valid numberOfParticipants, valid duration
   Check: valid value of TpConfCallIdentifier is returned

2. Method call **createSubConference()** on IpConfCall
   Parameters: valid conferenceSessionID returned in 1., valid appSubConference, valid conferencePolicy
   Check: valid value of TpSubConfCallIdentifier is returned

3. Method call **getSubConferences()** on IpConfCall
   Parameters: valid conferenceSessionID returned in 1.
   Check: valid value of TpSubConfCallIdentifierSet is returned which contains TpSubConfCallIdentifier returned in 2.

or

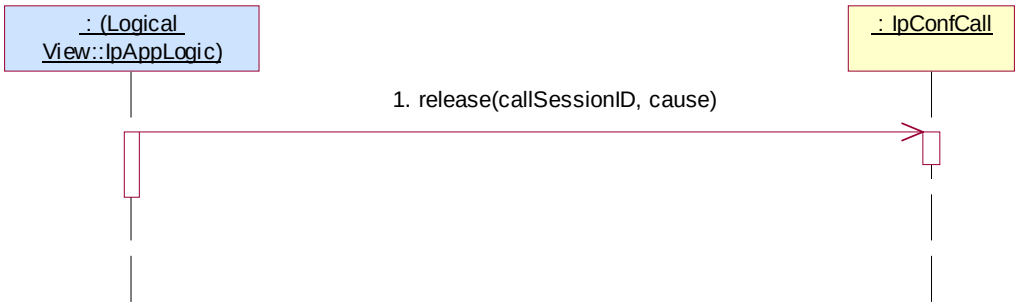
1. Method call **reserveResources()**
   Parameters: valid appInterface, valid startTime, valid numberOfParticipants, valid duration, valid conferencePolicy
   Check: valid value of TpResourceReservation is returned

2. Triggered action: cause IUT to call **conferenceCreated()** on Tester's (application's) IpAppConfCallControlManager interface
   Parameters: valid conferenceCall.

3. Method call **getSubConferences()** on IpConfCall
   Parameters: valid conferenceSessionID returned in 1.
   Check: valid value of TpSubConfCallIdentifierSet is returned which contains TpSubConfCallIdentifier returned in 2.

Test Sequence:

1. Method call **release()** on IpConfCall
   Parameters: valid callSessionID returned in preamble, valid cause
   Check: no exception is returned



Test CCC _ IpConfCall _03

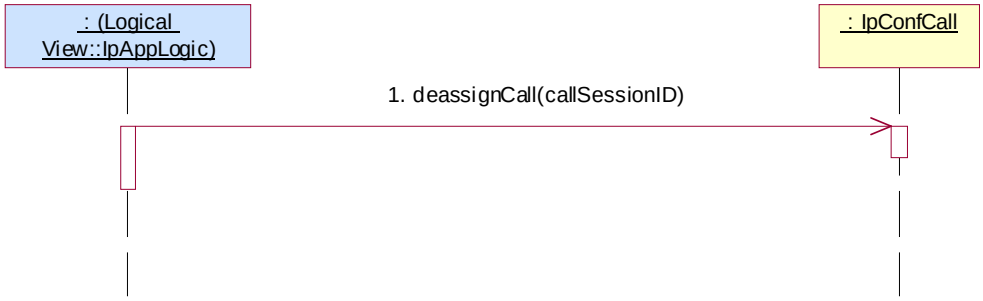
Summary: IpConfCall, all methods mandatory, successful

Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as CCC _ IpConfCall _02

Test Sequence:

1. Method call **deassignCall()** on IpConfCall
   Parameters: valid callSessionID returned in preamble.
   Check: no exception is returned



5.2.4.2.2 Mandatory, invalid behaviour

Test CCC _ IpConfCall _04

Summary: IpConfCall createSubConference, P_INVALID_SESSION_ID

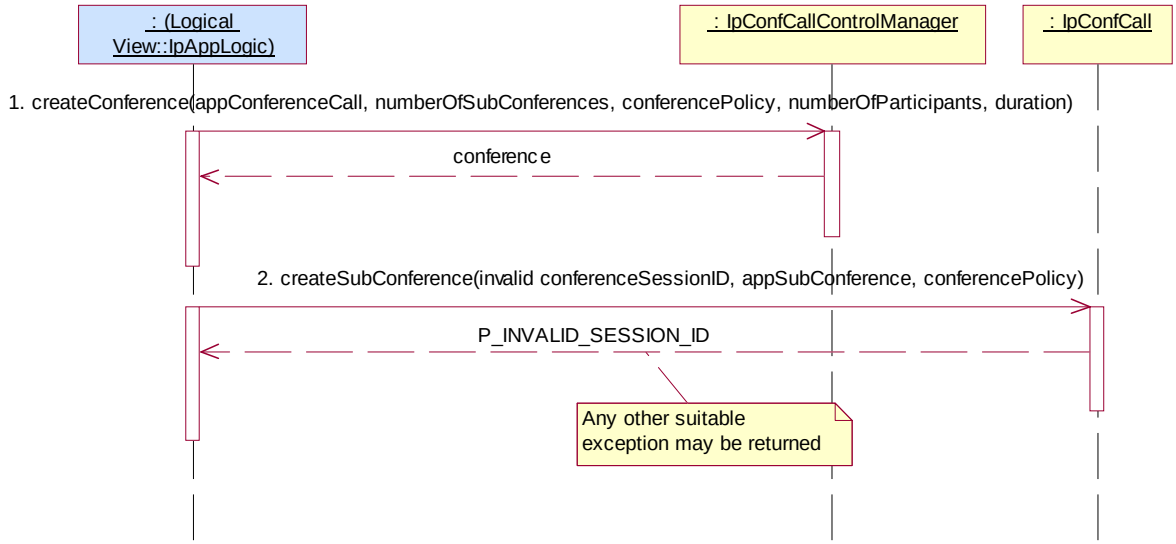
Reference: ES 201 915-4 [1], clause 9.3.3

Application has a reference interface used for callbacks.

Test Sequence:

1. Method call **createConference()** on IpConfCallControlManager
   Parameters: valid appConferenceCall, valid numberOfSubConferences, valid conferencePolicy, valid numberOfParticipants, valid duration
   Check: valid value of TpConfCallIdentifier is returned

2. Method call **createSubConference()** on IpConfCall
   Parameters: invalid conferenceSessionID, valid appSubConference, valid conferencePolicy
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpConfCall _05

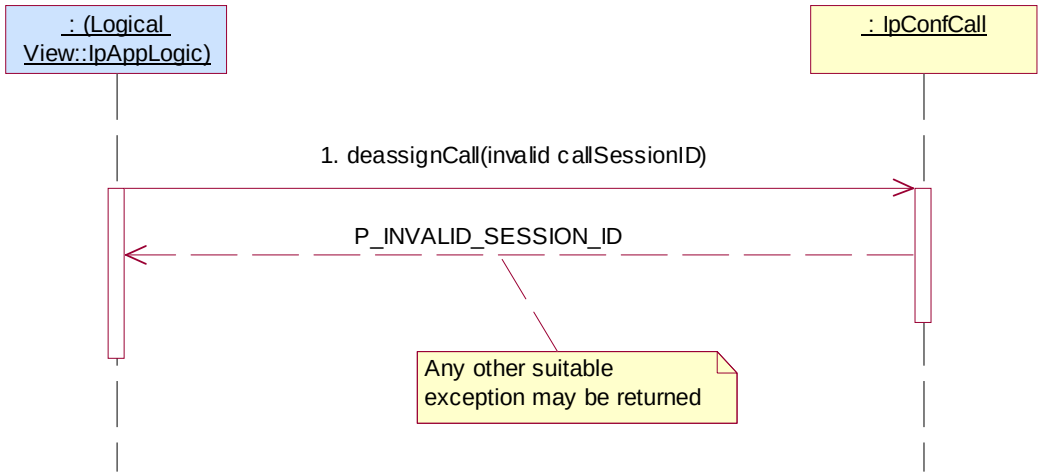
Summary: IpConfCall, deassignCall, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Same as CCC _ IpConfCall _02

Test Sequence:

1. Method call **deassignCall()** on IpConfCall
   Parameters: invalid callSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpConfCall _06

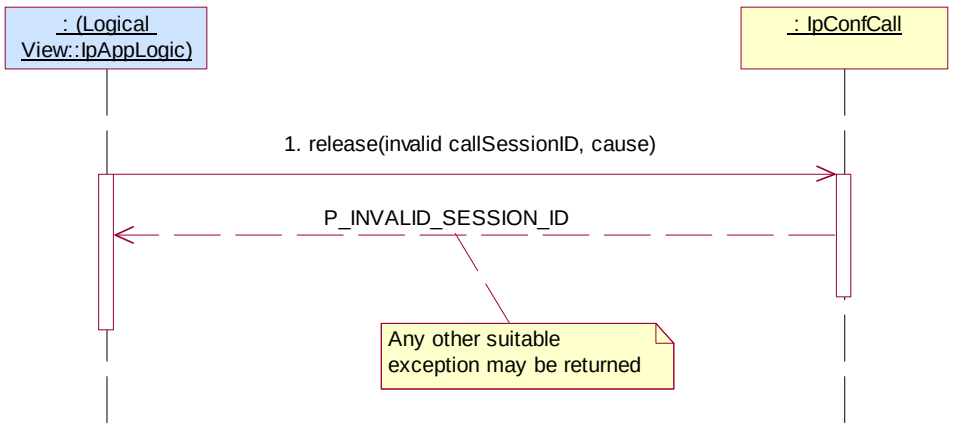
Summary: IpConfCall, release, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Same as CCC _ IpConfCall _02

Test Sequence:

1. Method call **release()** on IpConfCall
   Parameters: invalid callSessionID, valid cause
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



5.2.4.2.3 Optional, valid behaviour

Test CCC _ IpConfCall _07

Summary: IpConfCall, all methods, successful

Reference: ES 201 915-4 [1], clauses 9.3.1 and 9.3.3.

Preamble: Application has a reference interface used for callbacks.

Condition: leaveMonitorReq method is supported.

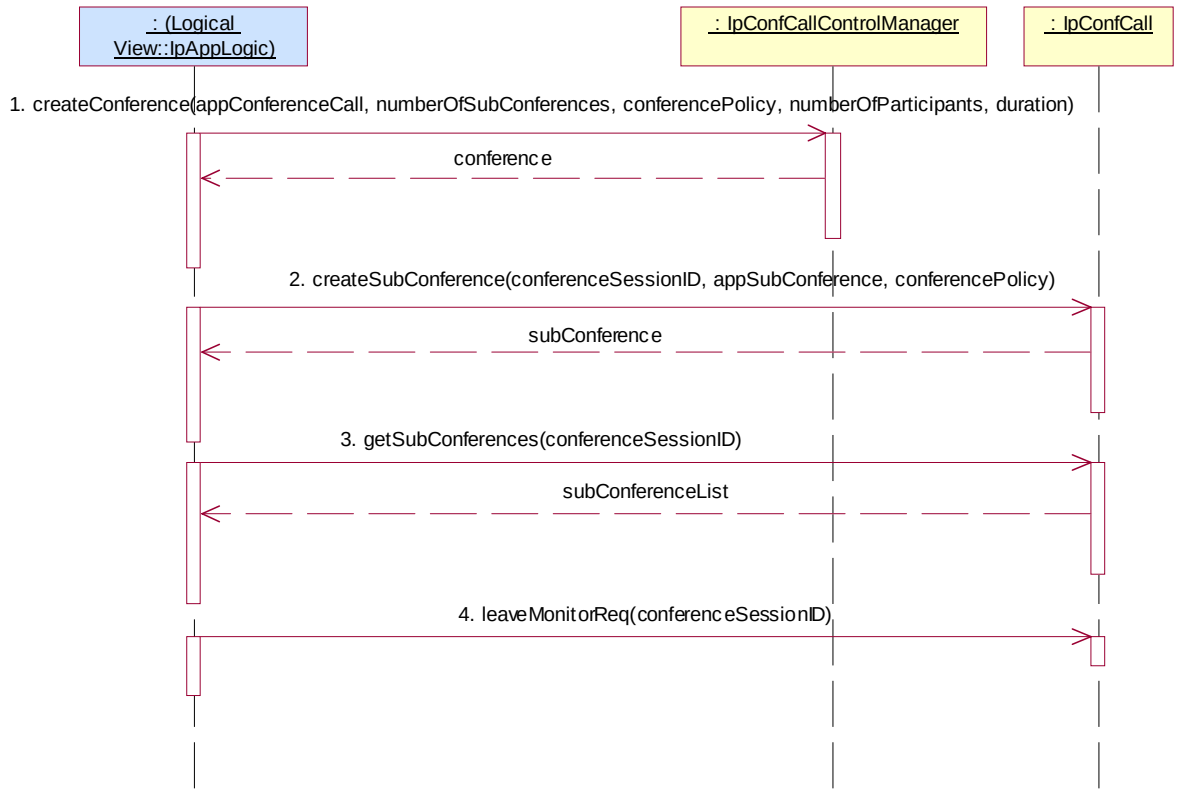
Test Sequence:

1. Method call **createConference()** on IpConfCallControlManager
   Parameters: valid appConferenceCall, valid numberOfSubConferences, valid conferencePolicy, valid numberOfParticipants, valid duration
   Check: valid value of TpConfCallIdentifier is returned

2. Method call **createSubConference()** on IpConfCall
   Parameters: valid conferenceSessionID returned in 1., valid appSubConference, valid conferencePolicy
   Check: valid value of TpSubConfCallIdentifier is returned

3. Method call **getSubConferences()** on IpConfCall
   Parameters: valid conferenceSessionID returned in 1.
   Check: valid value of TpSubConfCallIdentifierSet is returned which contains TpSubConfCallIdentifier returned in 2.

4. Method call **leaveMonitorReq()** on IpConfCall
   Parameters: valid conferenceSessionID returned in 1.
   Check: no exception is returned



Test CCC _ IpConfCall _08

Summary: IpConfCall, all methods, successful

Reference: ES 201 915-4 [1], clauses 7.3.1 and 8.3.1

Preamble: Application has a valid callSessionID returned by one of the two following sequence:

1. Method call **createConference()** on IpConfCallControlManager
   Parameters: valid appConferenceCall, valid numberOfSubConferences, valid conferencePolicy, valid numberOfParticipants, valid duration
   Check: valid value of TpConfCallIdentifier is returned

2. Method call **createSubConference()** on IpConfCall
   Parameters: valid conferenceSessionID returned in 1., valid appSubConference, valid conferencePolicy
   Check: valid value of TpSubConfCallIdentifier is returned

3. Method call **getSubConferences()** on IpConfCall
   Parameters: valid conferenceSessionID returned in 1.
   Check: valid value of TpSubConfCallIdentifierSet is returned which contains TpSubConfCallIdentifier returned in 2.

4. Method call **createCallLeg()** on IpSubConfCall
   Parameters: valid callSessionID returned in 2., valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

or

1. Method call **reserveResources()**
   Parameters: valid appInterface, valid startTime, valid numberOfParticipants, valid duration, valid conferencePolicy
   Check: valid value of TpResourceReservation is returned

2. Triggered action: cause IUT to call **conferenceCreated()** on Tester's (application's) IpAppConfCallControlManager interface
   Parameters: valid conferenceCall.

3. Method call **getSubConferences()** on IpConfCall
   Parameters: valid conferenceSessionID returned in 1.
   Check: valid value of TpSubConfCallIdentifierSet is returned which contains TpSubConfCallIdentifier returned in 2.

4. Method call **createCallLeg()** on IpSubConfCall
   Parameters: valid callSessionID reported in 2., valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

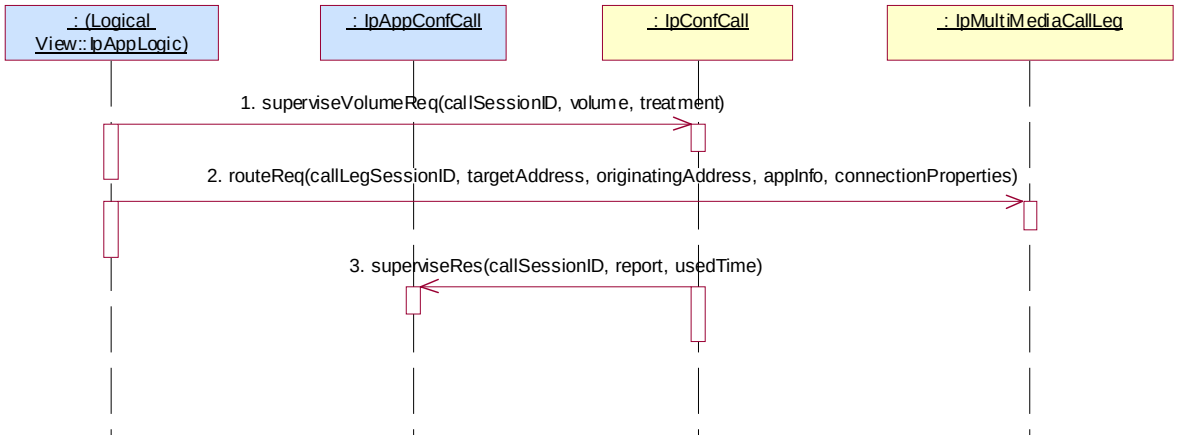
Condition: superviseVolumeReq method is supported.

Test Sequence:

1. Method call **superviseVolumeReq()** on IpConfCall
   Parameters: valid callSessionID returned in preamble, valid volume, valid treatment
   Check: no exception is returned

2. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

3. Triggered action: cause IUT to call Method **superviseVolumeRes()** method on the tester's (application) **IpConfCall** interface.
   Parameters: callSessionID, report, usedVolume


: (Logical View::IpAppLogic)
: IpAppConfCall
: IpConfCall
: IpMultiMediaCallLeg
1. superviseVolumeReq(callSessionID, volume, treatment)
2. routeReq(callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties)
3. superviseRes(callSessionID, report, usedTime)

Test CCC _ IpConfCall _09

Summary: IpConfCall, getInfoReq, successful

Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as CCC _ IpConfCall _08

Condition: createCallLeg and getInfoReq methods are supported.

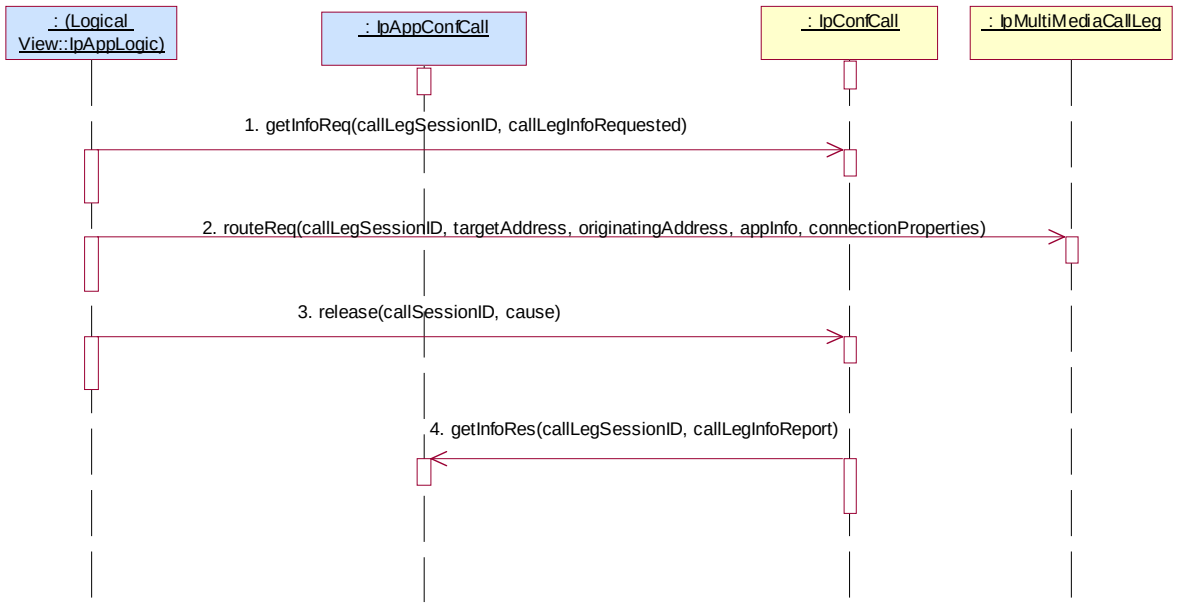
Test Sequence:

1. Method call **getInfoReq()** on IpConfCall
   Parameters: valid callSessionID returned in preamble, valid callInfoRequested
   Check: no exception is returned

2. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

4. Method call **release()** on IpConfCall
   Parameters: valid callSessionID returned in preamble, valid cause
   Check: no exception is returned

5 Triggered action: cause IUT to call **getInfoRes()** method on the tester's (Application) **IpAppConfCall** interface.
   Parameters: callSessionID given in 1., valid callInfoReport.



Test CCC _ IpConfCall _10

Summary: IpConfCall, setChargePlan, successful

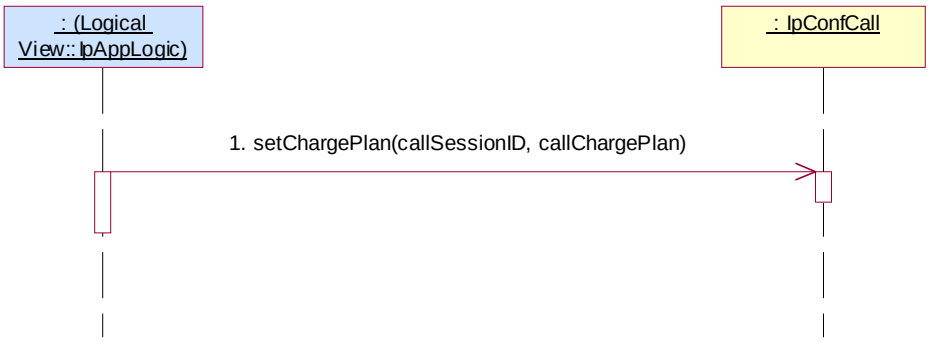
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as CCC _ IpConfCall _02

Condition: createCallLeg and setChargePlan methods are supported.

Test Sequence:

1. Method call **setChargePlan()** on IpConfCall
   Parameters: valid callSessionID returned in 1., valid callChargePlan
   Check: no exception is returned



Test CCC _ IpConfCall _11

Summary: IpConfCall, setAdviceOfCharge, successful

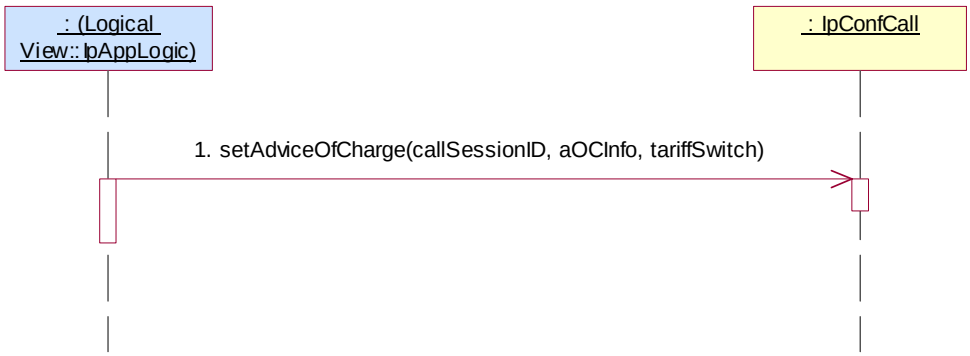
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as CCC _ IpConfCall _02

Condition: createCallLeg and setAdviceOfCharge methods are supported.

Test Sequence:

1. Method call **setAdviceOfCharge()** on IpConfCall
   Parameters: valid callSessionID returned in 1., valid aOCInfo, valid tariffSwitch
   Check: no exception is returned



Test CCC _ IpConfCall _12

Summary: IpConfCall, superviseReq, successful

Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as CCC _ IpConfCall _08

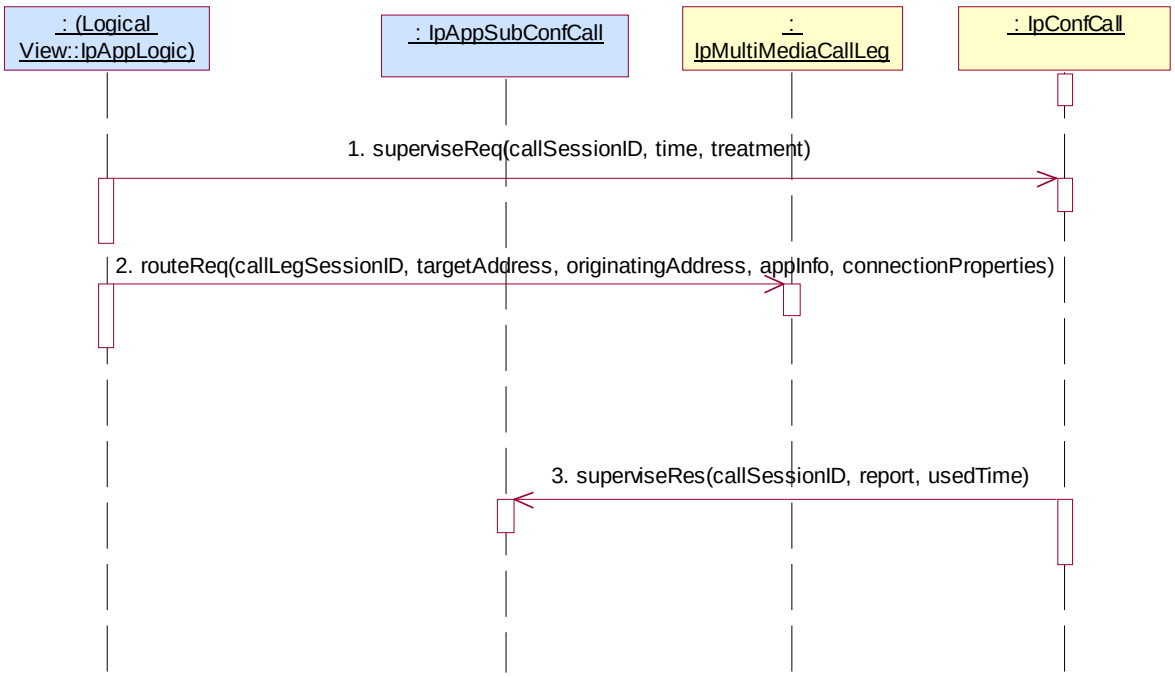
Condition: createCallLeg and superviseReq methods are supported.

Test Sequence:

1. Method call **superviseReq()** on IpConfCall
   Parameters: valid callSessionID returned in preamble, valid time, valid treatment
   Check: no exception is returned

2. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

3 Triggered action: cause IUT to call **superviseRes()** method on the tester's (Application) **IpAppConfCall** interface.
   Parameters: callSessionID given in 1., valid report, valid usedTime.



Test CCC _ IpConfCall _13

Summary: IpConfCall, all methods, successful

Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as CCC _ IpConfCall _08

Condition: getCallLegs method is supported.

Test Sequence:

1. Method call **getCallLegs()** on IpConfCall
   Parameters: valid callSessionID returned in preamble.
   Check: valid value of TpCallLegIdentifierSet which contains CallLegIdentifier returned in preamble.

5.2.4.2.4 Optional, invalid behaviour

Test CCC _ IpConfCall _14

Summary: IpConfCall, getSubConferences, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 9.3.3

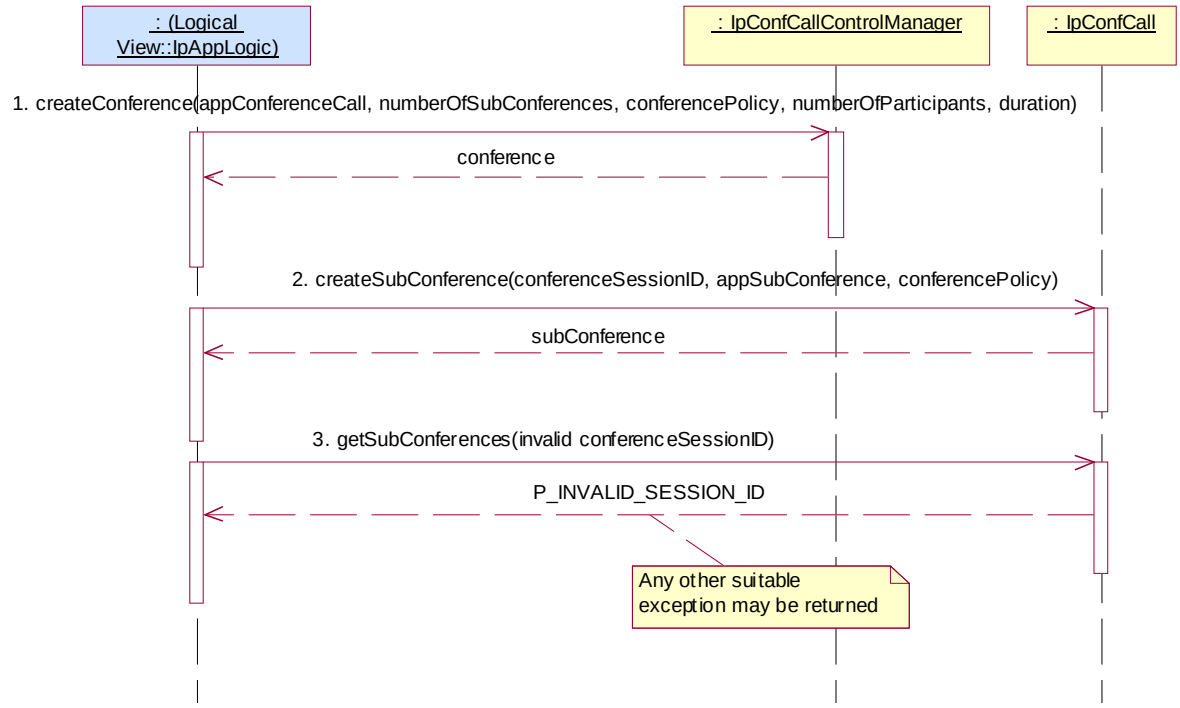
Preamble: Application has a reference interface used for callbacks.

Test Sequence:

1. Method call **createConference()** on IpConfCallControlManager
   Parameters: valid appConferenceCall, valid numberOfSubConferences, valid conferencePolicy, valid numberOfParticipants, valid duration
   Check: valid value of TpConfCallIdentifier is returned

2. Method call **createSubConference()** on IpConfCall
   Parameters: valid conferenceSessionID returned in 1., valid appSubConference, valid conferencePolicy
   Check: valid value of TpSubConfCallIdentifier is returned

3. Method call **getSubConferences()**
   Parameters: invalid conferenceSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpConfCall _15

Summary: IpConfCall, leaveMonitorReq, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 9.3.3

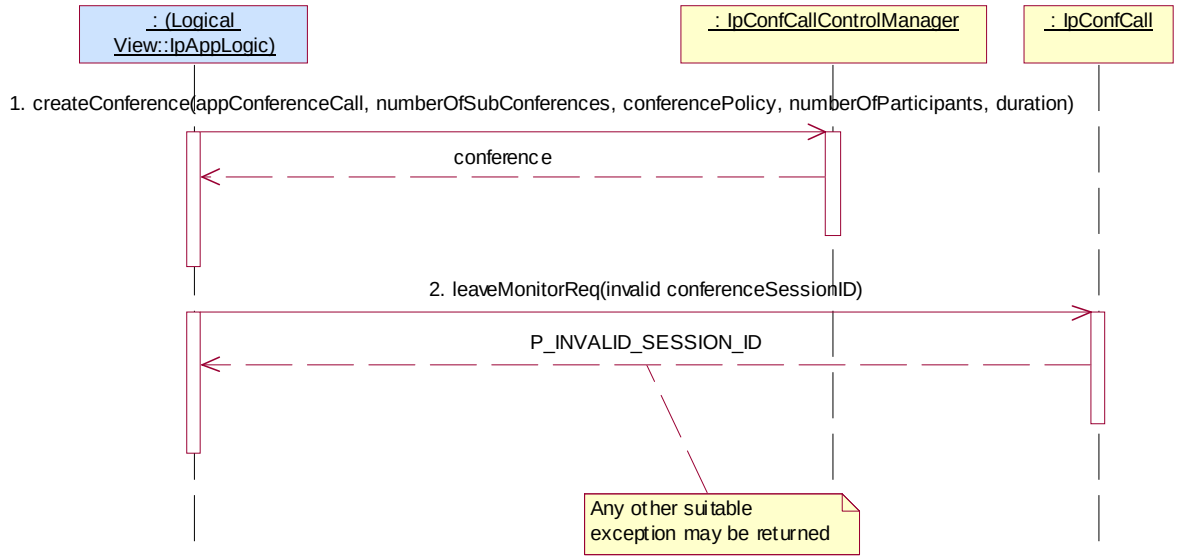
Preamble: Application has a reference interface used for callbacks.

Condition: leaveMonitorReq method is supported.

Test Sequence:

1. Method call **createConference()** on IpConfCallControlManager
   Parameters: valid appConferenceCall, valid numberOfSubConferences, valid conferencePolicy, valid numberOfParticipants, valid duration
   Check: valid value of TpConfCallIdentifier is returned

2. Method call **leaveMonitorReq()**
   Parameters: invalid conferenceSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpConfCall _16

Summary: IpConfCall, superviseVolumeReq, P_INVALID_SESSION_ID

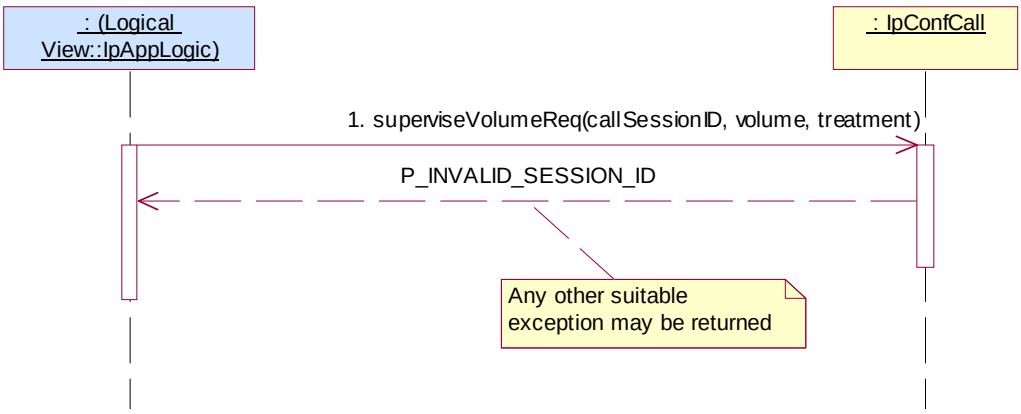
Reference: ES 201 915-4 [1], clause 8.3.1

Preamble: Same as CCC _ IpConfCall _02

Condition: superviseVolumeReq method is supported.

Test Sequence:

1. Method call **superviseVolumeReq()** on IpConfCall
   Parameters: invalid callSessionID, valid volume, valid treatment
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpConfCall _17

Summary: IpConfCall, getInfoReq, P_INVALID_SESSION_ID

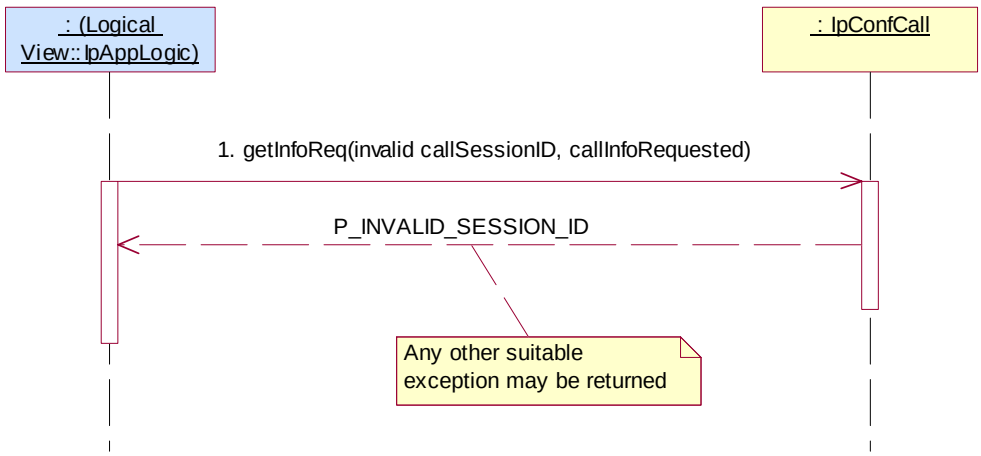
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as CCC _ IpConfCall _08

Condition: getInfoReq method is supported.

Test Sequence:

1. Method call **getInfoReq()** on IpConfCall
   Parameters: invalid callSessionID, valid callInfoRequested
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpConfCall _18

Summary: IpConfCall, setChargePlan, P_INVALID_SESSION_ID

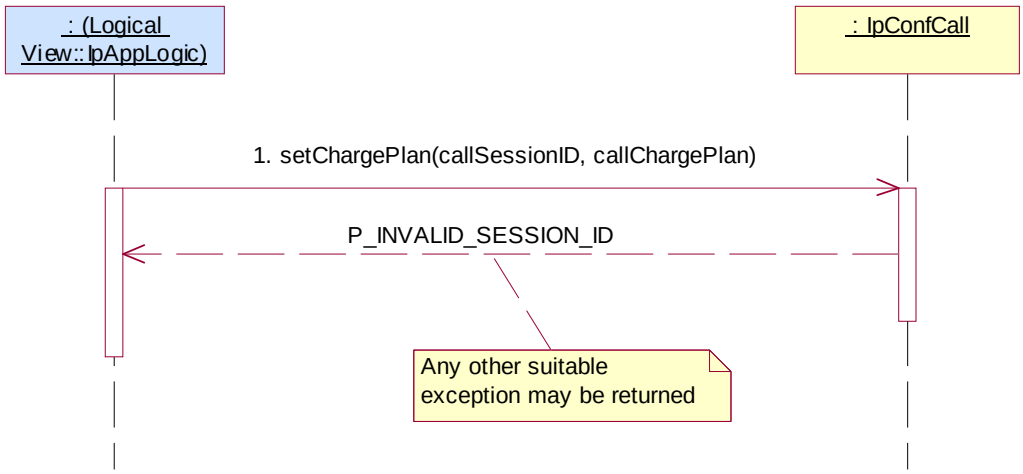
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as CCC _ IpConfCall _02

Condition: setChargePlan method is supported.

Test Sequence:

1. Method call **setChargePlan()** on IpConfCall
   Parameters: invalid callSessionID, valid callChargePlan
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpConfCall _19

Summary: IpConfCall, setAdviceOfCharge, P_INVALID_SESSION_ID

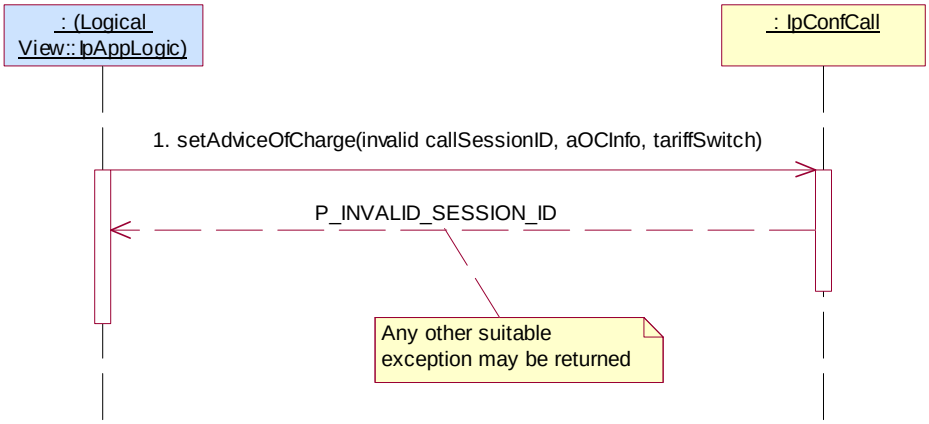
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as CCC _ IpConfCall _02

Condition: setAdviceOfCharge method is supported.

Test Sequence:

1. Method call **setAdviceOfCharge()** on IpConfCall
   Parameters: invalid callSessionID, valid aOCInfo, valid tariffSwitch
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpConfCall _20

Summary: IpConfCall, setAdviceOfCharge, P_INVALID_CURRENCY

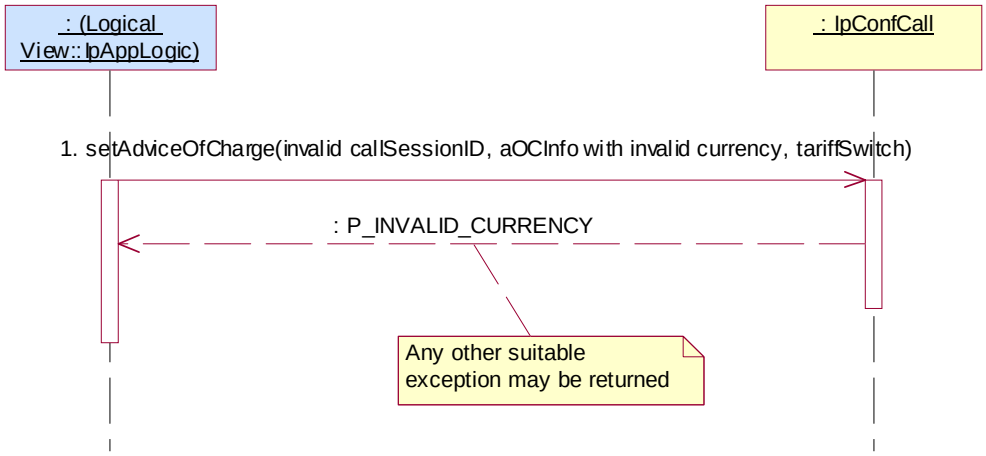
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as CCC _ IpConfCall _02

Condition: setAdviceOfCharge method is supported.

Test Sequence:

1. Method call **setAdviceOfCharge()** on IpConfCall
   Parameters: valid callSessionID returned in 1., aOCInfo with invalid currency, valid tariffSwitch
   Check: P_INVALID_CURRENCY, or another suitable exception, is returned



Test CCC _ IpConfCall _21

Summary: IpConfCall, setAdviceOfCharge, P_INVALID_AMOUNT

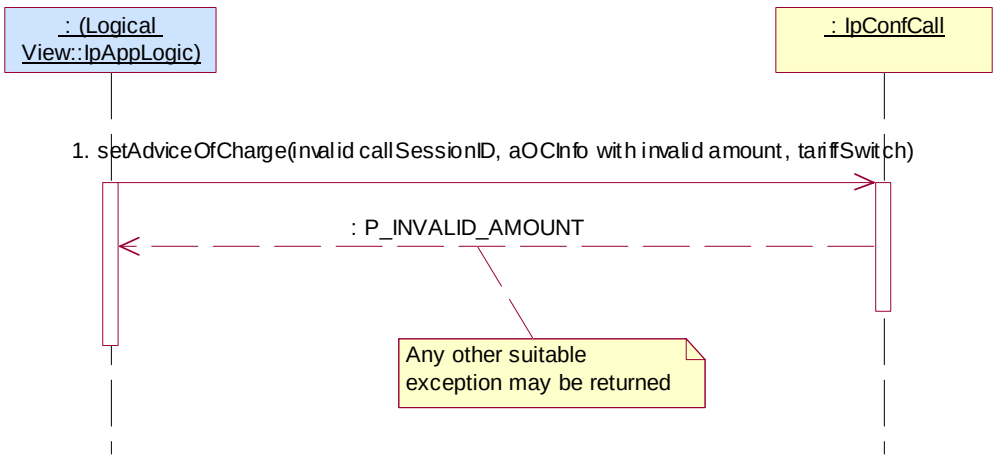
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as CCC _ IpConfCall _02

Condition: setAdviceOfCharge method is supported.

Test Sequence:

1. Method call **setAdviceOfCharge()** on IpConfCall
   Parameters: valid callSessionID returned in 1., aOCInfo, with invalid amount, valid tariffSwitch
   Check: P_INVALID_AMOUNT, or another suitable exception, is returned



Test CCC _ IpConfCall _22

Summary: IpConfCall, superviseReq, P_INVALID_SESSION_ID

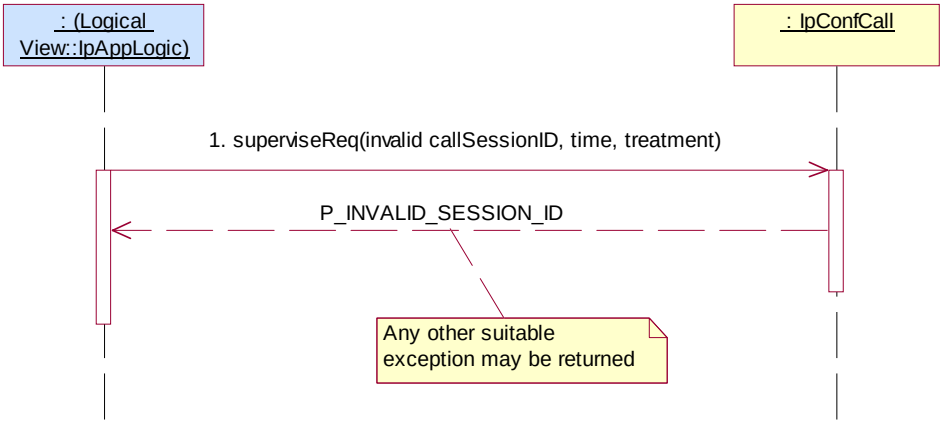
Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Same as CCC _ IpConfCall _02

Condition: superviseReq method is supported.

Test Sequence:

1. Method call **superviseReq()** on IpConfCall
   Parameters: invalid callSessionID, valid time, valid treatment
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpConfCall _23

Summary: IpConfCall, getCallLegs, P_INVALID_SESSION_ID

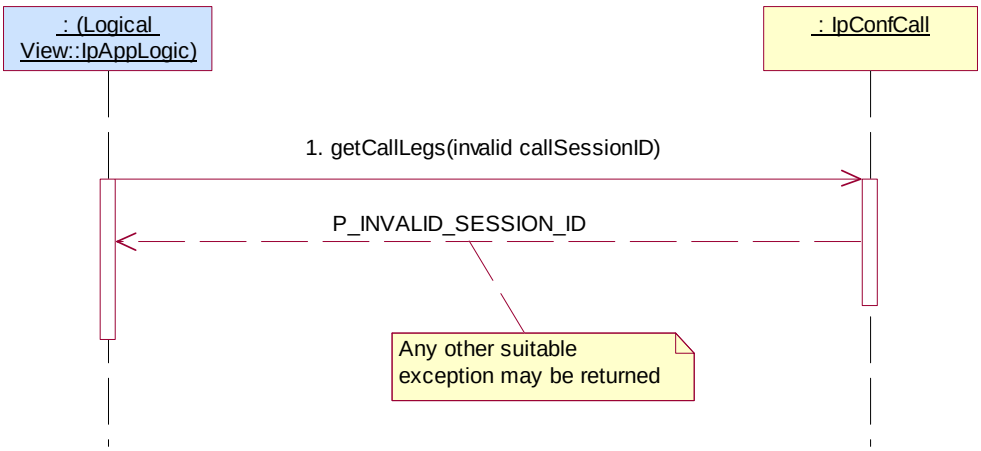
Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Same as CCC _ IpConfCall _08

Condition: createCallLeg, getCallLegs methods are supported.

Test Sequence:

1. Method call **getCallLegs()** on IpConfCall
   Parameters: invalid callSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



5.2.4.3 IpSubConfCall

5.2.4.3.1 Mandatory, valid behaviour

Test CCC _ IpSubConfCall _01

Summary: IpSubConfCall, all methods mandatory, successful

Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Application has a valid callSessionID returned by one of the three following sequence:

1. Method call **createConference()** on IpConfCallControlManager
   Parameters: valid appConferenceCall, valid numberOfSubConferences, valid conferencePolicy, valid numberOfParticipants, valid duration
   Check: valid value of TpConfCallIdentifier is returned

2. Method call **createSubConference()** on IpConfCall
   Parameters: valid conferenceSessionID returned in 1., valid appSubConference, valid conferencePolicy
   Check: valid value of TpSubConfCallIdentifier is returned

3. Method call **getSubConferences()** on IpConfCall
   Parameters: valid conferenceSessionID returned in 1.
   Check: valid value of TpSubConfCallIdentifierSet is returned which contains TpSubConfCallIdentifier returned in 2.

either

4. Method call **createCallLeg()** on IpSubConfCall
   Parameters: valid callSessionID returned in 2., valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

5. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 3., valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

or

4. Method call **createAndRouteCallLegReq()** on IpSubConfCall
   Parameters: valid callSessionID returned in 2., valid eventsRequested, valid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: valid value of TpCallLegIdentifier

or

1. Method call **reserveResources()**
   Parameters: valid appInterface, valid startTime, valid numberOfParticipants, valid duration, valid conferencePolicy
   Check: valid value of TpResourceReservation is returned

2. Triggered action: cause IUT to call **conferenceCreated()** on Tester's (application's) IpAppConfCallControlManager interface
   Parameters: valid conferenceCall.

3. Method call **getSubConferences()** on IpConfCall
   Parameters: valid conferenceSessionID returned in 1.
   Check: valid value of TpSubConfCallIdentifierSet is returned which contains TpSubConfCallIdentifier returned in 2.

either

4. Method call **createCallLeg()** on IpSubConfCall
   Parameters: valid callSessionID reported in 2., valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

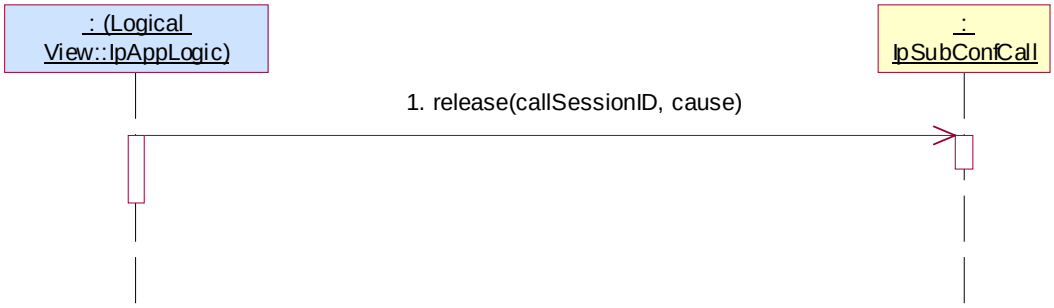
5. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 3., valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

or

4. Method call **createAndRouteCallLegReq()** on IpSubConfCall
   Parameters: valid callSessionID reported in 2., valid eventsRequested, valid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: valid value of TpCallLegIdentifier

Test Sequence:

1. Method call **release()** on IpSubConfCall
   Parameters: valid callSessionID returned in preamble, valid cause
   Check: no exception is returned



Test CCC _ IpSubConfCall _02

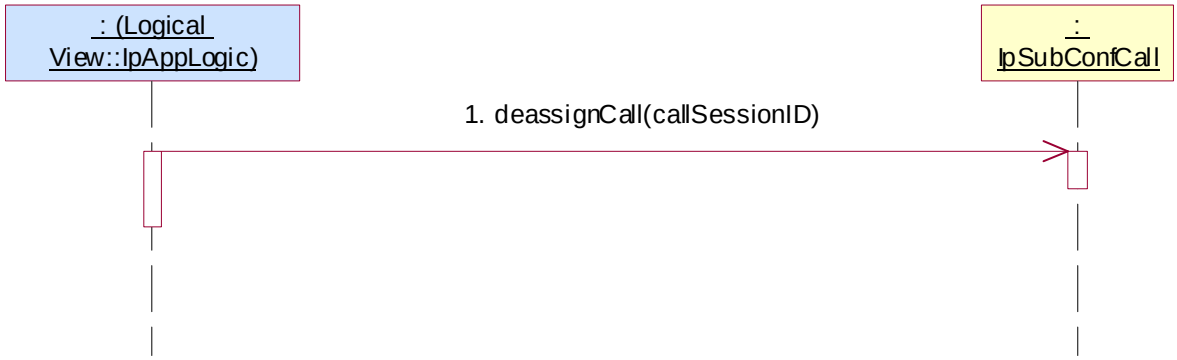
Summary: IpSubConfCall, all methods mandatory, successful

Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as CCC _ IpSubConfCall _01

Test Sequence:

1. Method call **deassignCall()** on IpSubConfCall
   Parameters: valid callSessionID returned in preamble.
   Check: no exception is returned



According Call Control SCF specification, at least one of the two following test sequence is mandatory

Test CCC _ IpSubConfCall _03

Summary: IpSubConfCall, all methods mandatory, successful

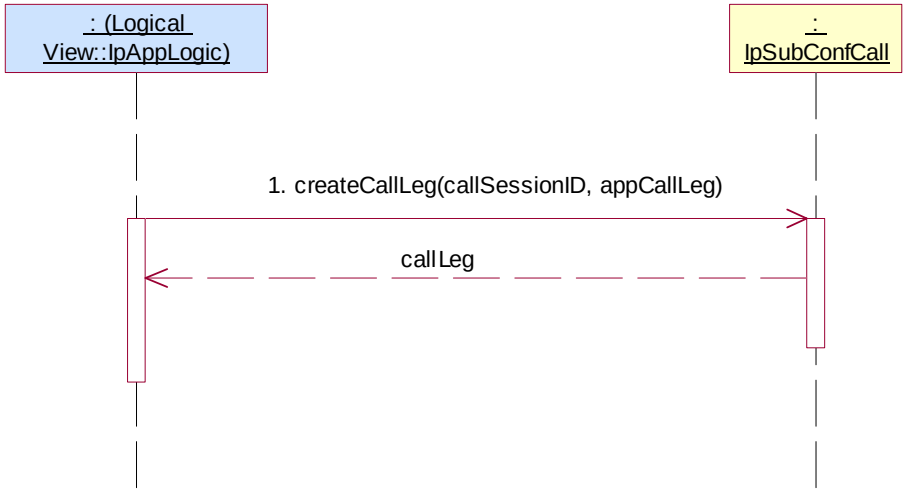
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Condition: createCall, createCallLeg methods are supported.

Preamble: Same as CCC_IpConfCall_02

Test Sequence:

1. Method call **createCallLeg()** on IpSubConfCall
Parameters: valid callSessionID returned in 2., valid appCallLeg
Check: valid value of TpCallLegIdentifier is returned



Test CCC _ IpSubConfCall _04

Summary: IpSubConfCall, all methods mandatory, successful

Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Condition: createCall, createAndRouteCallLeg method is supported.

Preamble CCC_IpConfCall_02

1. Method call **createAndRouteCallLegReq()** on IpSubConfCall
   Parameters: valid callSessionID returned in 2., valid eventsRequested, valid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: valid value of TpCallLegIdentifier

According Call Control SCF specification, at least one of the two following test sequence is mandatory

Test CCC _ IpSubConfCall _05

Summary: IpSubConfCall, all methods mandatory, successful

Reference: ES 201 915-4 [1], clauses 9.3.1 and 9.3.3.

Preamble: Application has a valid subConferenceSessionID returned by one of the following sequence:

1. Method call **createConference()** on IpConfCallControlManager
   Parameters: valid appConferenceCall, valid numberOfSubConferences equal to 1, valid conferencePolicy, valid numberOfParticipants, valid duration
   Check: valid value of TpConfCallIdentifier is returned

2. Method call **getSubConferences()** on IpConfCall
   Parameters: valid conferenceSessionID returned in 1.
   Check: valid value of TpSubConfCallIdentifierSet is returned.

either

3. Method call **createCallLeg()** on IpSubConfCall
   Parameters: valid callSessionID, valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

4. Method call **eventReportReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 3, valid eventsRequested
   Check: no exception is returned

5. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 3, valid targetAddress, valid originatingAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

or

3. Method call **createAndRouteCallLegReq()** on IpSubConfCall
   Parameters: valid callSessionID, valid eventsRequested, valid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: valid value of TpCallLegIdentifier

and then, for both:

6/4 Triggered action: cause IUT to interrupted call leg processing with a notification or an event: cause IUT to call **eventReportRes()** method on the tester's (Application) **IpAppMultiMediaCallLeg** interface.
Parameters: callLegSessionID, errorIndication

or

1. Method call **reserveResources()**
   Parameters: valid appInterface, valid startTime, valid numberOfParticipants, valid duration, valid conferencePolicy
   Check: valid value of TpResourceReservation is returned

2. Triggered action: cause IUT to call **conferenceCreated()** on Tester's (application's) IpAppConfCallControlManager interface
   Parameters: valid conferenceCall.

3. Method call **getSubConferences()** on IpConfCall
   Parameters: valid conferenceSessionID returned in 1.
   Check: valid value of TpSubConfCallIdentifierSet is returned which contains TpSubConfCallIdentifier returned in 2.

either

4. Method call **createCallLeg()** on IpSubConfCall
   Parameters: valid callSessionID, valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

5. Method call **eventReportReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 3, valid eventsRequested
   Check: no exception is returned

6. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 3, valid targetAddress, valid originatingAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

or

4. Method call **createAndRouteCallLegReq()** on IpSubConfCall
   Parameters: valid callSessionID, valid eventsRequested, valid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: valid value of TpCallLegIdentifier

and then, for both:

7/5 Triggered action: cause IUT to interrupted call leg processing with a notification or an event: cause IUT to call **eventReportRes()** method on the tester's (Application) **IpAppMultiMediaCallLeg** interface.
Parameters: callLegSessionID, errorIndication

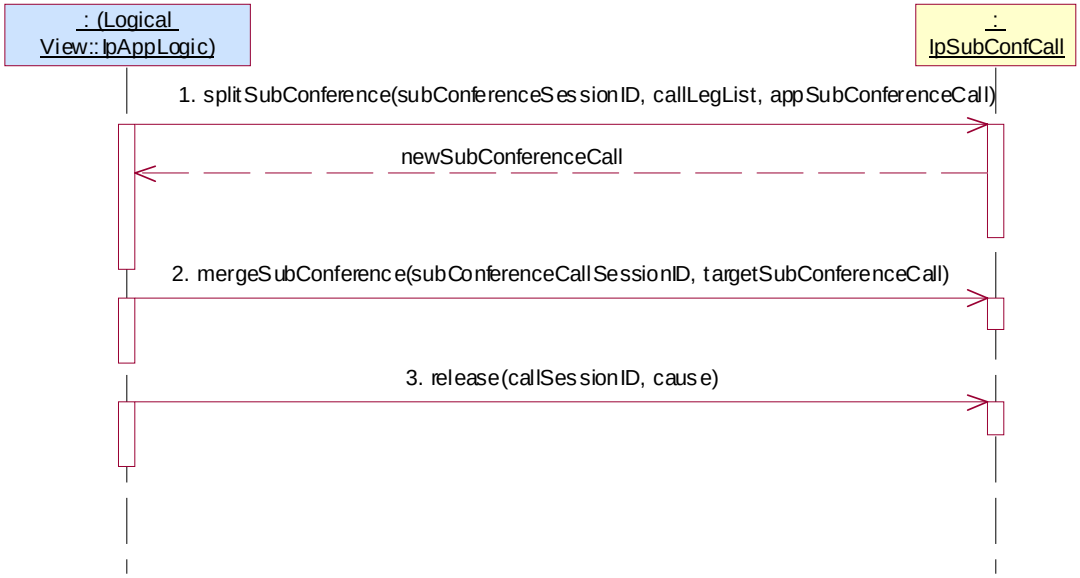
Condition: splitSubConference method is supported.

Test Sequence:

1. Method call **splitSubConference()**
   Parameters: valid subConferenceSessionID returned in preamble, valid callLegList, valid appSubConferenceCall
   Check: valid value of TpSubConfCallIdentifierSet is returned.

2. Method call **mergeSubConference()**
   Parameters: valid subConferenceSessionID returned in preamble, valid targetSubConferenceCall returned in 3.
   Check: no exception is returned.

3. Method call **release() on** IpSubConfCall
   Parameters: valid callLegSessionID returned in preamble, valid cause
   Check: no exception is returned



Test CCC _ IpSubConfCall _06

Summary: IpSubConfCall, all methods mandatory , successful

Reference: ES 201 915-4 [1], clauses 9.3.1 and 9.3.3.

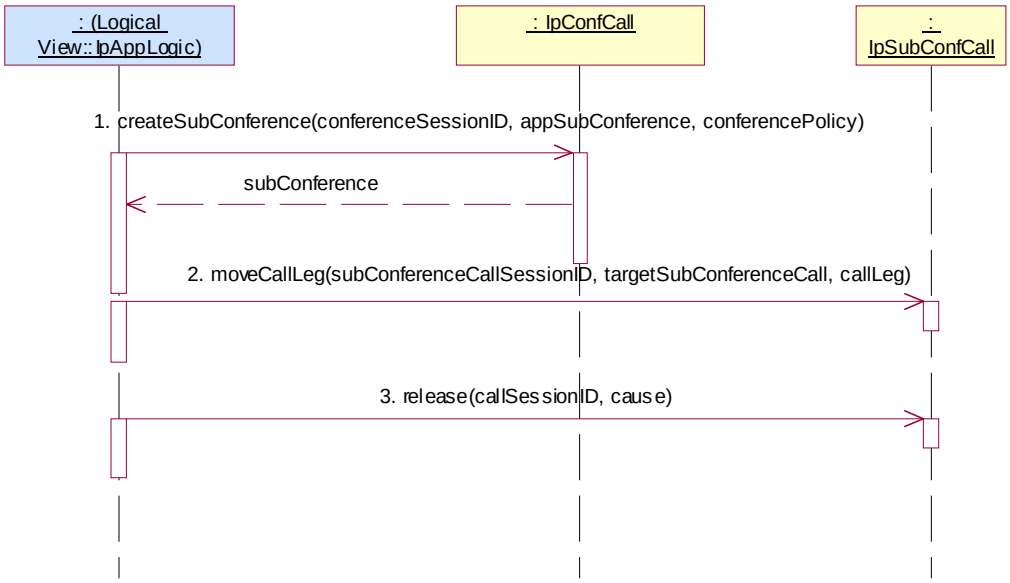
Preamble: Same as CCC _ IpSubConfCall _05

Test Sequence:

1. Method call **createSubConference() on** IpConfCall
   Parameters: valid conferenceSessionID returned in preamble, valid appSubConference, valid conferencePolicy
   Check: valid value of TpSubConfCallIdentifier is returned

2. Method call **moveCallLeg()**
   Parameters: valid subConferenceSessionID returned in preamble, valid targetSubConferenceCall returned in 1., valid callLeg
   Check: no exception is returned.

3. Method call **release()** on IpConfCall
   Parameters: valid callSessionID returned in preamble, valid cause
   Check: no exception is returned



5.2.4.3.2 Mandatory, invalid behaviour

Test CCC _ IpSubConfCall _07

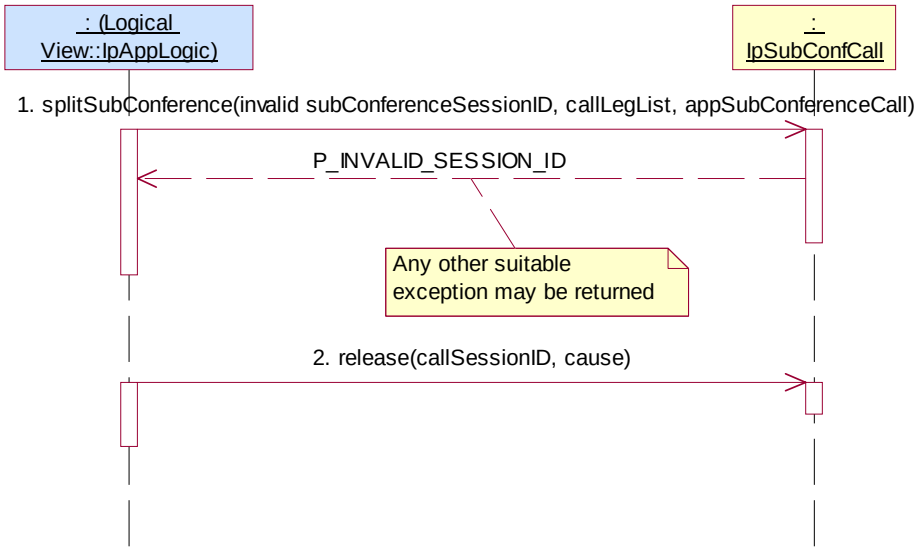
Summary: IpSubConfCall, splitSubConference, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clauses 9.3.1 and 9.3.3.

Preamble: Same as CCC _ IpSubConfCall _05

Test Sequence:

1. Method call **splitSubConference()**
   Parameters: invalid subConferenceSessionID, valid callLegList, valid appSubConferenceCall
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned.



Test CCC _ IpSubConfCall _08

Summary: IpSubConfCall, mergeSubConference, P_INVALID_SESSION_ID

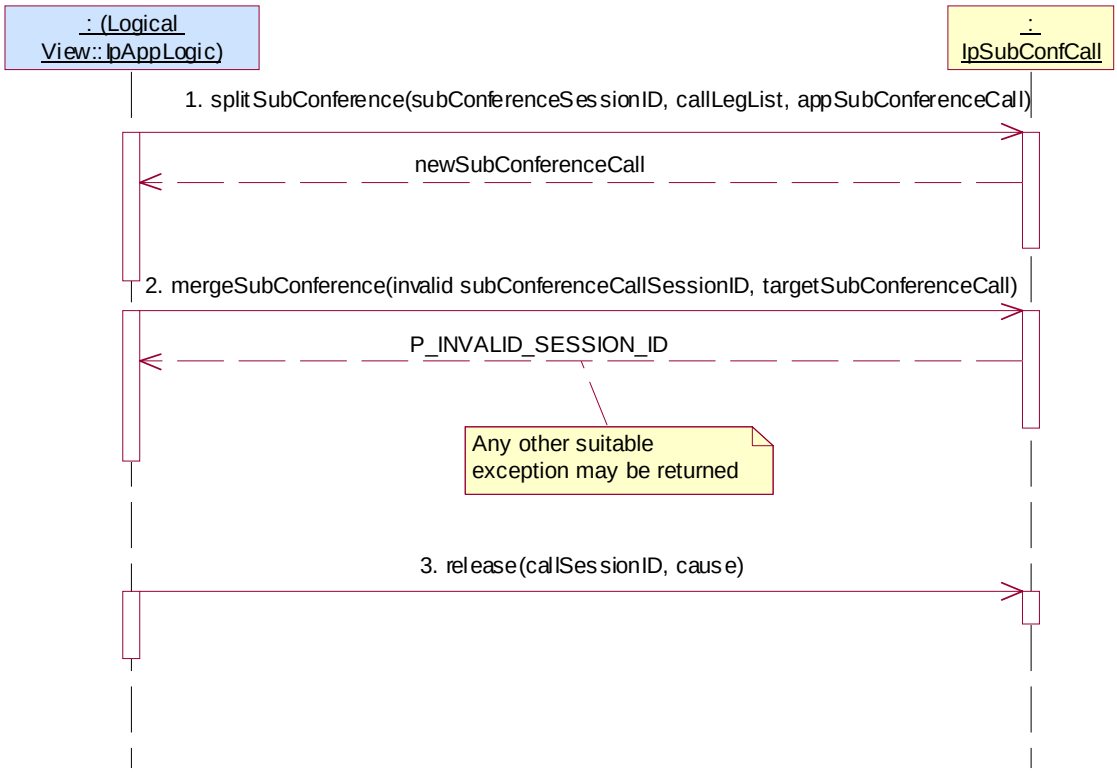
Reference: ES 201 915-4 [1], clauses 9.3.1 and 9.3.3.

Preamble: Same as CCC _ IpSubConfCall _05

Test Sequence:

1. Method call **splitSubConference()**
   Parameters: valid subConferenceSessionID returned in preamble, valid callLegList, valid appSubConferenceCall
   Check: valid value of TpSubConfCallIdentifierSet is returned.

2. Method call **mergeSubConference()**
   Parameters: invalid subConferenceSessionID, valid targetSubConferenceCall
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned.



Test CCC _ IpSubConfCall _09

Summary: IpSubConfCall, mergeSubConference, P_INVALID_SESSION_ID

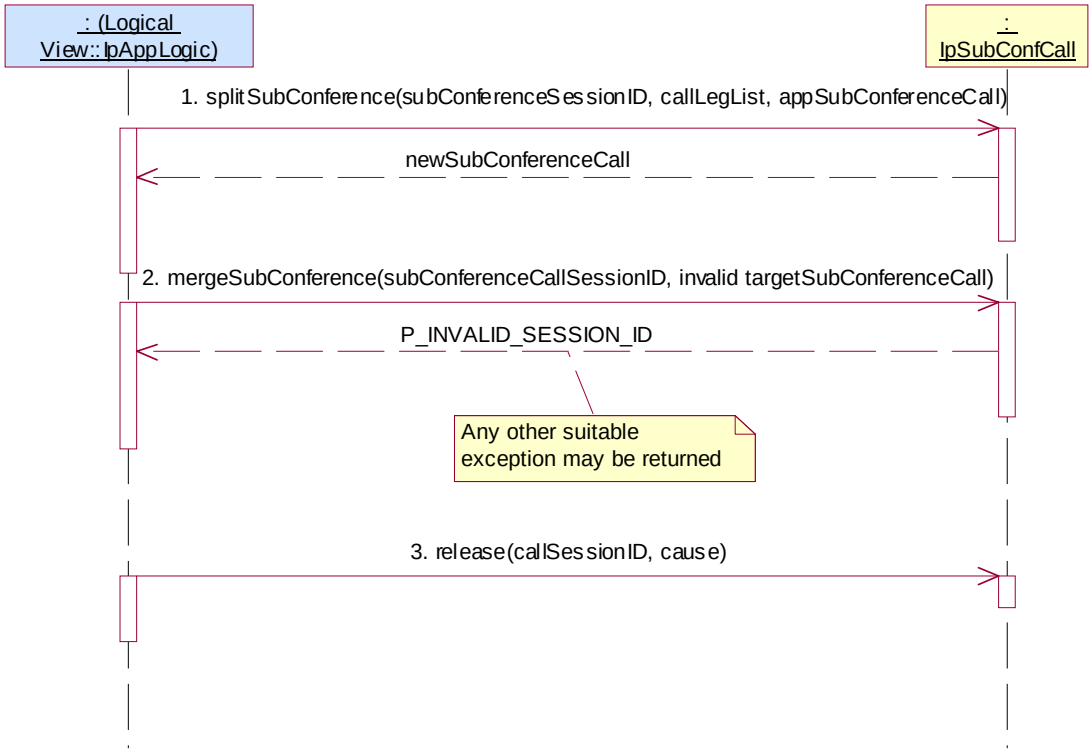
Reference: ES 201 915-4 [1], clauses 9.3.1 and 9.3.3.

Preamble: Same as CCC _ IpSubConfCall _05

Test Sequence:

1. Method call **splitSubConference()**
   Parameters: valid subConferenceSessionID returned in preamble, valid callLegList, valid appSubConferenceCall
   Check: valid value of TpSubConfCallIdentifierSet is returned.

2. Method call **mergeSubConference()**
   Parameters: valid subConferenceSessionID, invalid targetSubConferenceCall
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned.



Test CCC _ IpSubConfCall _10

Summary: IpSubConfCall, moveCallLeg, P_INVALID_SESSION_ID

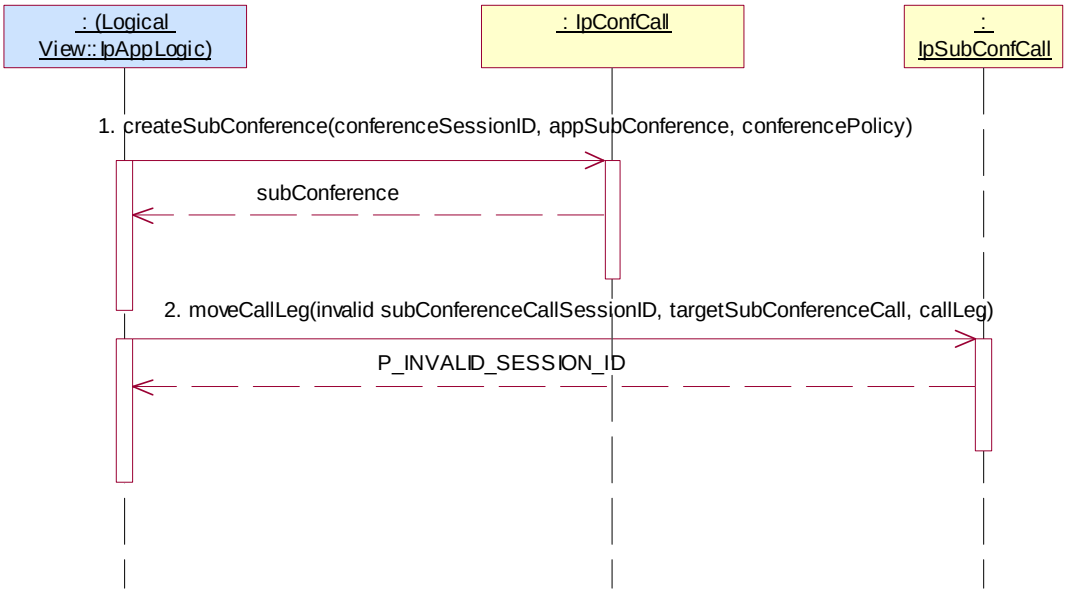
Reference: ES 201 915-4 [1], clauses 9.3.1 and 9.3.3.

Preamble: Same as CCC _ IpSubConfCall _05

Test Sequence:

1. Method call **createSubConference() on** IpConfCall
   Parameters: valid conferenceSessionID returned in preamble, valid appSubConference, valid conferencePolicy
   Check: valid value of TpSubConfCallIdentifier is returned

2. Method call **moveCallLeg()**
   Parameters: invalid subConferenceSessionID, valid targetSubConferenceCall, valid callLeg
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned.



Test CCC _ IpSubConfCall _11

Summary: IpSubConfCall, moveCallLeg, P_INVALID_SESSION_ID

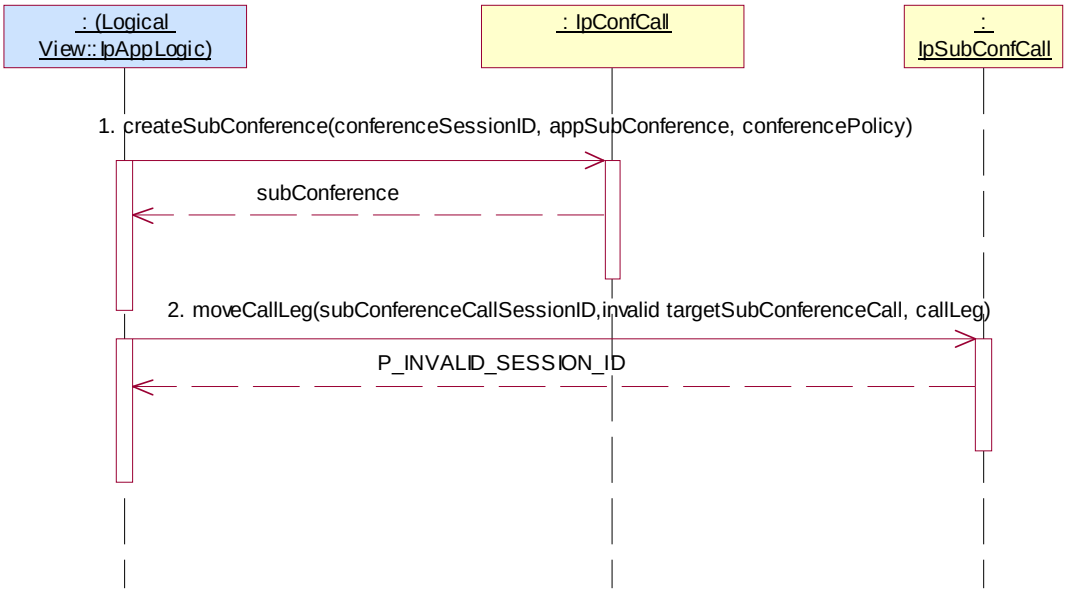
Reference: ES 201 915-4 [1], clauses 9.3.1 and 9.3.3.

Preamble: Same as CCC _ IpSubConfCall _05

Test Sequence:

1. Method call **createSubConference() on** IpConfCall
   Parameters: valid conferenceSessionID returned in preamble, valid appSubConference, valid conferencePolicy
   Check: valid value of TpSubConfCallIdentifier is returned

2. Method call **moveCallLeg()**
   Parameters: valid subConferenceSessionID, invalid targetSubConferenceCall, valid callLeg
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned.



Test CCC _ IpSubConfCall _12

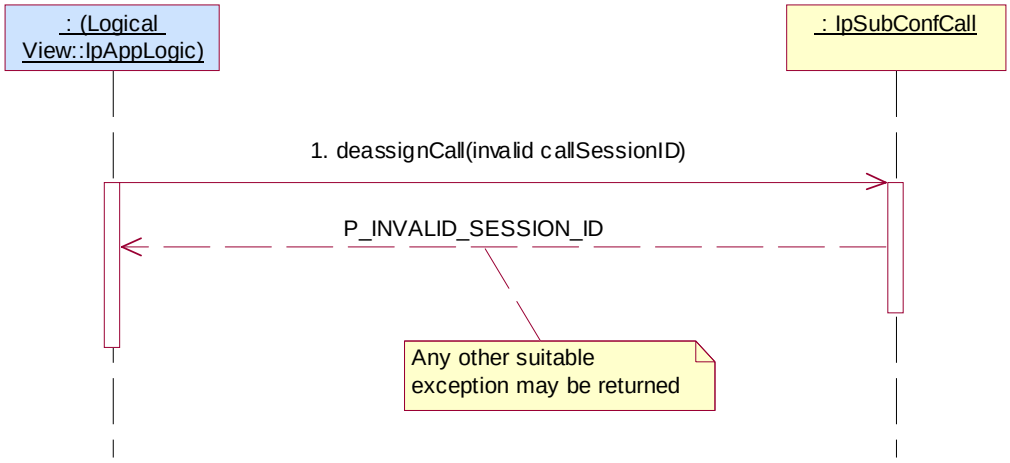
Summary: IpSubConfCall, deassignCall, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Same as CCC _ IpSubConfCall _01

Test Sequence:

1. Method call **deassignCall()** on IpSubConfCall
   Parameters: invalid callSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpSubConfCall _13

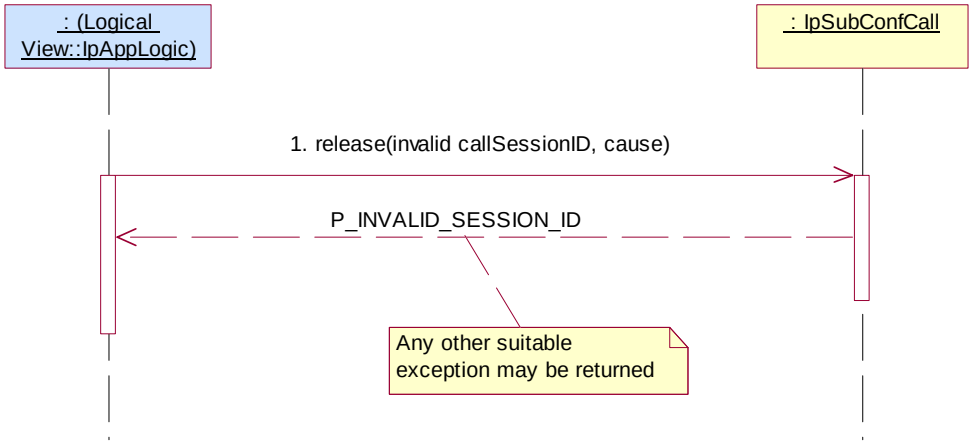
Summary: IpSubConfCall, release, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Same as CCC _ IpSubConfCall _01

Test Sequence:

1. Method call **release()** on IpSubConfCall
   Parameters: invalid callSessionID, valid cause
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpSubConfCall _14

Summary: IpSubConfCall, createCallLeg, P_INVALID_SESSION_ID

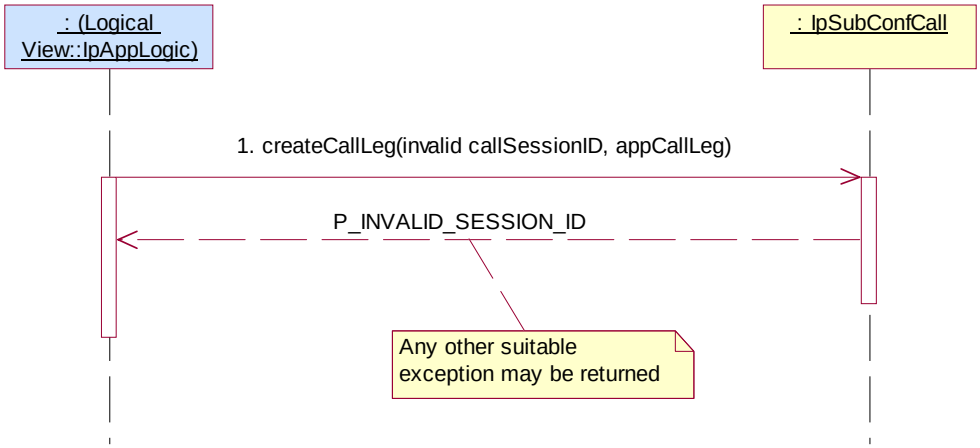
Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Same as CCC_IpConfCall_02

Condition: createCallLeg method is supported.

Test Sequence:

1. Method call **createCallLeg()** on IpSubConfCall
   Parameters: invalid callSessionID, valid appCallLeg
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpSubConfCall _15

Summary: IpSubConfCall, createCallLeg, P_INVALID_INTERFACE_TYPE

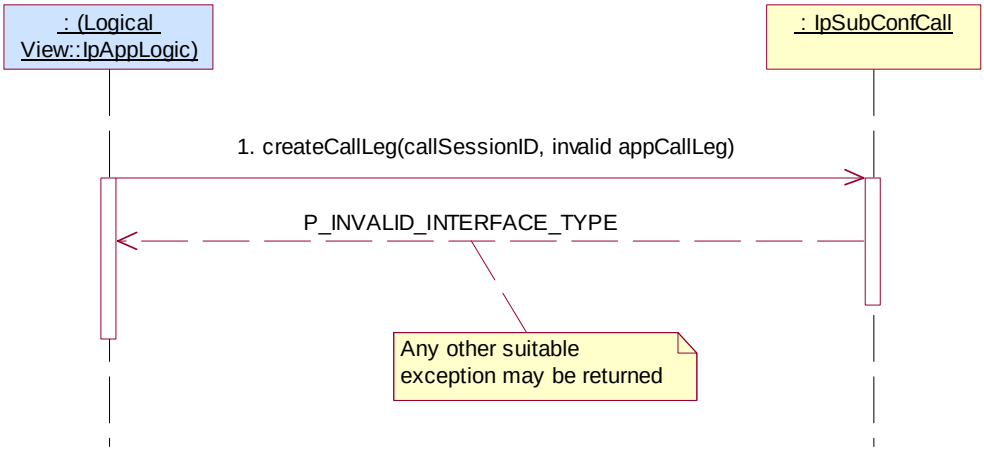
Reference: ES 201 915-4 [1], clause 7.3.1 and 7.3.3.

Preamble: Same as CCC_IpConfCall_02

Condition: CreateCallLeg method is supported.

Test Sequence:

1. Method call **createCallLeg()** on IpSubConfCall
   Parameters: valid callSessionID returned in preamble, invalid appCallLeg
   Check: P_INVALID_INTERFACE_TYPE, or another suitable exception, is returned



Test CCC _ IpSubConfCall _16

Summary: IpSubConfCall, createAndRouteCallLegReq, P_INVALID_SESSION_ID

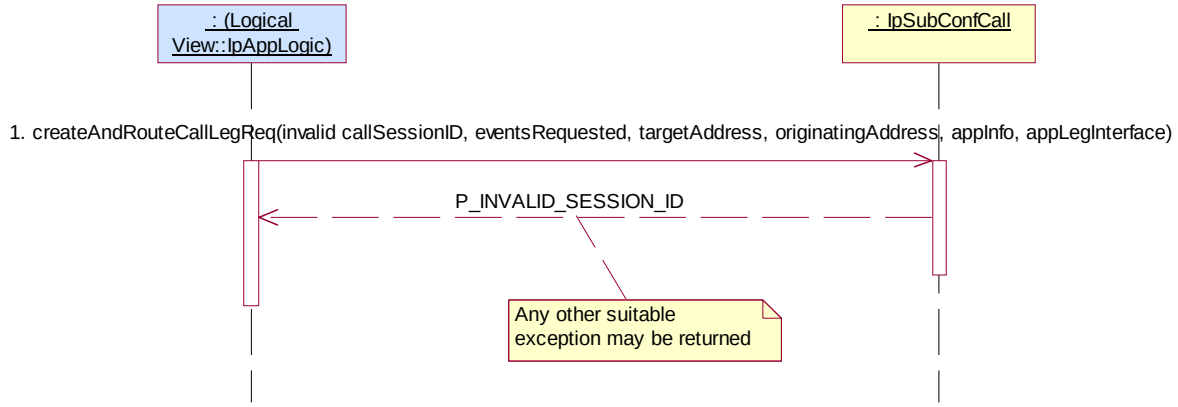
Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Same as CCC_IpConfCall_02

Condition: CreateAndRouteCallLeg method is supported.

Test Sequence:

1. Method call **createAndRouteCallLegReq()** on IpSubConfCall
   Parameters: invalid callSessionID, valid eventsRequested, valid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpSubConfCall _17

Summary: IpSubConfCall, createAndRouteCallLegReq, P_INVALID_INTERFACE_TYPE

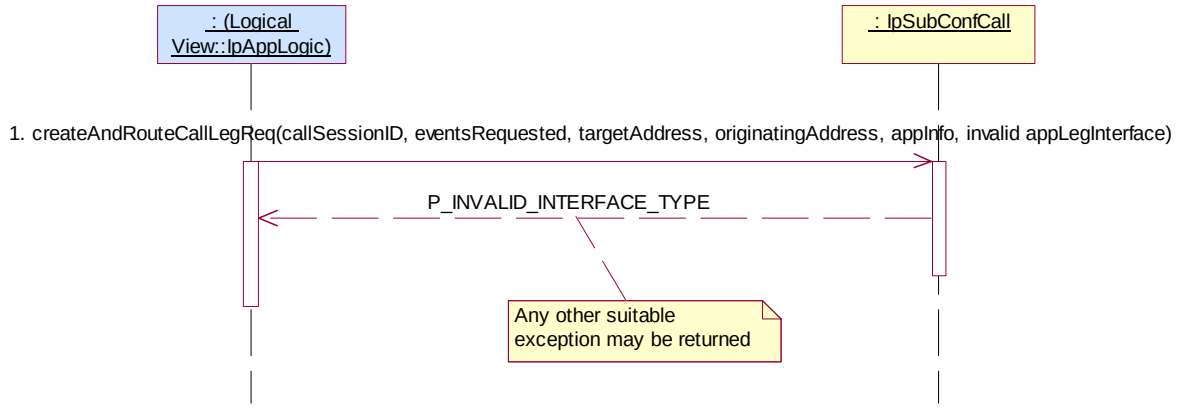
Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Same as CCC_IpConfCall_02

Condition: CreateAndRouteCallLeg method is supported.

Test Sequence:

1. Method call **createAndRouteCallLegReq()** on IpSubConfCall
   Parameters: valid callSessionID, valid eventsRequested, valid targetAddress, valid originatingAddress, valid appInfo, invalid appLegInterface
   Check: P_INVALID_INTERFACE_TYPE, or another suitable exception, is returned



Test CCC _ IpSubConfCall _18

Summary: IpSubConfCall, createAndRouteCallLegReq, P_INVALID_ADDRESS

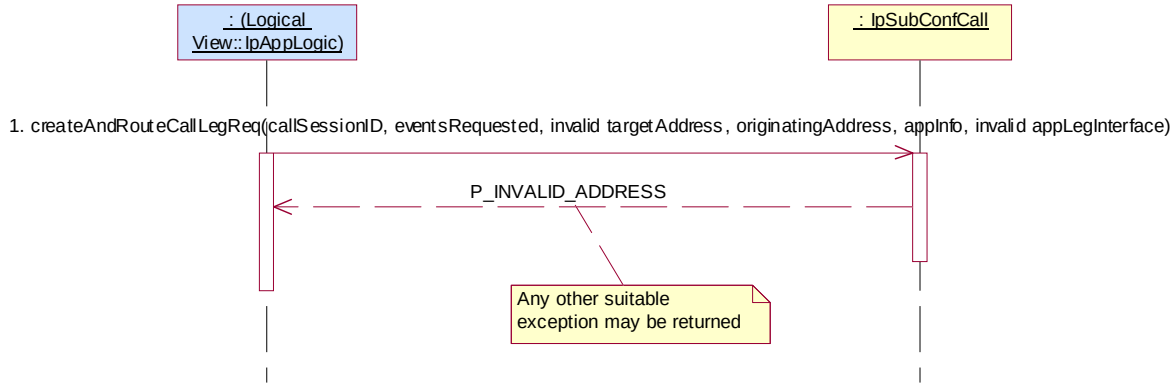
Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Same as CCC_IpConfCall_02

Condition: CreateAndRouteCallLeg method is supported.

Test Sequence:

1. Method call **createAndRouteCallLegReq()** on IpSubConfCall
   Parameters: valid callSessionID, valid eventsRequested, invalid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: P_INVALID_ADDRESS, or another suitable exception, is returned



Test CCC _ IpSubConfCall _19

Summary: IpSubConfCall, createAndRouteCallLegReq, P_INVALID_ADDRESS

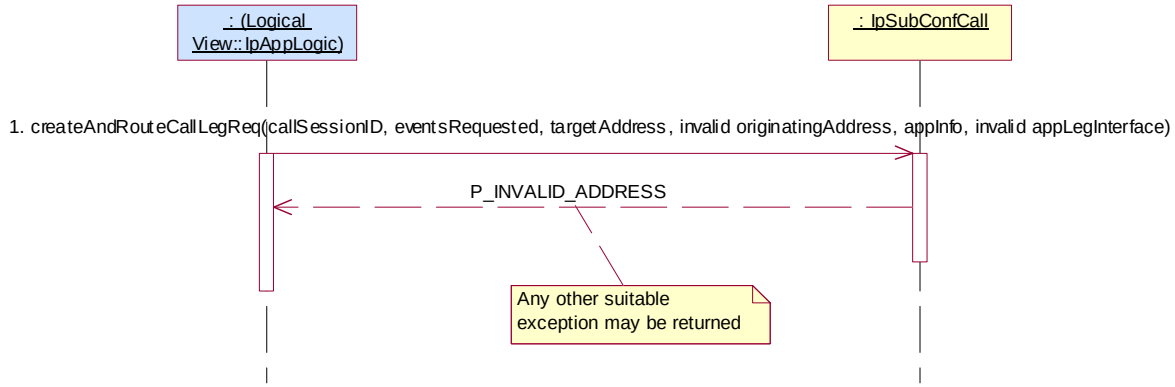
Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Same as CCC_IpConfCall_02

Condition: CreateAndRouteCallLeg method is supported.

Test Sequence:

1. Method call **createAndRouteCallLegReq()** on IpSubConfCall
   Parameters: valid callSessionID, valid eventsRequested, valid targetAddress, invalid originatingAddress, valid appInfo, valid appLegInterface
   Check: P_INVALID_ADDRESS, or another suitable exception, is returned



Test CCC _ IpSubConfCall _20

Summary: IpSubConfCall, createAndRouteCallLegReq, P_INVALID_CRITERIA

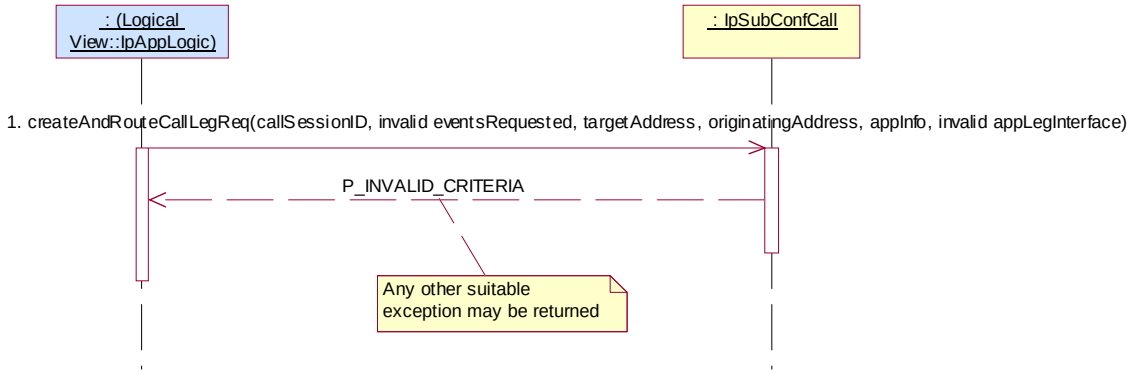
Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble: Same as CCC_IpConfCall_02

Condition: CreateAndRouteCallLeg method is supported.

Test Sequence:

1. Method call **createAndRouteCallLegReq()** on IpSubConfCall
   Parameters: valid callSessionID, invalid eventsRequested, valid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: P_INVALID_CRITERIA, or another suitable exception, is returned



5.2.4.3.3 Optional, valid behaviour

Test CCC _ IpSubConfCall _21

Summary: IpSubConfCall, all methods, successful

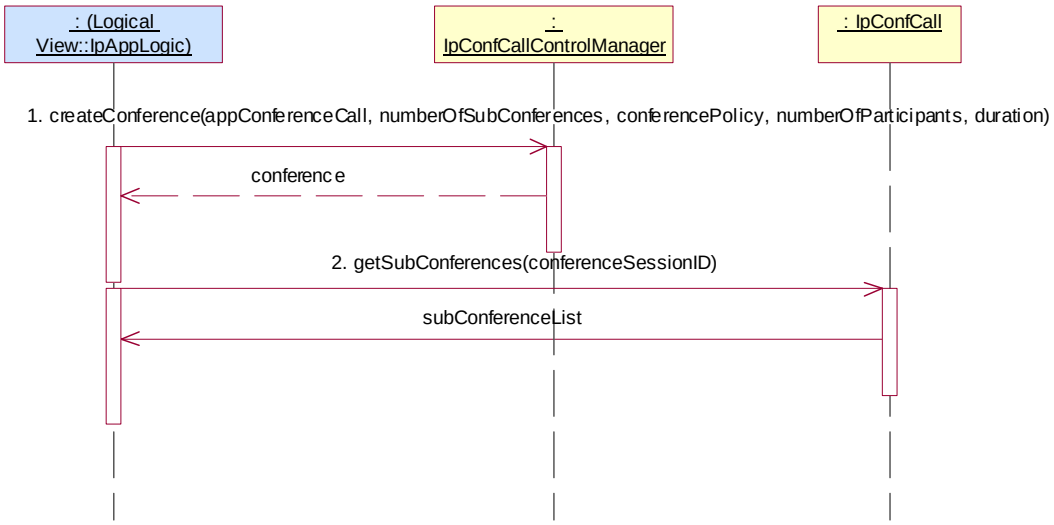
Reference: ES 201 915-4 [1], clause 9.3.1 and 9.3.3.

Preamble: Application has a reference interface used for callbacks.

Test Sequence:

1. Method call **createConference() on** IpConfCallControlManager
   Parameters: valid appConferenceCall, valid numberOfSubConferences equal to 1, valid conferencePolicy, valid numberOfParticipants, valid duration
   Check: valid value of TpConfCallIdentifier is returned

2. Method call **getSubConferences()** on IpConfCall
   Parameters: valid conferenceSessionID returned in 1.
   Check: valid value of TpSubConfCallIdentifierSet is returned.



Test CCC _ IpSubConfCall _22

Summary: IpSubConfCall, all methods, successful

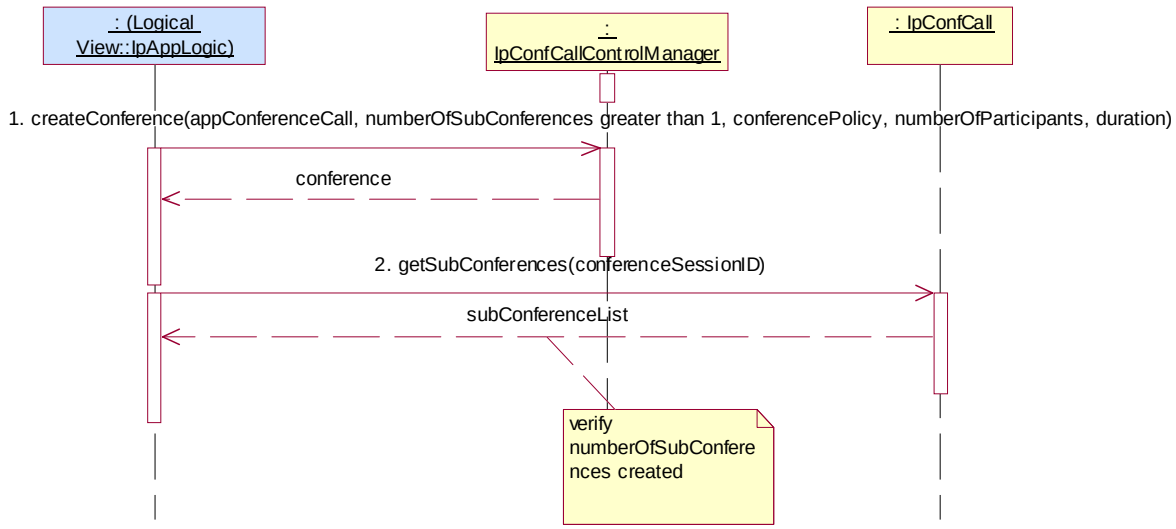
Reference: ES 201 915-4 [1], clauses 9.3.1 and 9.3.3.

Preamble: Application has a reference interface used for callbacks.

Test Sequence:

1. Method call **createConference() on** IpConfCallControlManager
   Parameters: valid appConferenceCall, valid numberOfSubConferences greater than 1, valid conferencePolicy, valid numberOfParticipants, valid duration
   Check: valid value of TpConfCallIdentifier is returned

2. Method call **getSubConferences()** on IpConfCall
   Parameters: valid conferenceSessionID returned in 1.
   Check: valid value of TpSubConfCallIdentifierSet is returned and verify numberOfSubConferences created.



Test CCC _ IpSubConfCall _23

Summary: IpSubConfCall, all methods, successful

Reference: ES 201 915-4 [1], clauses 9.3.1 and 9.3.3.

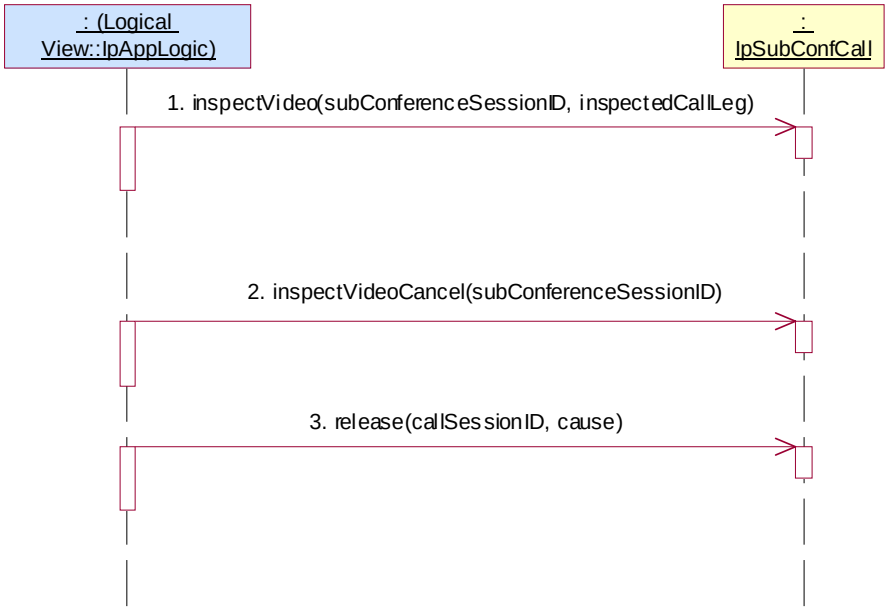
Preamble: Same as CCC _ IpSubConfCall _01

Test Sequence:

1. Method call **inspectVideo()**
   Parameters: valid subConferenceSessionID returned in preamble, valid inspectedCallLeg
   Check: no exception is returned.

2. Method call **inspectVideoCancel()**
   Parameters: valid subConferenceSessionID returned in preamble
   Check: no exception is returned.

3. Method call **release() on** IpSubConfCall
   Parameters: valid callLegSessionID returned in preamble, valid cause
   Check: no exception is returned



Test CCC _ IpSubConfCall _24

Summary: IpSubConfCall, all methods, successful

Reference: ES 201 915-4 [1], clauses 9.3.1 and 9.3.3.

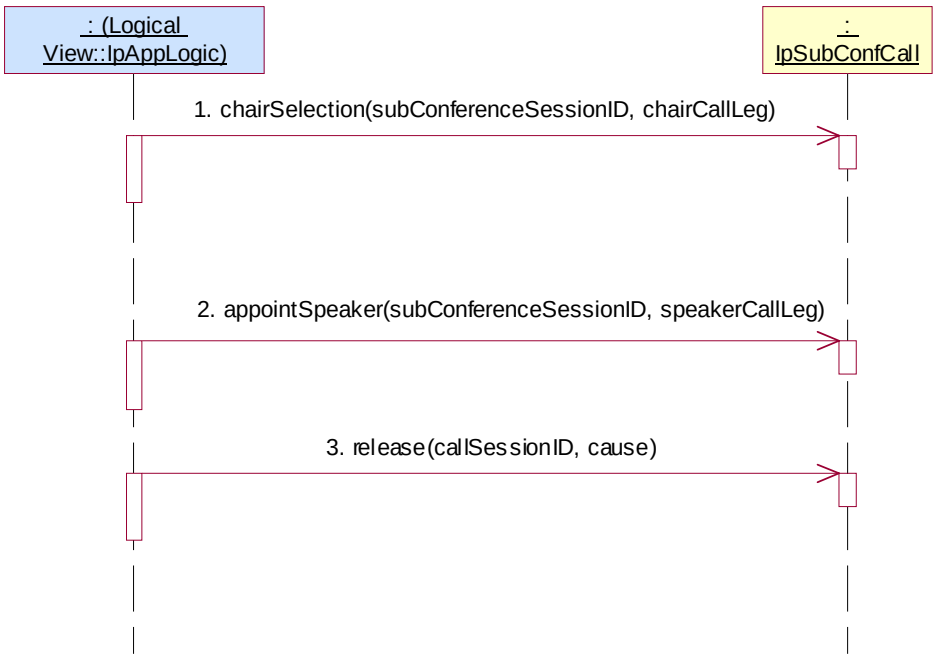
Preamble: Same as CCC _ IpSubConfCall _01

Test Sequence:

1. Method call **chairSelection()**
   Parameters: valid subConferenceSessionID returned in preamble, valid chairCallLeg
   Check: no exception is returned.

2. Method call **appointSpeaker()**
   Parameters: valid subConferenceSessionID returned in preamble, valid speakerCallLeg
   Check: no exception is returned.

3. Method call **release() on** IpSubConfCall
   Parameters: valid callLegSessionID returned in preamble, valid cause
   Check: no exception is returned



Test CCC _ IpSubConfCall _25

Summary: IpSubConfCall, all methods, successful

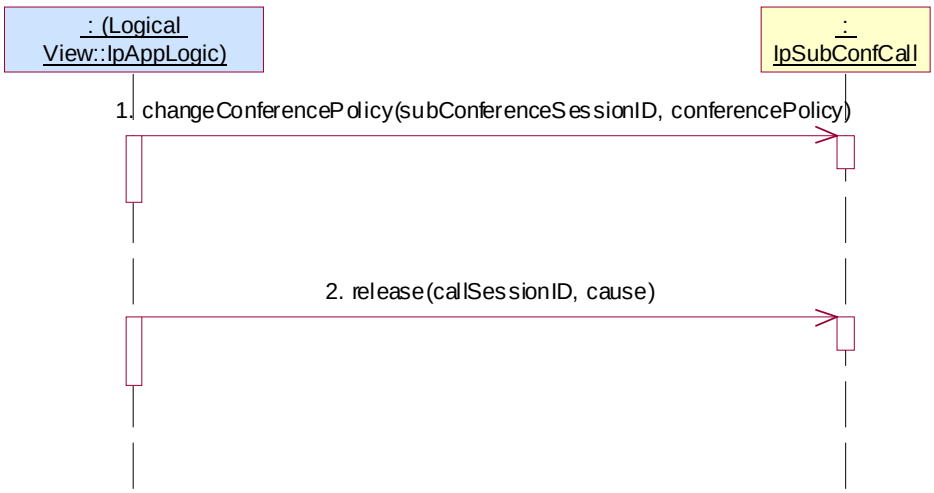
Reference: ES 201 915-4 [1], clauses 9.3.1 and 9.3.3.

Preamble: Same as CCC _ IpSubConfCall _01

Test Sequence:

1. Method call **changeConferencePolicy()**
   Parameters: valid subConferenceSessionID returned in preamble, valid conferencePolicy
   Check: no exception is returned.

2. Method call **release() on** IpSubConfCall
   Parameters: valid callLegSessionID returned in preamble, valid cause
   Check: no exception is returned



Test CCC _ IpSubConfCall _26

Summary: IpSubConfCall, getInfoReq, successful

Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as CCC _ IpConfCall _08

Condition: createCallLeg and getInfoReq methods are supported.

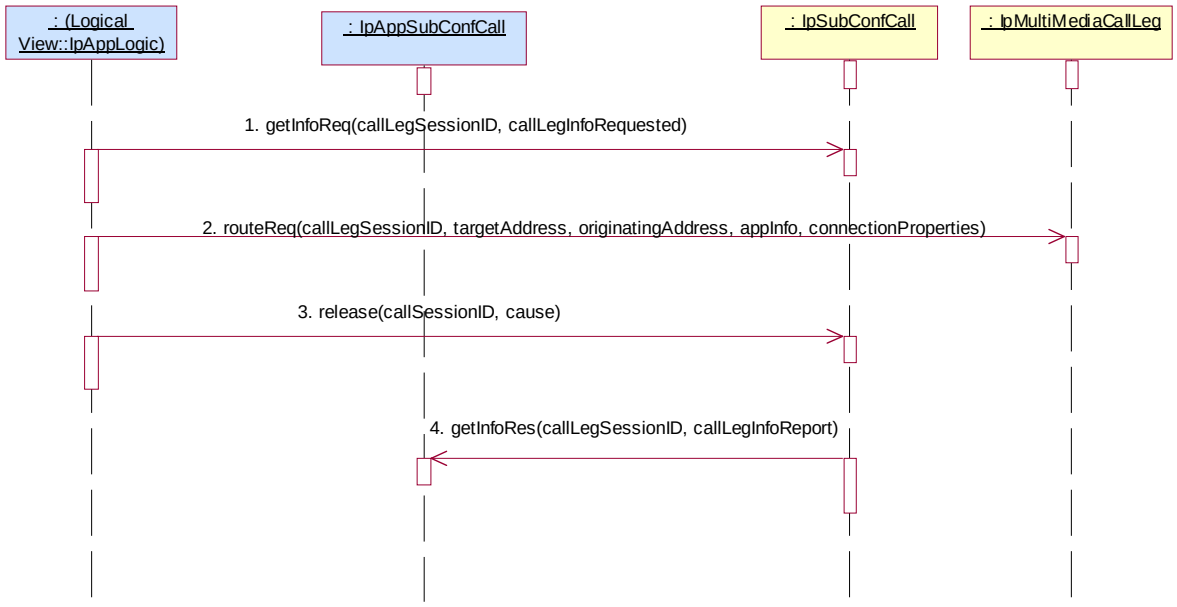
Test Sequence:

1. Method call **getInfoReq()** on IpSubConfCall
   Parameters: valid callSessionID returned in preamble, valid callInfoRequested
   Check: no exception is returned

2. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

3. Method call **release()** on IpSubConfCall
   Parameters: valid callSessionID returned in preamble, valid cause
   Check: no exception is returned

4 Triggered action: cause IUT to call **getInfoRes()** method on the tester's (Application) **IpAppSubConfCall** interface.
   Parameters: callSessionID given in 1., valid callInfoReport.



Test CCC _ IpSubConfCall _27

Summary: IpSubConfCall, setChargePlan, successful

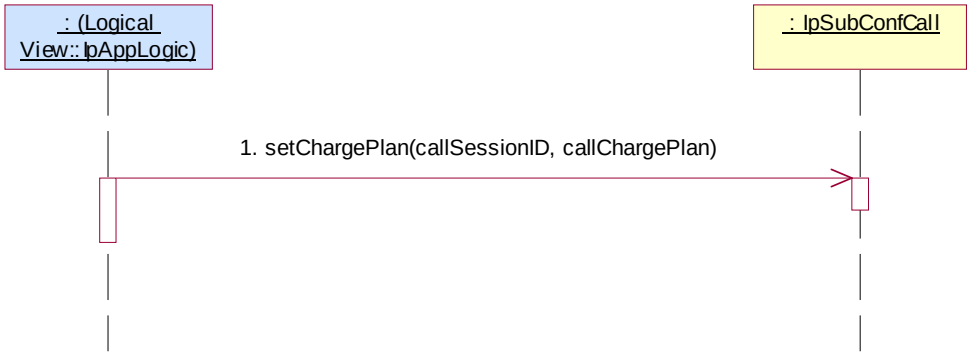
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as CCC _ IpConfCall _08

Condition: createCallLeg and setChargePlan methods are supported.

Test Sequence:

1. Method call **setChargePlan()** on IpSubConfCall
   Parameters: valid callSessionID returned in 1., valid callChargePlan
   Check: no exception is returned



Test CCC _ IpSubConfCall _28

Summary: IpSubConfCall, setAdviceOfCharge, successful

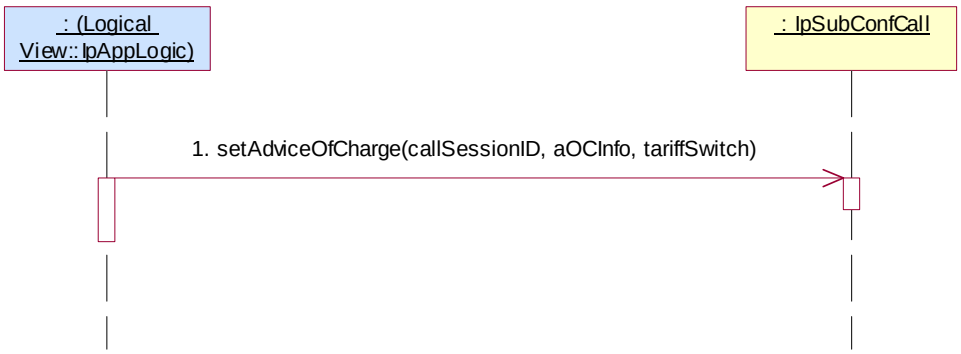
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as CCC _ IpConfCall _08

Condition: createCallLeg and setAdviceOfCharge methods are supported.

Test Sequence:

1. Method call **setAdviceOfCharge()** on IpSubConfCall
   Parameters: valid callSessionID returned in 1., valid aOCInfo, valid tariffSwitch
   Check: no exception is returned



Test CCC _ IpSubConfCall _29

Summary: IpSubConfCall, all methods, successful

Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as CCC _ IpSubConfCall _05

Condition: getCallLegs method is supported.

Test Sequence:

1. Method call **getCallLegs()** on IpSubConfCall
   Parameters: valid callSessionID returned in preamble.
   Check: valid value of TpCallLegIdentifierSet which contains CallLegIdentifier returned in preamble.

5.2.4.3.4 Optional, invalid behaviour

Test CCC _ IpSubConfCall _30

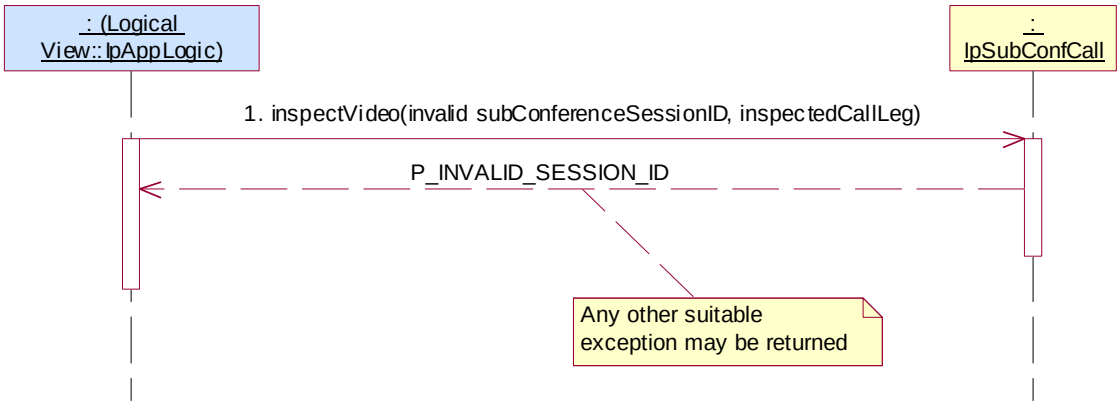
Summary: IpSubConfCall, inspectVideo, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clauses 9.3.1 and 9.3.3.

Preamble: Same as CCC _ IpSubConfCall _01

Test Sequence:

1. Method call **inspectVideo()**
   Parameters: invalid subConferenceSessionID , valid inspectedCallLeg
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned.



Test CCC _ IpSubConfCall _31

Summary: IpSubConfCall, inspectVideoCancel, P_INVALID_SESSION_ID

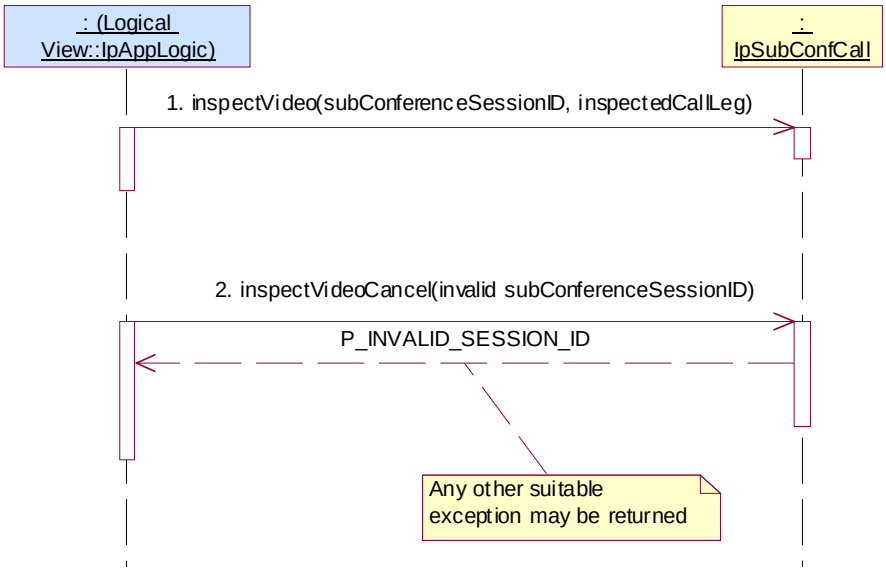
Reference: ES 201 915-4 [1], clauses 9.3.1 and 9.3.3.

Preamble: Same as CCC _ IpSubConfCall _01

Test Sequence:

1. Method call **inspectVideo()**
   Parameters: valid subConferenceSessionID returned in preamble, valid inspectedCallLeg
   Check: no exception is returned.

2. Method call **inspectVideoCancel()**
   Parameters: invalid subConferenceSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned.



Test CCC _ IpSubConfCall _32

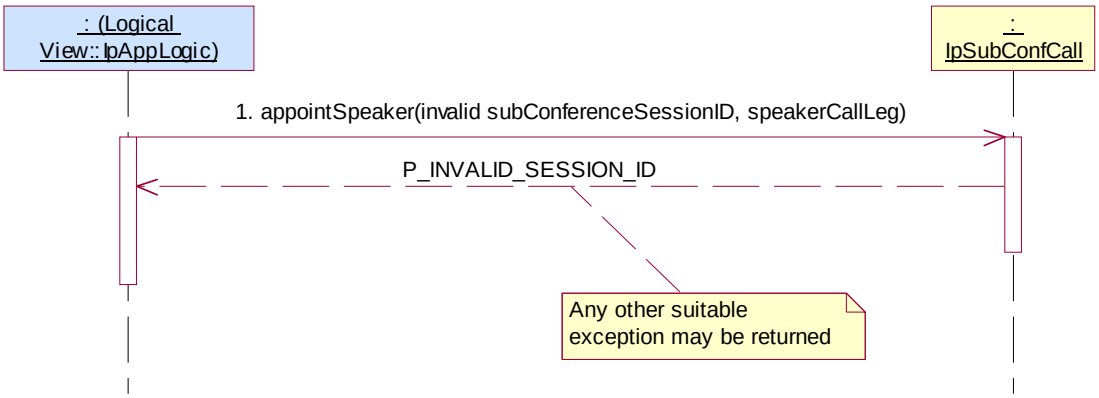
Summary: IpSubConfCall, appointSpeaker, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clauses 9.3.1 and 9.3.3.

Preamble: Same as CCC _ IpSubConfCall _01

Test Sequence:

1. Method call **appointSpeaker()**
   Parameters: invalid subConferenceSessionID, valid speakerCallLeg
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned.



Test CCC _ IpSubConfCall _33

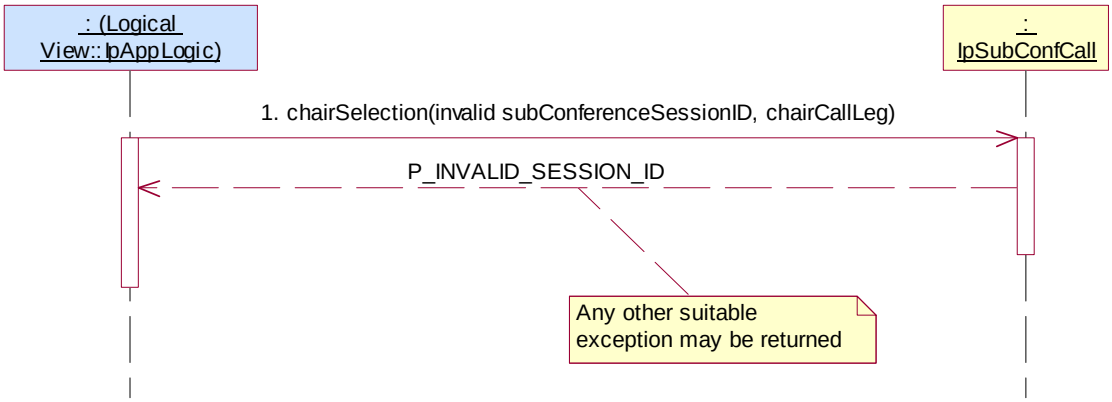
Summary: IpSubConfCall, chairSelection, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clauses 9.3.1 and 9.3.3.

Preamble: Same as CCC _ IpSubConfCall _01

Test Sequence:

1. Method call **chairSelection()**
   Parameters: invalid subConferenceSessionID, valid chairCallLeg
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned.



Test CCC _ IpSubConfCall _34

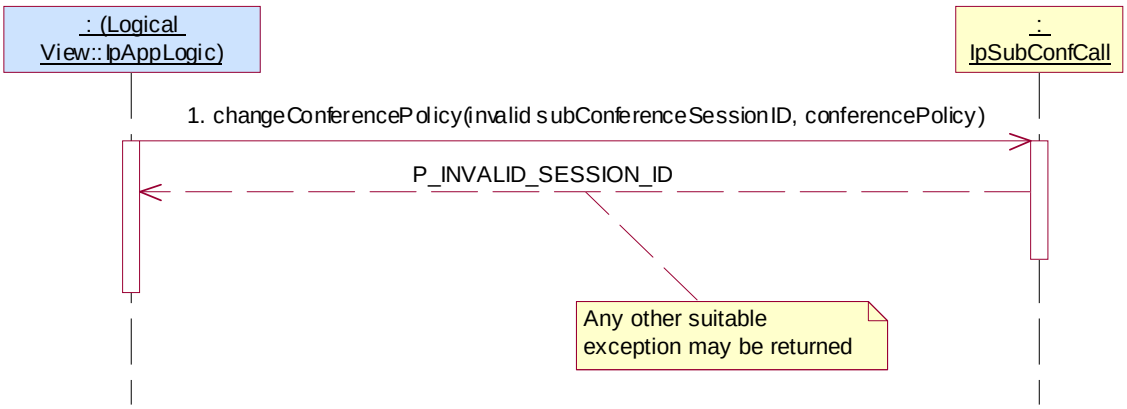
Summary: IpSubConfCall, changeConferencePolicy, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clauses 9.3.1 and 9.3.3.

Preamble: Same as CCC _ IpSubConfCall _01

Test Sequence:

1. Method call **changeConferencePolicy()**
   Parameters: invalid subConferenceSessionID, valid conferencePolicy
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned.



Test CCC _ IpSubConfCall _35

Summary: IpSubConfCall, getInfoReq, P_INVALID_SESSION_ID

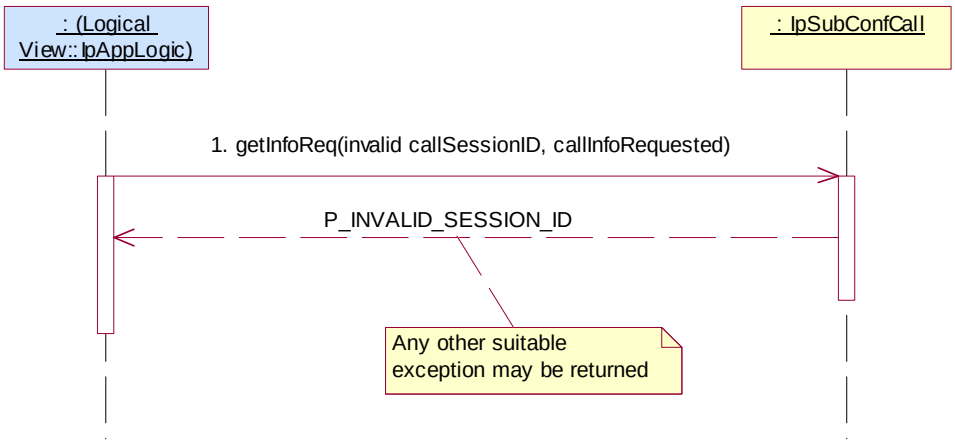
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble: Same as CCC _ IpSubConfCall _01

Condition: createCallLeg and getInfoReq methods are supported.

Test Sequence:

1. Method call **getInfoReq()** on IpSubConfCall
   Parameters: invalid callSessionID, valid callInfoRequested
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpSubConfCall _36

Summary: IpSubConfCall, setChargePlan, P_INVALID_SESSION_ID

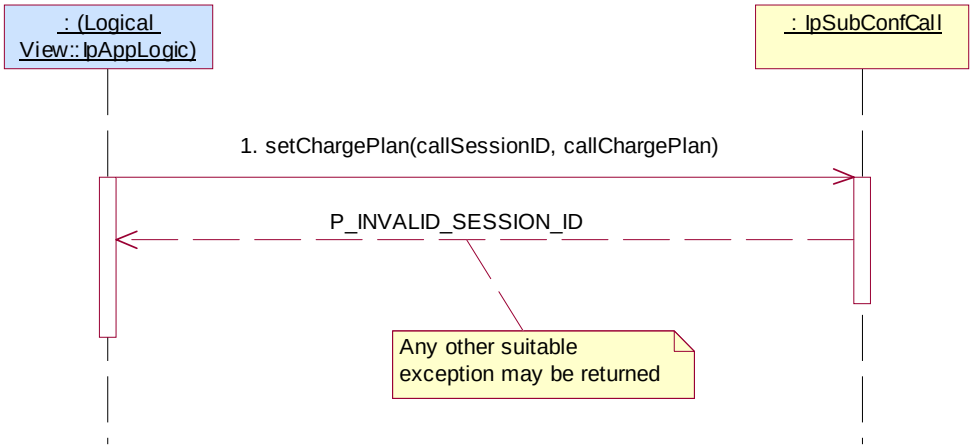
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble Same as CCC_IpConfCall_02

Condition: createCallLeg and setChargePlan methods are supported.

Test Sequence:

1. Method call **setChargePlan()** on IpSubConfCall
   Parameters: invalid callSessionID, valid callChargePlan
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpSubConfCall _37

Summary: IpSubConfCall, setAdviceOfCharge, P_INVALID_SESSION_ID

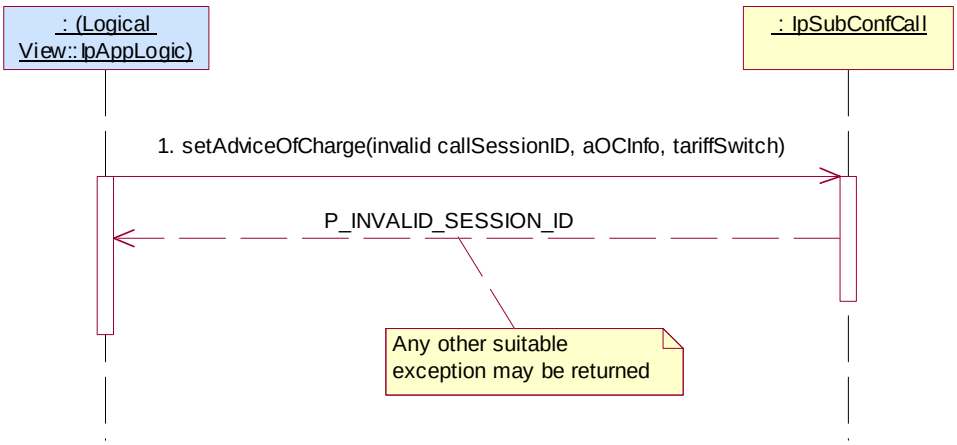
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble Same as CCC_IpConfCall_02

Condition: createCallLeg and setAdviceOfCharge methods are supported.

Test Sequence:

1. Method call **setAdviceOfCharge()** on IpSubConfCall
   Parameters: invalid callSessionID, valid aOCInfo, valid tariffSwitch
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpSubConfCall _38

Summary: IpSubConfCall, setAdviceOfCharge, P_INVALID_CURRENCY

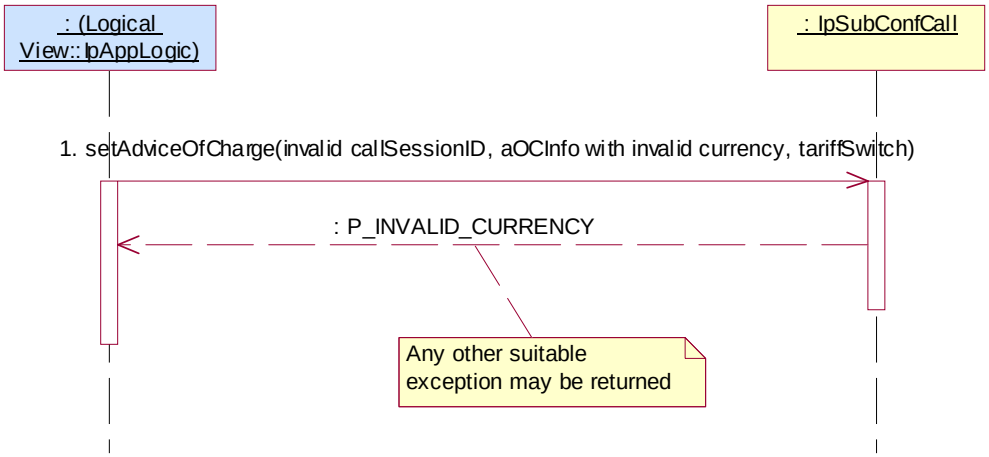
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble Same as CCC_IpConfCall_02

Condition: createCallLeg and setAdviceOfCharge methods are supported.

Test Sequence:

1. Method call **setAdviceOfCharge()** on IpSubConfCall
   Parameters: valid callSessionID returned in 1., aOCInfo with invalid currency, valid tariffSwitch
   Check: P_INVALID_CURRENCY, or another suitable exception, is returned



Test CCC _ IpSubConfCall _39

Summary: IpSubConfCall, setAdviceOfCharge, P_INVALID_AMOUNT

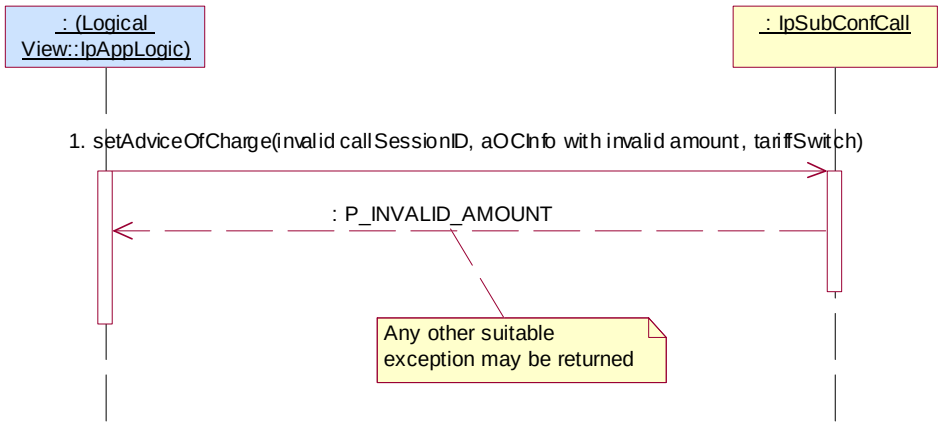
Reference: ES 201 915-4 [1], clauses 7.3.1 and 7.3.3.

Preamble Same as CCC_IpConfCall_02

Condition: createCallLeg and setAdviceOfCharge methods are supported.

Test Sequence:

1. Method call **setAdviceOfCharge()** on IpSubConfCall
   Parameters: valid callSessionID returned in 1., aOCInfo, with invalid amount, valid tariffSwitch
   Check: P_INVALID_AMOUNT, or another suitable exception, is returned



Test CCC _ IpSubConfCall _40

Summary: IpSubConfCall, getCallLegs, P_INVALID_SESSION_ID

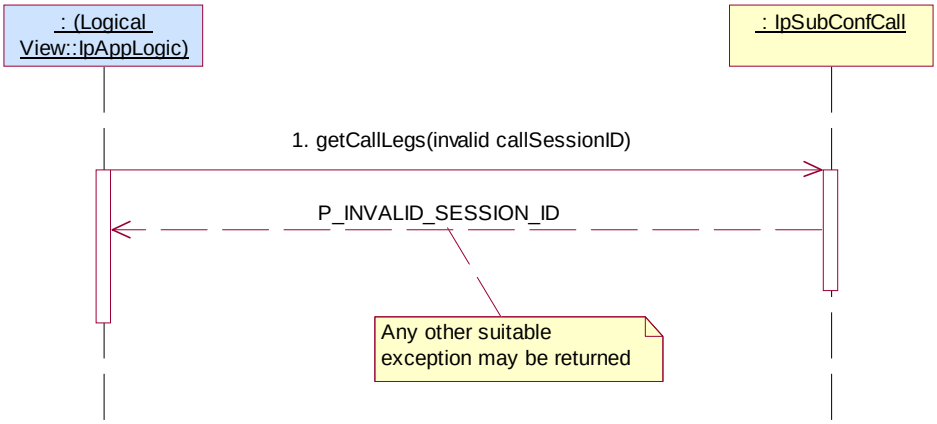
Reference: ES 201 915-4 [1], clause 7.3.3.

Preamble Same as CCC_IpConfCall_02

Condition: CreateCallLeg method is supported.

Test Sequence:

1. Method call **getCallLegs()** on IpSubConfCall
   Parameters: invalid callSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



5.2.4.4 IpMultiMediaCallLeg

5.2.4.4.1 Mandatory, valid behaviour

Test CCC _ IpMultiMediaCallLeg _01

Summary: IpMultiMediaCallLeg, all methods mandatory, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 8.3.5.

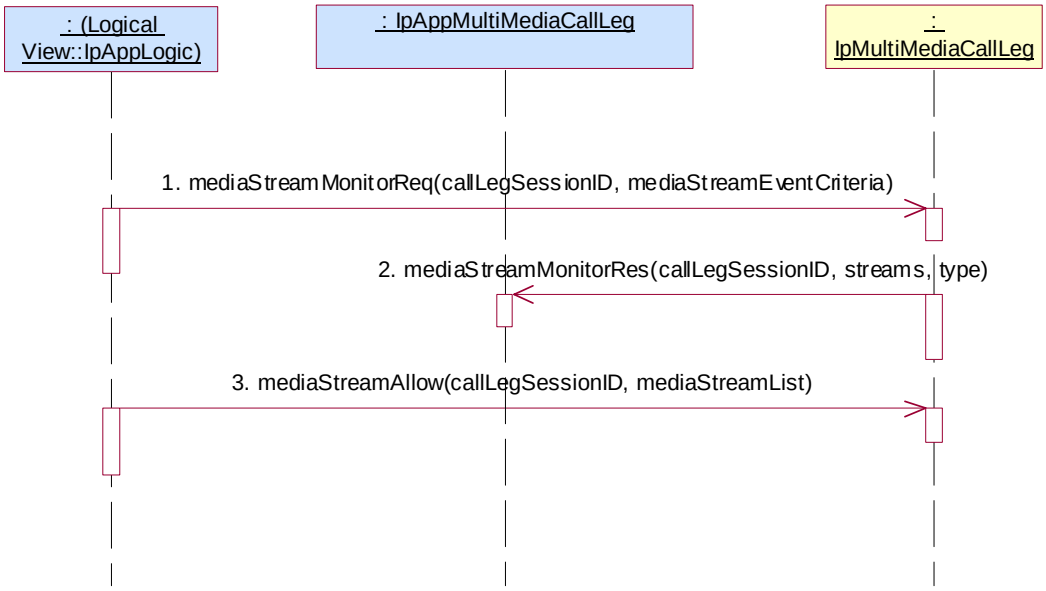
Preamble: Same as CCC _ IpSubConfCall_01

Test Sequence:

1. Method call **mediaStreamMonitorReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid mediaStreamEventCriteria
   Check: no exception is returned

2. Triggered action: cause IUT to call Method **mediaStreamMonitorRes()** method on the tester's (application)
   Parameters: callLegSessionID, streams, type

3. Method call **mediaStreamAllow()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid mediaStreamList
   Check: no exception is returned



Test CCC _ IpMultiMediaCallLeg _02

Summary: IpMultiMediaCallLeg, all mandatory methods, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as CCC _ IpConfCall _08

Condition: createCallLeg method is supported

Test Sequence:

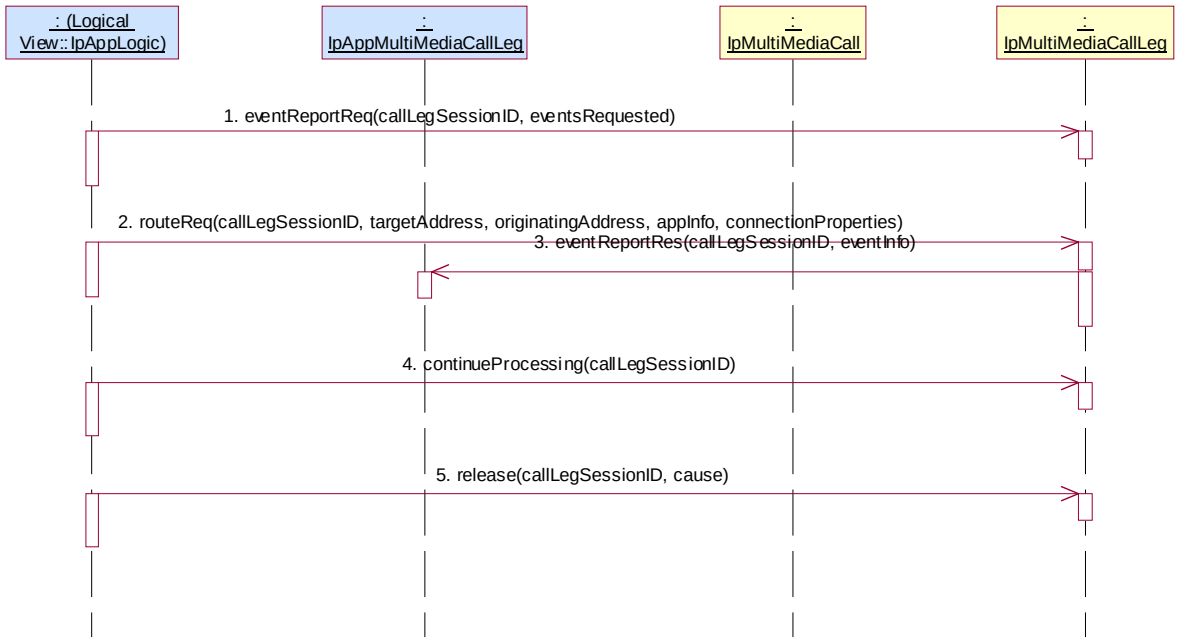
1. Method call **eventReportReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 1, valid eventsRequested
   Check: no exception is returned

2. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 1., valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

3. Triggered action: cause IUT to interrupt call leg processing with a notification or an event: cause IUT to call **eventReportRes()** method on the tester's (Application) **IpAppMultiMediaCallLeg** interface.
   Parameters: callLegSessionID, errorIndication

4. Method call **continueProcessing()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 1.
   Check: no exception is returned

5. Method call **release()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 1, valid cause
   Check: no exception is returned



Test CCC _ IpMultiMediaCallLeg _03

Summary: IpMultiMediaCallLeg, all mandatory methods, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as CCC_IpConfCall_02

Condition: createAndRouteCallLeg method is supported

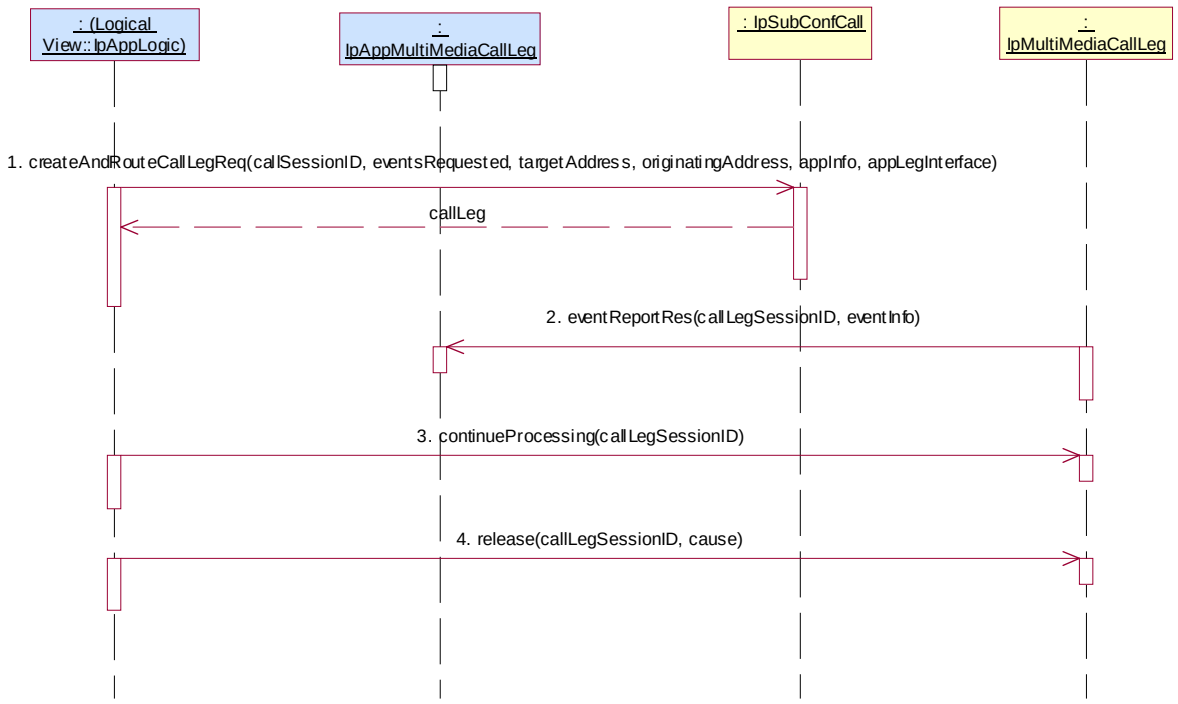
Test Sequence:

1. Method call **createAndRouteCallLeg()** on IpSubConfCall
   Parameters: valid callSessionID returned in preamble, valid eventsRequested, valid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: valid value of TpCallLegIdentifier is returned

2 Triggered action: cause IUT to interrupt call leg processing with a notification or an event: cause IUT to call **eventReportRes()** method on the tester's (Application) **IpAppMultiMediaCallLeg** interface.
   Parameters: callLegSessionID returned in 1, errorIndication

3. Method call **continueProcessing()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble.
   Check: no exception is returned

4. Method call **release()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid cause
   Check: no exception is returned



Test CCC _ IpMultiMediaCallLeg _04

Summary: IpMultiMediaCallLeg, all mandatory methods, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as CCC_ IpSubConfCall _01

Test Sequence:

1. Method call **deassign()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble.
   Check: no exception is returned

5.2.4.4.2 Mandatory, invalid behaviour

Test CCC _ IpMultiMediaCallLeg _05

Summary: IpMultiMediaCallLeg, continueProcessing, : P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.5

Preamble: Same as CCC_IpConfCall_02

Test Sequence:

1. Method call **createCallLeg()** on IpSubConfCall
   Parameters: valid callSessionID returned in preamble, valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

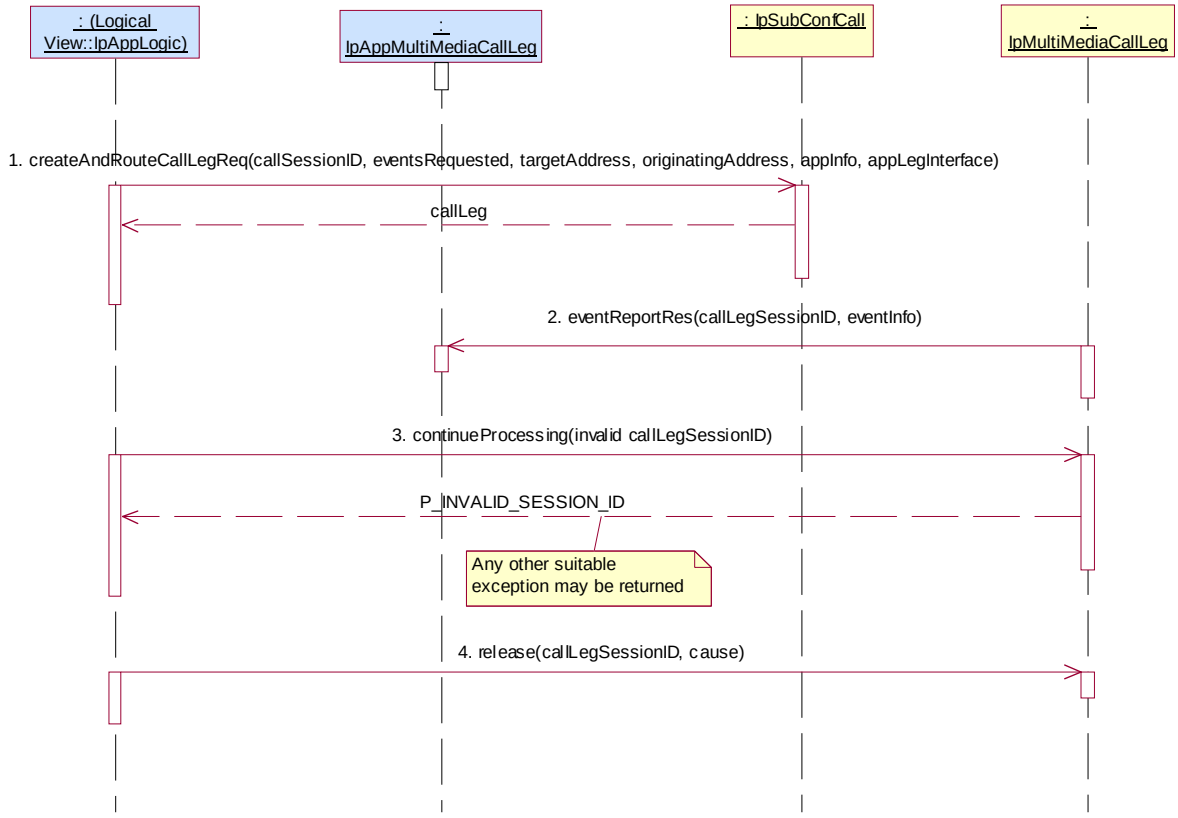
2. Method call **eventReportReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 1, valid eventsRequested
   Check: no exception is returned

3. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 1., valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

4 Triggered action: cause IUT to interrupt call leg processing with a notification or an event: cause IUT to call **eventReportRes()** method on the tester's (Application) **IpAppCallLeg** interface.
   Parameters: callLegSessionID, errorIndication

5. Method call **continueProcessing()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned

6. Method call **release()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 1, valid cause
   Check: no exception is returned



Test CCC _ IpMultiMediaCallLeg _06

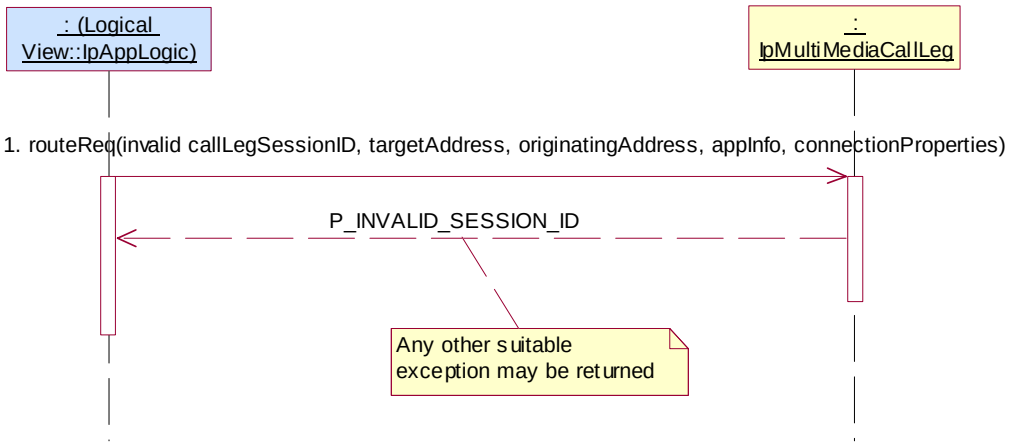
Summary: IpMultiMediaCallLeg, routeReq: P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.5

Preamble: Same as CCC _ IpConfCall _08

Test Sequence:

1. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID, valid targetAddress, valid originatingAddress, valid appInfo, valid connectionProperties
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpMultiMediaCallLeg _07

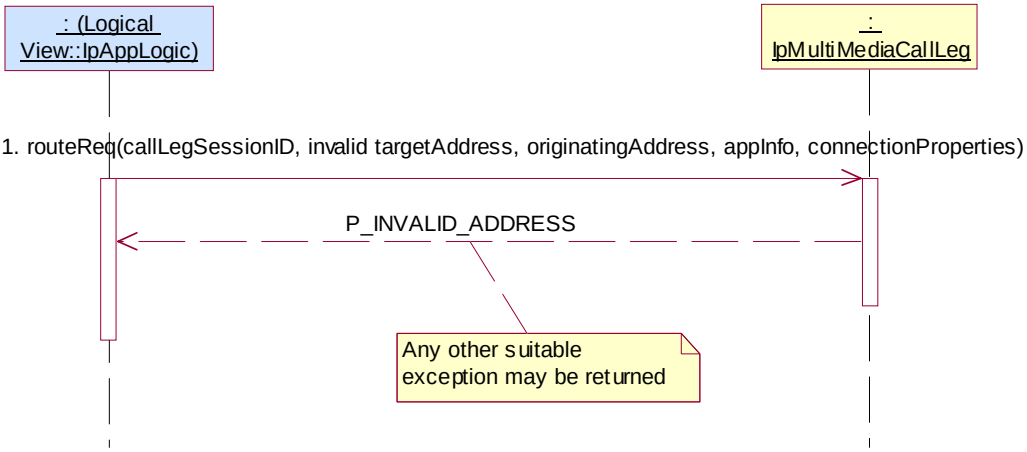
Summary: IpMultiMediaCallLeg, routeReq, P_INVALID_ADDRESS

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as CCC _ IpConfCall _08

Test Sequence:

1. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, invalid targetAddress, valid originatingAddress, valid appInfo, valid connectionProperties
   Check: P_INVALID_ADDRESS, or another suitable exception, is returned



Test CCC _ IpMultiMediaCallLeg _08

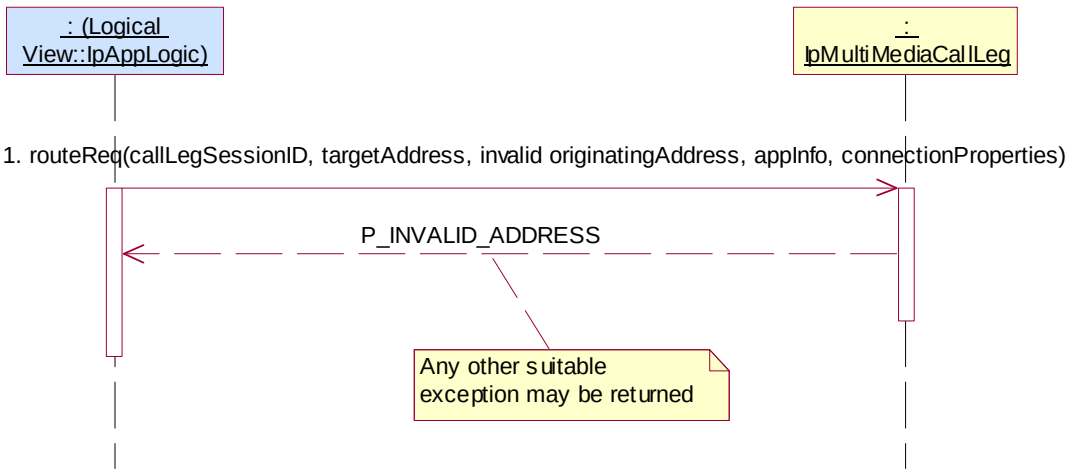
Summary: IpMultiMediaCallLeg, routeReq, P_INVALID_ADDRESS

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as CCC _ IpConfCall _08

Test Sequence:

1. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid targetAddress, invalid originatingAddress, valid appInfo, valid connectionProperties
   Check: P_INVALID_ADDRESS, or another suitable exception, is returned



Test CCC _ IpMultiMediaCallLeg _09

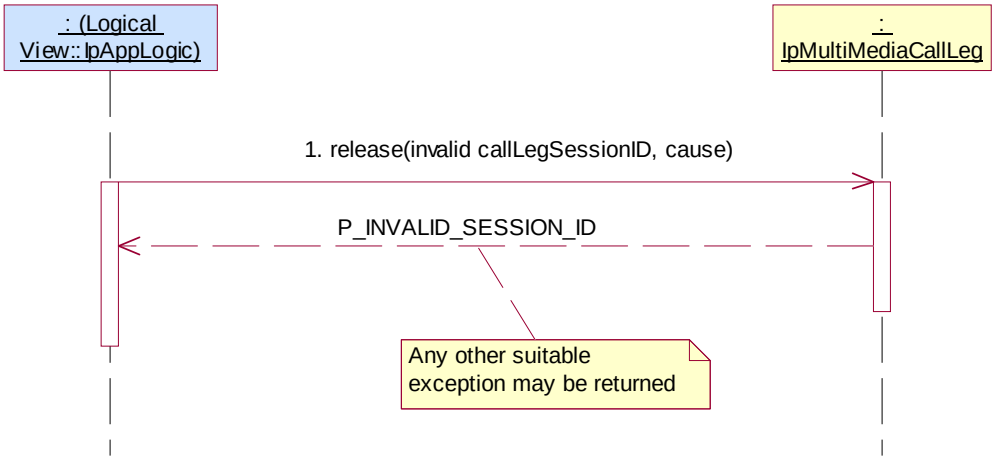
Summary: IpMultiMediaCallLeg, release**,** P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.5

Preamble: Same as CCC _ IpSubConfCall_01

Test Sequence:

1. Method call **release()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID, valid cause
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpMultiMediaCallLeg _10

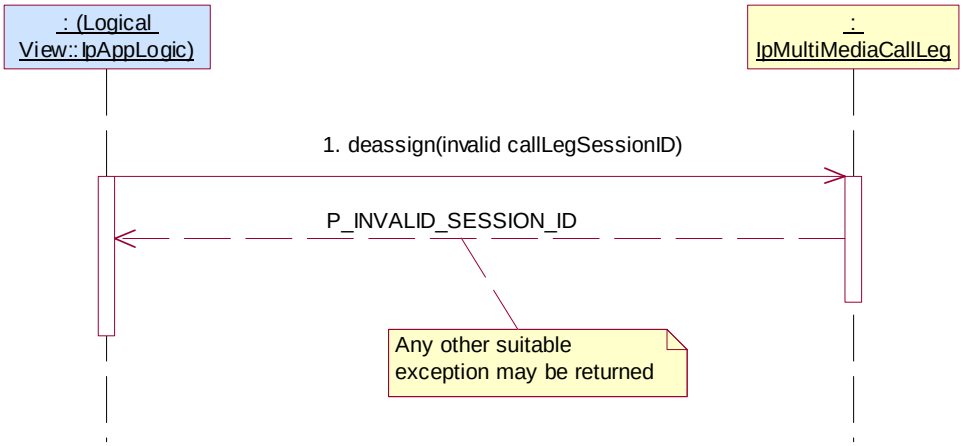
Summary: IpMultiMediaCallLeg, deassign**,** P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.5

Preamble: Same as CCC _ IpSubConfCall_01

Test Sequence:

1. Method call **deassign()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpMultiMediaCallLeg _11

Summary: IpMultiMediaCallLeg, mediaStreamAllow, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 8.3.5.

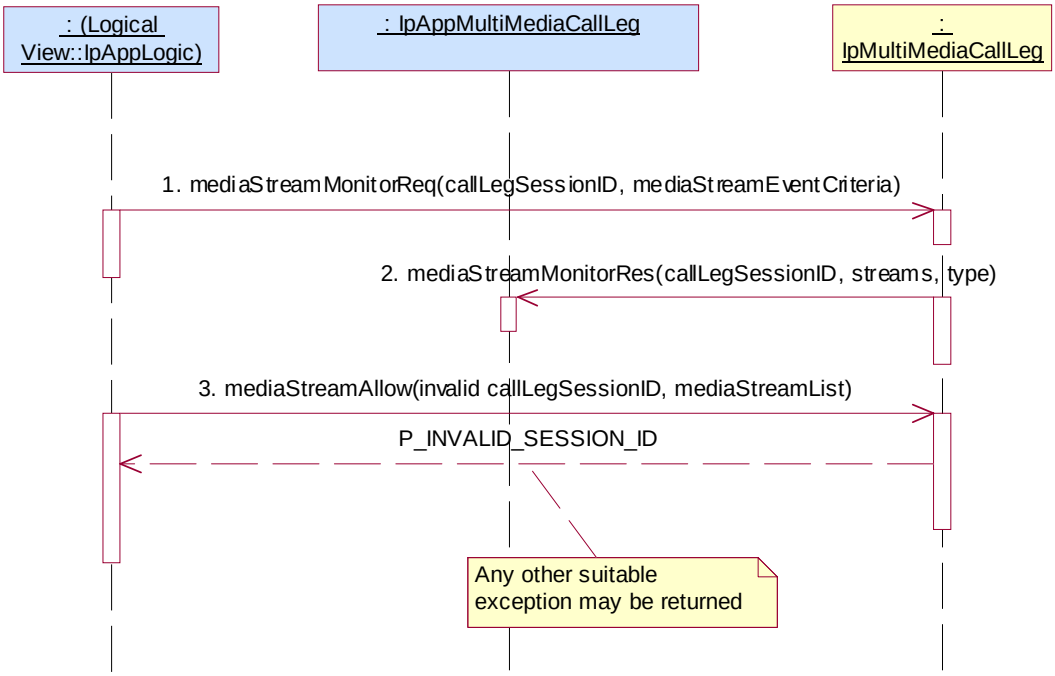
Preamble: Same as CCC _ IpSubConfCall_01

Test Sequence:

1. Method call **mediaStreamMonitorReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid mediaStreamEventCriteria
   Check: no exception is returned

2. Triggered action: cause IUT to call Method **mediaStreamMonitorRes()** method on the tester's (application)
   Parameters: callLegSessionID, streams, type

3. Method call **mediaStreamAllow()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID, valid mediaStreamList
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpMultiMediaCallLeg _12

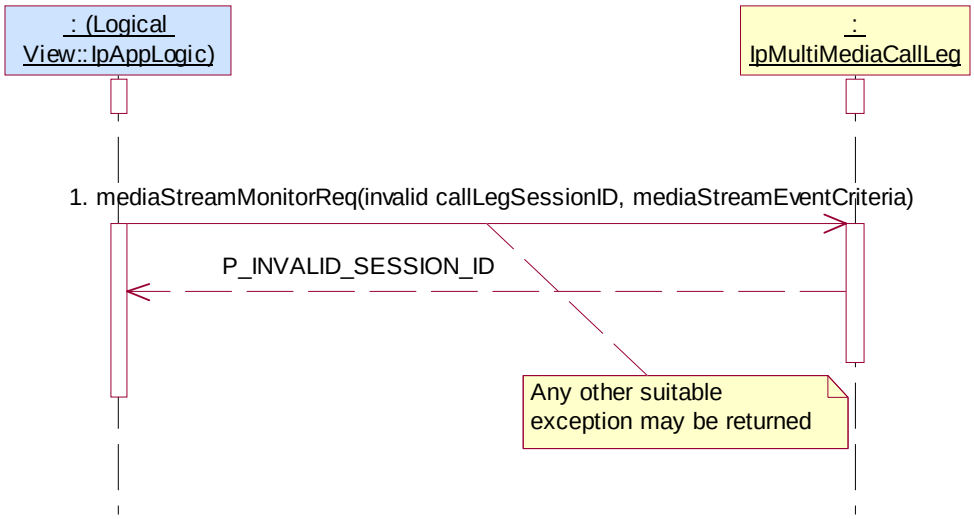
Summary: IpMultiMediaCallLeg, mediaStreamMonitorReq, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 8.3.5.

Preamble: Same as CCC _ IpSubConfCall_01

Test Sequence:

1. Method call **mediaStreamMonitorReq()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID, valid mediaStreamEventCriteria
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpMultiMediaCallLeg _13

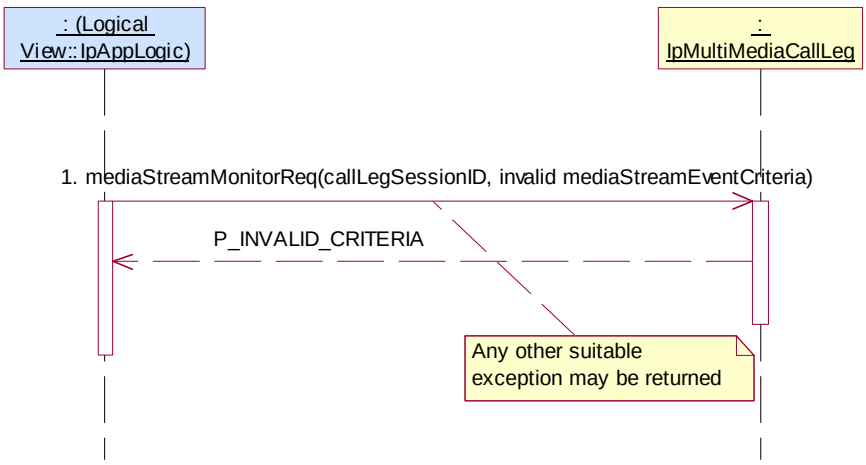
Summary: IpMultiMediaCallLeg, mediaStreamMonitorReq, P_INVALID_CRITERIA

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 8.3.5.

Preamble: Same as CCC _ IpSubConfCall_01

Test Sequence:

1. Method call **mediaStreamMonitorReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 2., valid mediaStreamEventCriteria with invalid criteria
   Check: P_INVALID_CRITERIA, or another suitable exception, is returned



5.2.4.4.3 Optional, valid behaviour

Test CCC _ IpMultiMediaCallLeg _14

Summary: IpMultiMediaCallLeg, getInfoReq, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

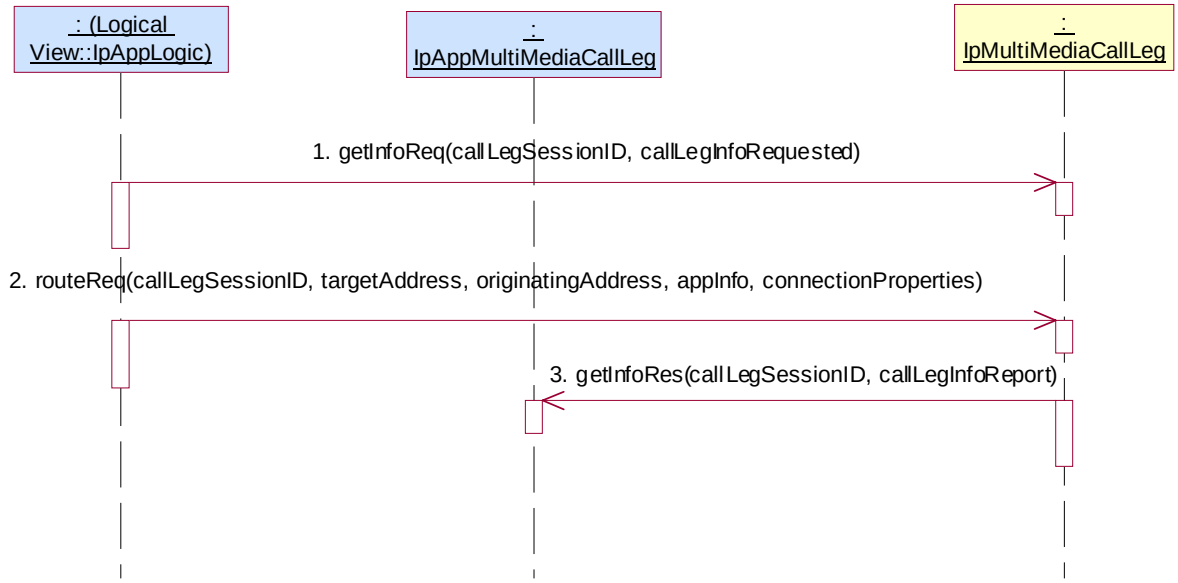
Preamble: Same as CCC _ IpConfCall _08

Test Sequence:

1. Method call **getInfoReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid callLegInfoRequested
   Check: no exception is returned

2. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

3. Triggered action: cause IUT to call **getInfoRes()** method on the tester's (Application) **IpAppMultiMediaCallLeg** interface.
   Parameters: callLegSessionID given in 1., valid callLegInfoReport.



Test CCC _ IpMultiMediaCallLeg _15

Summary: IpMultiMediaCallLeg, attachMediaReq, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Application has a valid callSessionID returned by one of the three following sequence:

1. Method call **createConference()** on IpConfCallControlManager
   Parameters: valid appConferenceCall, valid numberOfSubConferences, valid conferencePolicy, valid numberOfParticipants, valid duration
   Check: valid value of TpConfCallIdentifier is returned

2. Method call **createSubConference()** on IpConfCall
   Parameters: valid conferenceSessionID returned in 1., valid appSubConference, valid conferencePolicy
   Check: valid value of TpSubConfCallIdentifier is returned

3. Method call **getSubConferences()** on IpConfCall
   Parameters: valid conferenceSessionID returned in 1.
   Check: valid value of TpSubConfCallIdentifierSet is returned which contains TpSubConfCallIdentifier returned in 2.

4. Method call **createCallLeg()** on IpSubConfCall
   Parameters: valid callSessionID returned in 2., valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

5. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 3., valid targetAddress, valid appInfo, valid connectionProperties set to have explicit media management
   Check: no exception is returned

or

1. Method call **reserveResources()**
   Parameters: valid appInterface, valid startTime, valid numberOfParticipants, valid duration, valid conferencePolicy
   Check: valid value of TpResourceReservation is returned

2. Triggered action: cause IUT to call **conferenceCreated()** on Tester's (application's) IpAppConfCallControlManager interface
   Parameters: valid conferenceCall.

3. Method call **getSubConferences()** on IpConfCall
   Parameters: valid conferenceSessionID returned in 1.
   Check: valid value of TpSubConfCallIdentifierSet is returned which contains TpSubConfCallIdentifier returned in 2.

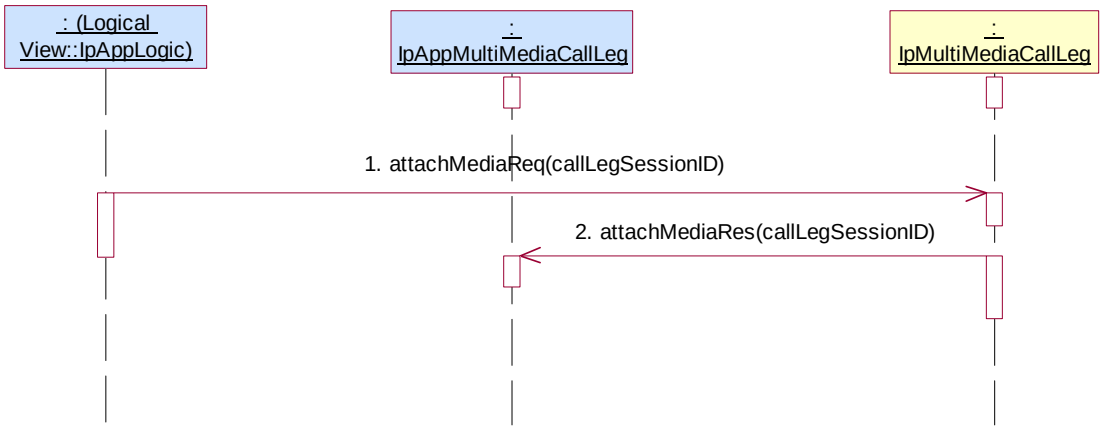
4. Method call **createCallLeg()** on IpSubConfCall
   Parameters: valid callSessionID reported in 2., valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

5. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 3., valid targetAddress, valid appInfo, valid connectionProperties set to have explicit media management
   Check: no exception is returned

Test Sequence:

1. Method call **attachMediaReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble.
   Check: no exception is returned

2 Triggered action: cause IUT to call **attachMediaRes()** method on the tester's (Application) **IpAppMultiMediaCallLeg** interface.
   Parameters: callLegSessionID



Test CCC _ IpMultiMediaCallLeg _16

Summary: IpMultiMediaCallLeg, detachMediaReq, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as CCC _ IpMultiMediaCallLeg_15

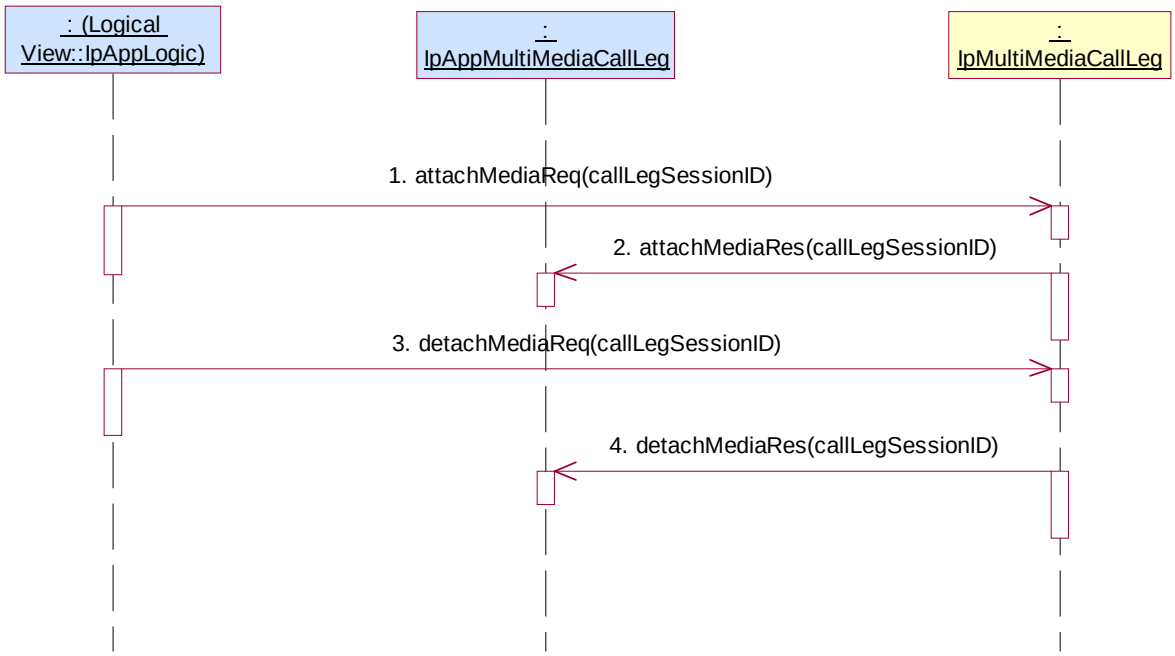
Test Sequence:

1. Method call **attachMediaReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble.
   Check: no exception is returned

2 Triggered action: cause IUT to call **attachMediaRes()** method on the tester's (Application) **IpAppMultiMediaCallLeg** interface.
   Parameters: callLegSessionID

3. Method call **detachMediaReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble.
   Check: no exception is returned

4 Triggered action: cause IUT to call **detachMediaRes()** method on the tester's (Application) **IpAppMultiMediaCallLeg** interface.
   Parameters: callLegSessionID



Test CCC _ IpMultiMediaCallLeg _17

Summary: IpMultiMediaCallLeg, getCurrentDestinationAddress, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as CCC _ IpSubConfCall_01

Test Sequence:

1. Method call **getCurrentDestinationAddress()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble.
   Check: valid value of TpAddress is returned

Test CCC _ IpMultiMediaCallLeg _18

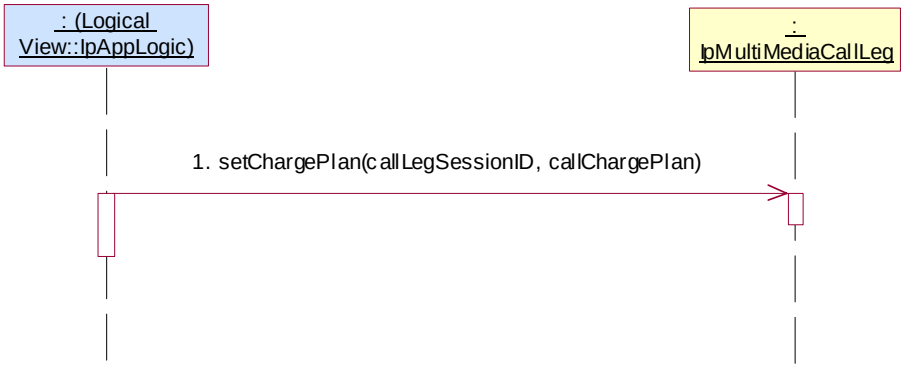
Summary: IpMultiMediaCallLeg, setChargePlan, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as CCC _ IpConfCall _08

Test Sequence:

1. Method call **setChargePlan()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid callChargePlan
   Check: no exception is returned



Test CCC _ IpMultiMediaCallLeg _19

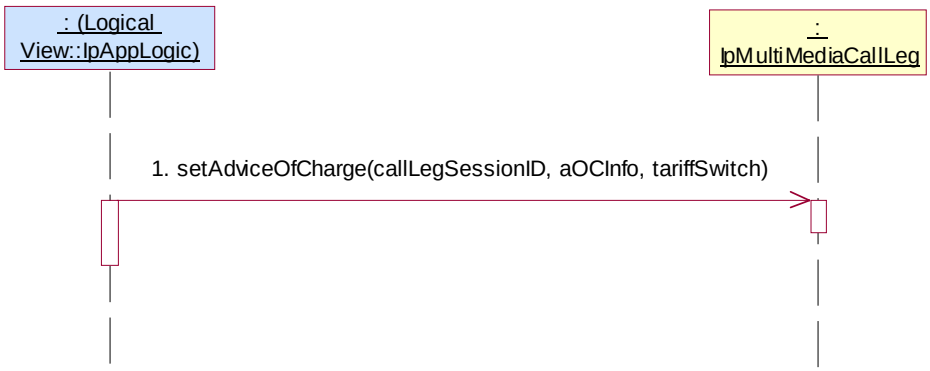
Summary: IpMultiMediaCallLeg, setAdviceOfCharge, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as CCC _ IpConfCall _08

Test Sequence:

1. Method call **setAdviceOfCharge()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid aOCInfo, valid tariffSwitch
   Check: no exception is returned



Test CCC _ IpMultiMediaCallLeg _20

Summary: IpMultiMediaCallLeg, superviseReq, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

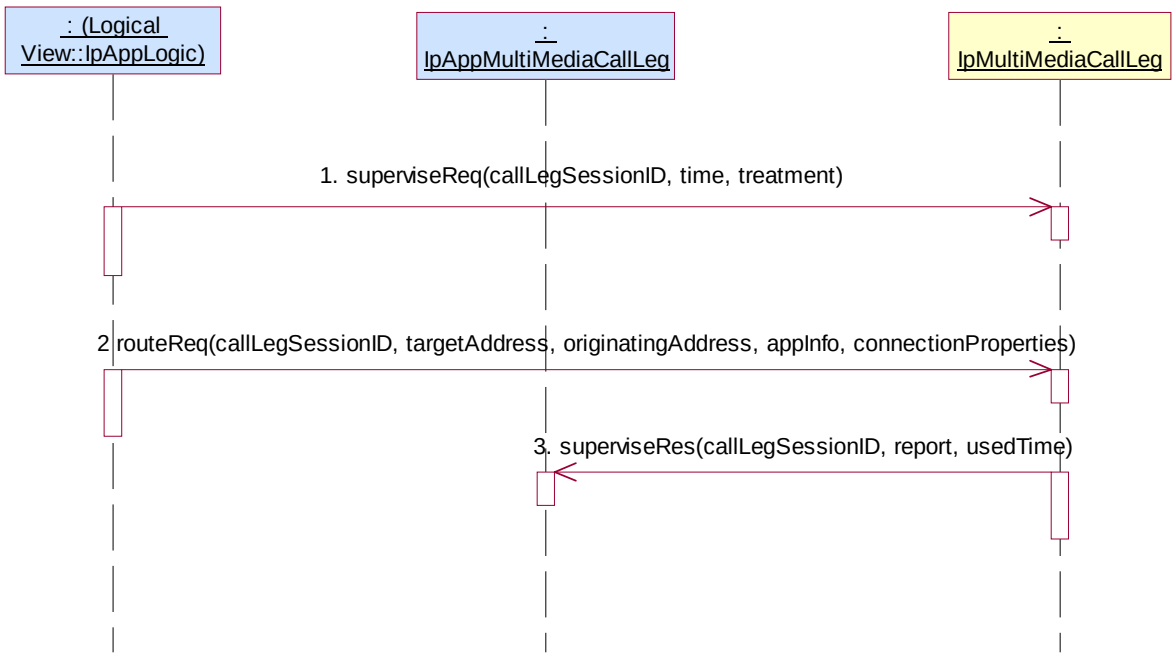
Preamble: Same as CCC _ IpConfCall _08

Test Sequence:

1. Method call **superviseReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid time, valid treatment
   Check: no exception is returned

2. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

3. Triggered action: cause IUT to call **superviseRes()** method on the tester's (Application) **IpAppMultiMediaCallLeg** interface.
   Parameters: callLegSessionID, report, usedTime



Test CCC _ IpMultiMediaCallLeg _21

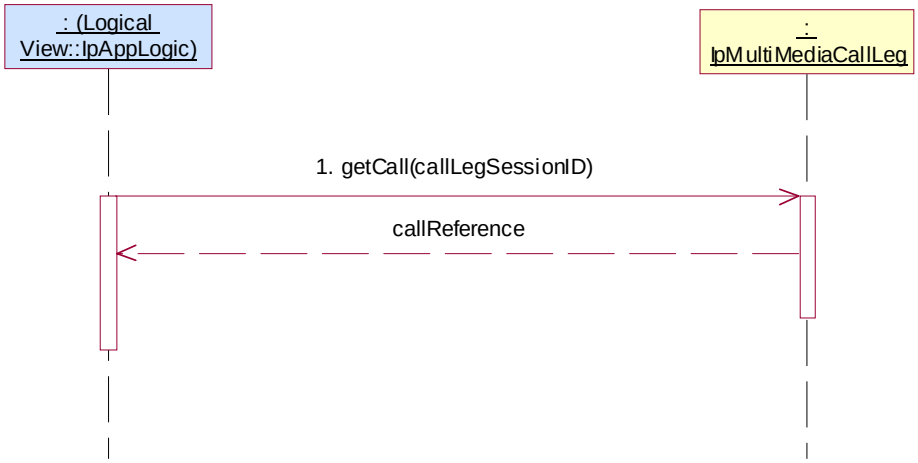
Summary: IpMultiMediaCallLeg, getCall, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as CCC _ IpConfCall _08

Test Sequence:

1. Method call **getCall()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble.
   Check: valid TpMultiPartyCallIdentifier is returned



Test CCC _ IpMultiMediaCallLeg _22

Summary: IpMultiMediaCallLeg, getCall, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

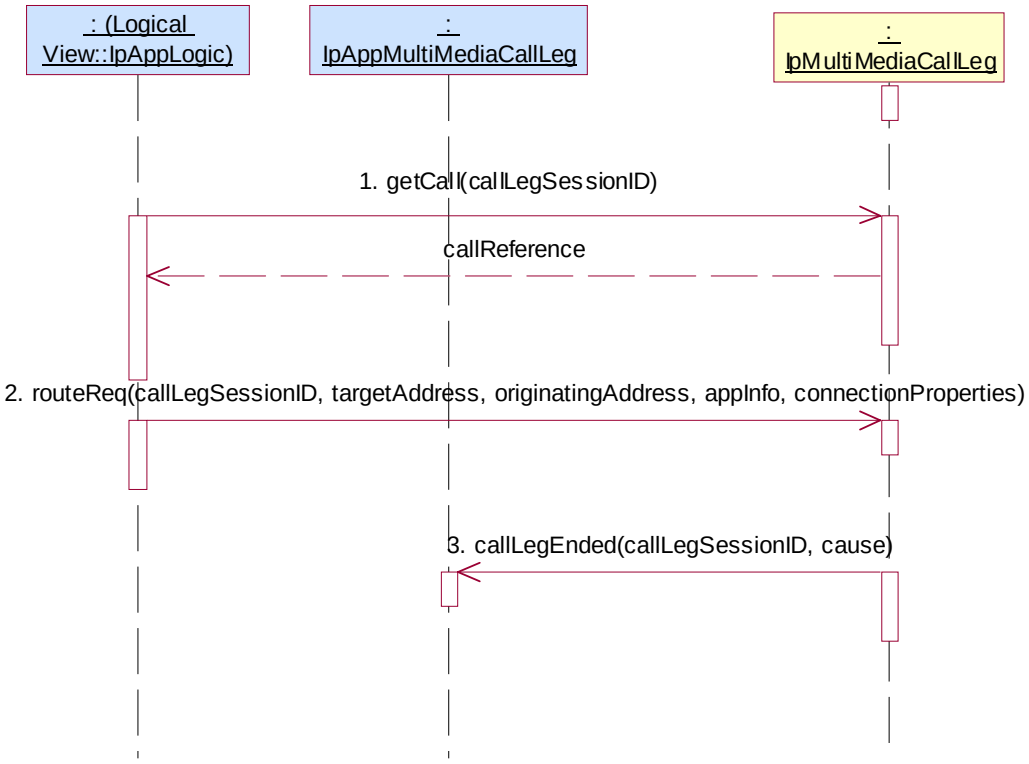
Preamble: Same as CCC _ IpConfCall _08

Test Sequence:

1. Method call **getCall()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble.
   Check: valid TpMultiPartyCallIdentifier is returned

2. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

3. Triggered action: cause IUT to call **callLegEnded()** method on the tester's (Application) **IpAppMultiMediaCallLeg** interface.
   Parameters: callLegSessionID, cause



5.2.4.4.4 Optional, invalid behaviour

Test CCC _ IpMultiMediaCallLeg _23

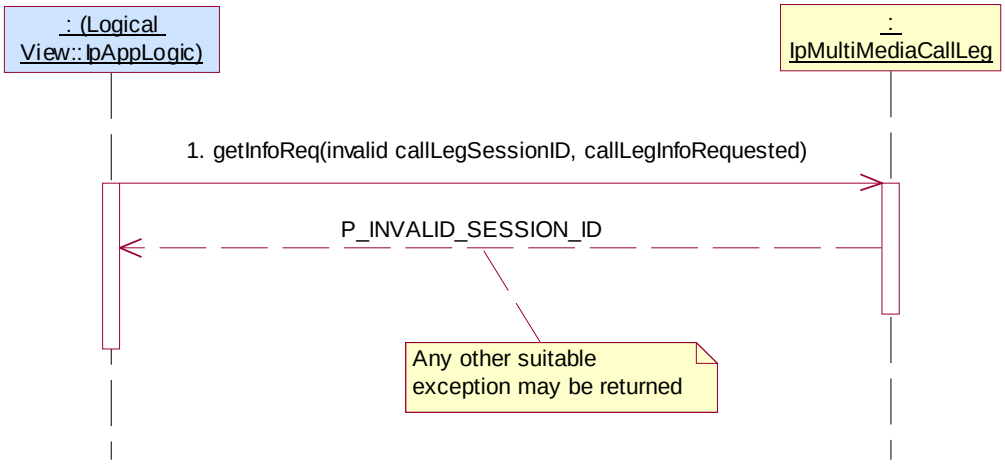
Summary: IpMultiMediaCallLeg, getInfoReq, : P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.5.

Preamble: Same as CCC _ IpConfCall _08

Test Sequence:

1. Method call **getInfoReq()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID, valid callLegInfoRequested
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpMultiMediaCallLeg _24

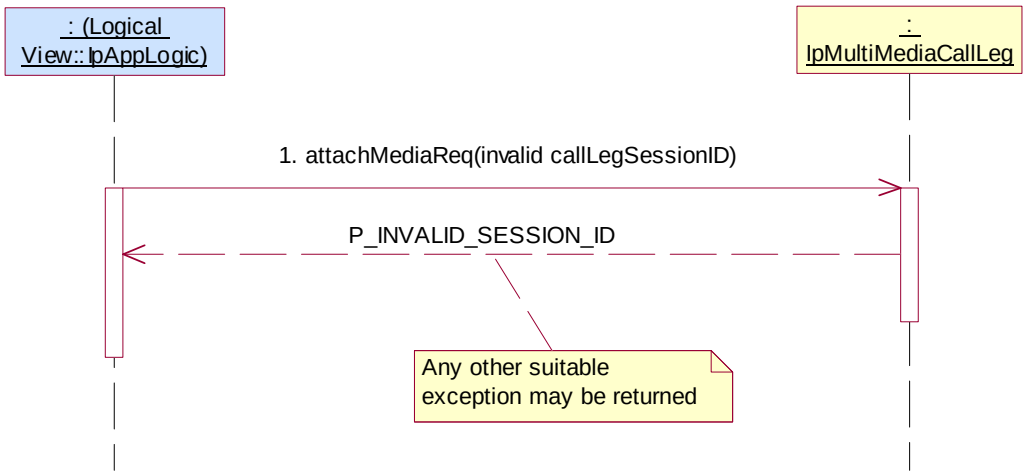
Summary: IpMultiMediaCallLeg, attachMediaReq, : P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.5.

Preamble: Same as CCC _ IpMultiMediaCallLeg_15

Test Sequence:

1. Method call **attachMediaReq()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpMultiMediaCallLeg _25

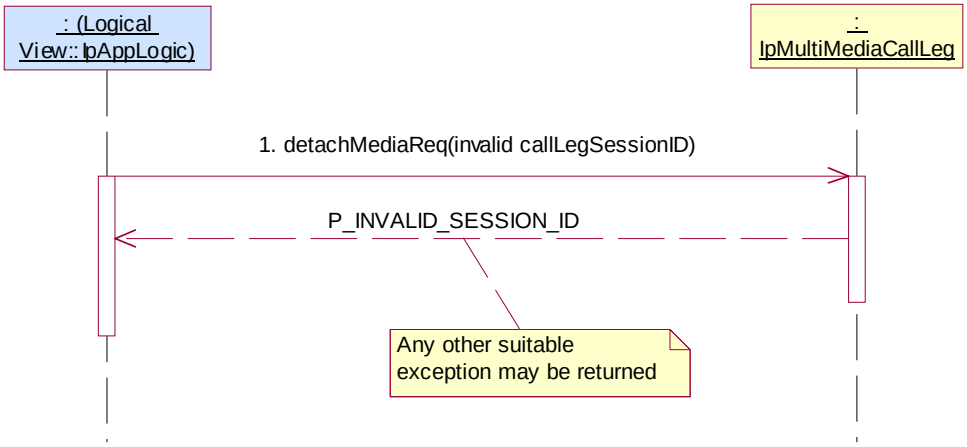
Summary: IpMultiMediaCallLeg, detachMediaReq, : P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.5.

Preamble: Same as CCC _ IpSubConfCall_01

Test Sequence:

1. Method call **detachMediaReq()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpMultiMediaCallLeg _26

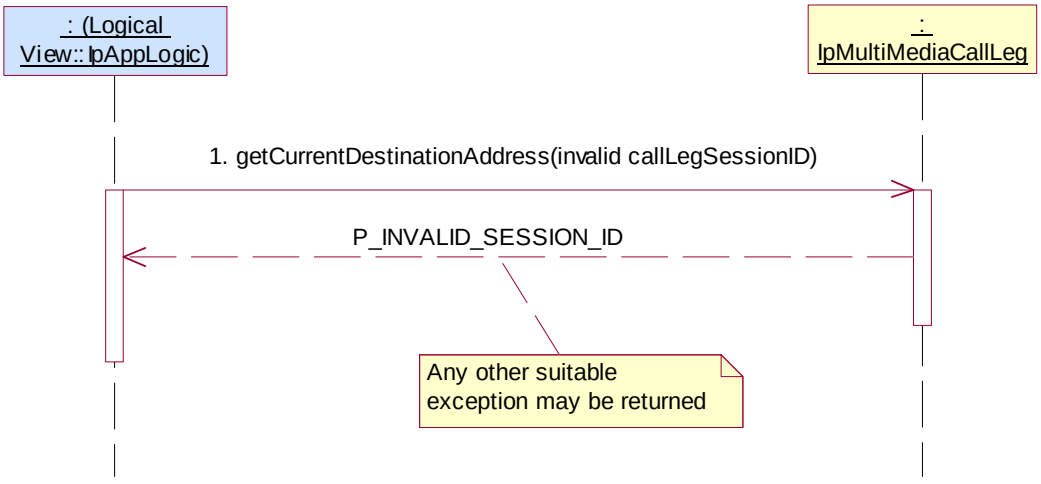
Summary: IpMultiMediaCallLeg, getCurrentDestinationAddress, : P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.5.

Preamble: Same as CCC _ IpSubConfCall_01

Test Sequence:

1. Method call **getCurrentDestinationAddress()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpMultiMediaCallLeg _27

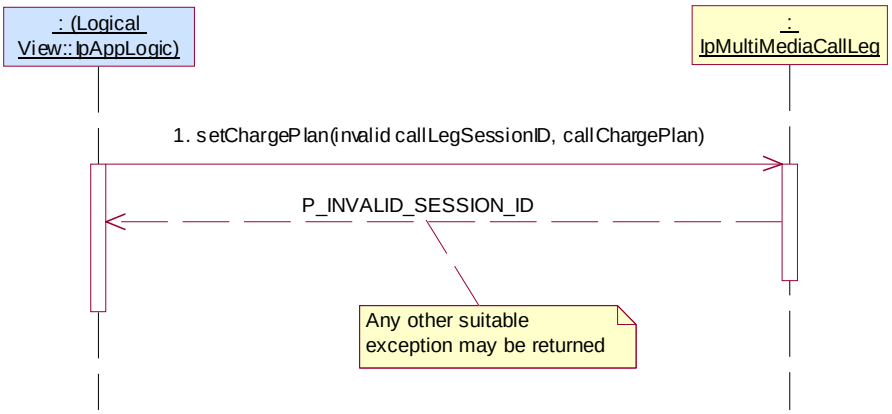
Summary: IpMultiMediaCallLeg, setChargePlan, : P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.5

Preamble: Same as CCC _ IpConfCall _08

Test Sequence:

1. Method call **setChargePlan()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID, valid callChargePlan
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpMultiMediaCallLeg _28

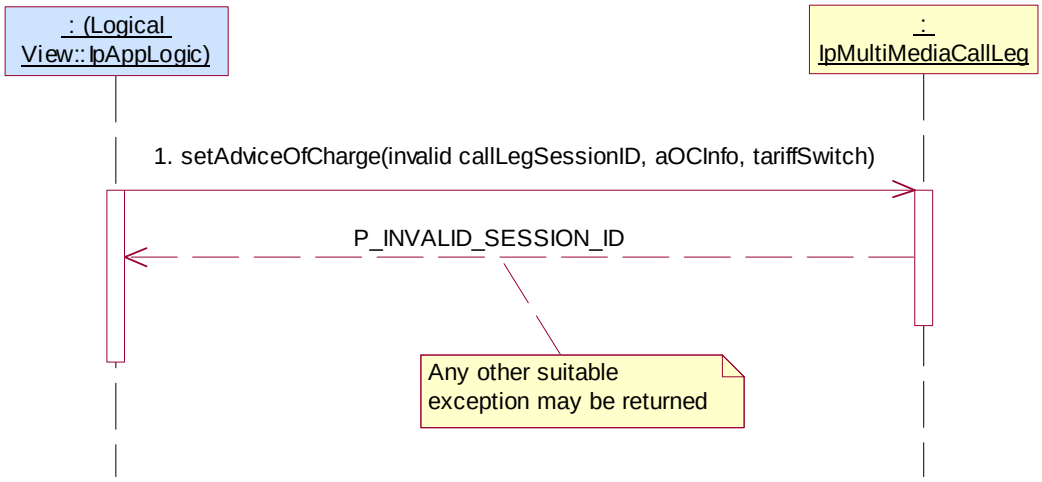
Summary: IpMultiMediaCallLeg, setAdviceOfCharge, : P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.5

Preamble: Same as CCC _ IpConfCall _08

Test Sequence:

1. Method call **setAdviceOfCharge()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID, valid aOCInfo, valid tariffSwitch
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpMultiMediaCallLeg _29

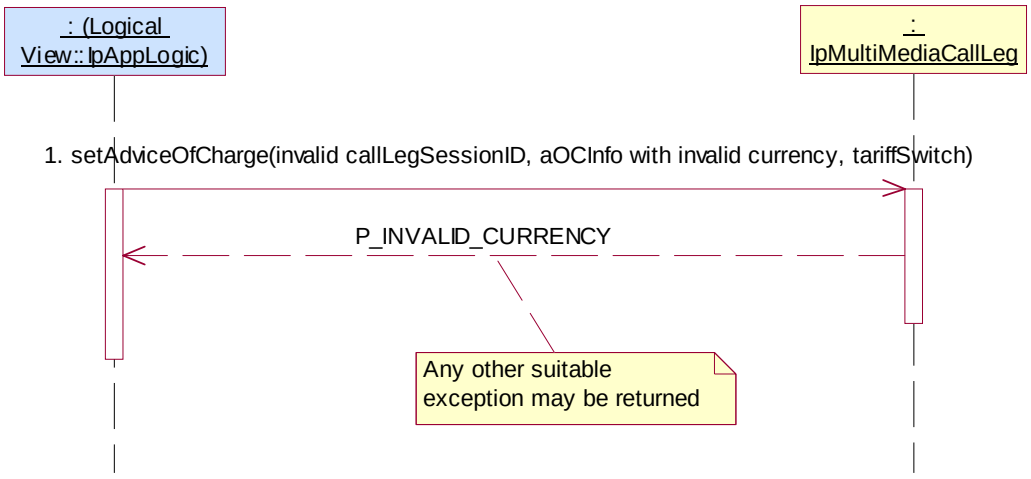
Summary: IpMultiMediaCallLeg, setAdviceOfCharge, P_INVALID_CURRENCY

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as CCC _ IpConfCall _08

Test Sequence:

1. Method call **setAdviceOfCharge()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, aOCInfo with invalid currency, valid tariffSwitch
   Check: P_INVALID_CURRENCY, or another suitable exception, is returned



Test CCC _ IpMultiMediaCallLeg _30

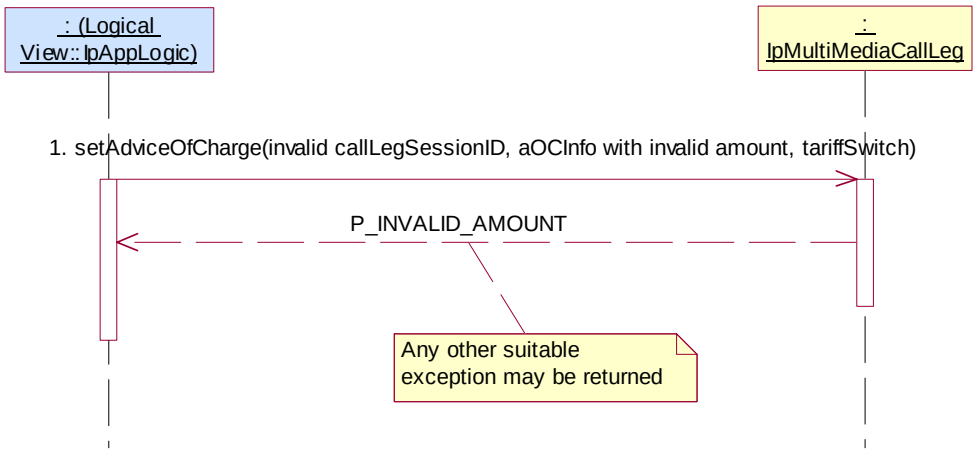
Summary: IpMultiMediaCallLeg, setAdviceOfCharge, P_INVALID_AMOUNT

Reference: ES 201 915-4 [1], clause 7.3.1, 7.3.3 and , 7.3.5.

Preamble: Same as CCC _ IpConfCall _08

Test Sequence:

1. Method call **setAdviceOfCharge()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, aOCInfo with invalid amount, valid tariffSwitch
   Check: P_INVALID_AMOUNT, or another suitable exception, is returned



Test CCC _ IpMultiMediaCallLeg _31

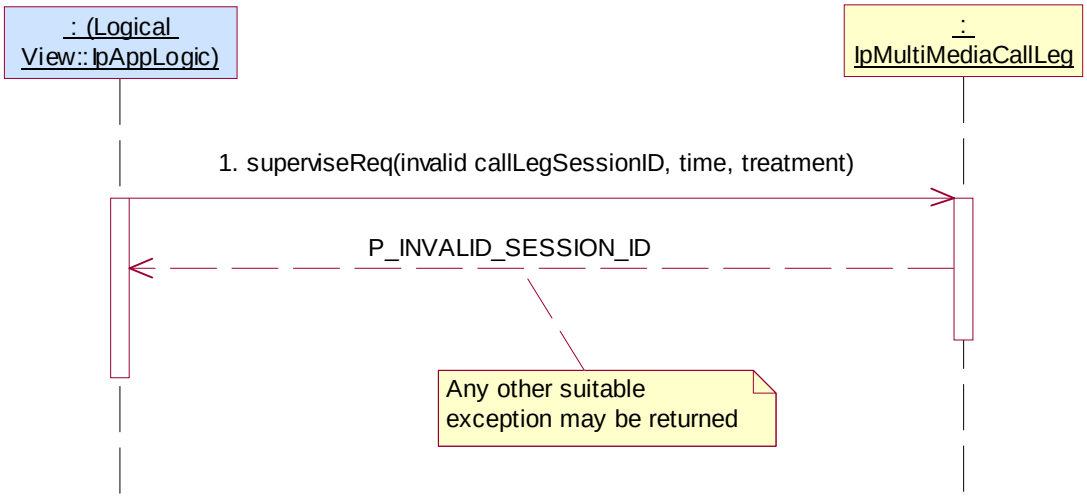
Summary: IpMultiMediaCallLeg, superviseReq, : P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.5

Preamble: Same as CCC _ IpConfCall _08

Test Sequence:

1. Method call **superviseReq()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID, valid time, valid treatment
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpMultiMediaCallLeg _32

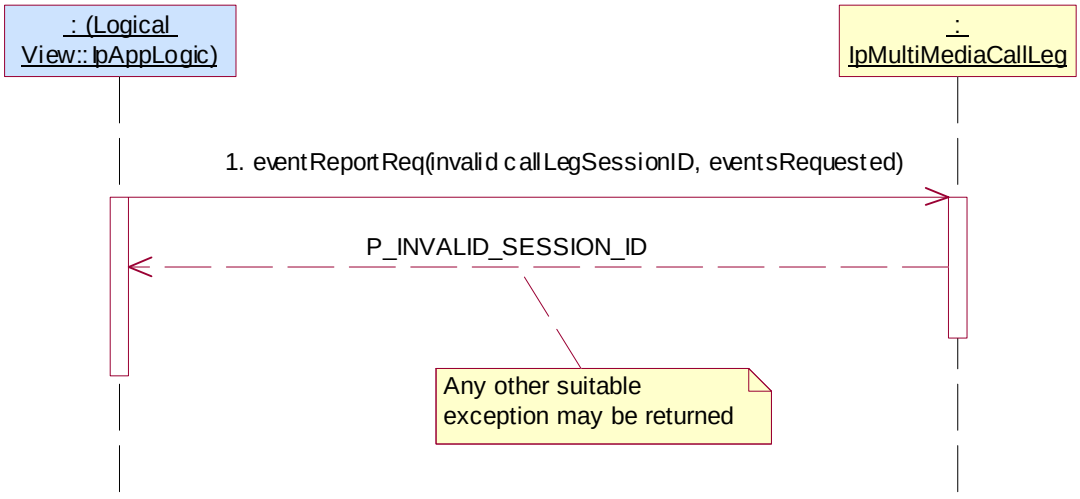
Summary: IpMultiMediaCallLeg, eventReportReq**,** P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.5

Preamble: Same as CCC _ IpConfCall _08

Test Sequence:

1. Method call **eventReportReq()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID, valid eventsRequested
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpMultiMediaCallLeg _33

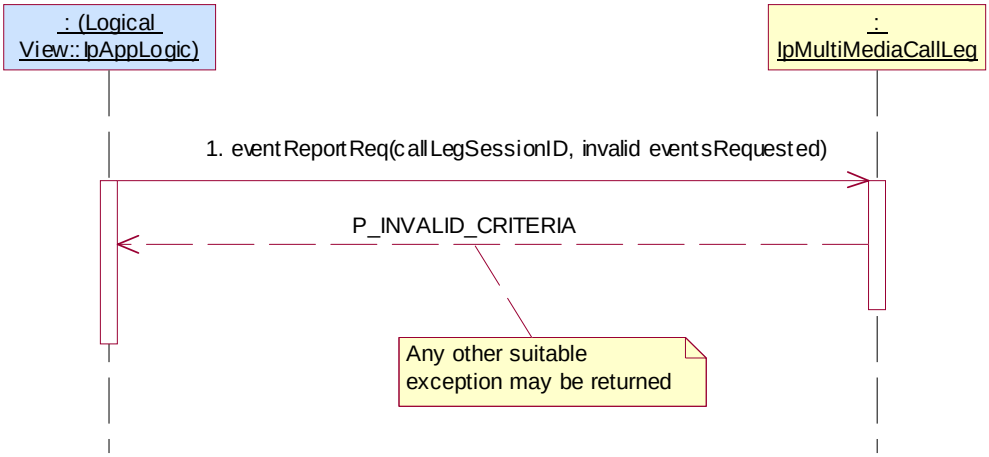
Summary: IpMultiMediaCallLeg, eventReportReq, P_INVALID_CRITERIA

Reference: ES 201 915-4 [1], clause 7.3.1, 7.3.3 and 7.3.5.

Preamble: Same as CCC _ IpConfCall _08

Test Sequence:

1. Method call **eventReportReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in preamble, invalid eventsRequested
   Check: P_INVALID_CRITERIA, or another suitable exception, is returned



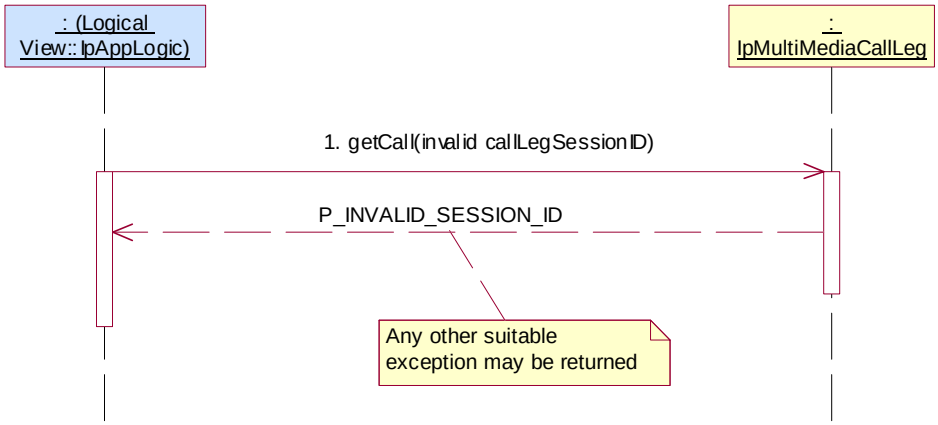
Test CCC _ IpMultiMediaCallLeg _34

Summary: IpMultiMediaCallLeg, getCall**,** P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 7.3.5

Preamble: Same as CCC _ IpConfCall _08

1. Method call **getCall()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



Test CCC _ IpMultiMediaCallLeg _35

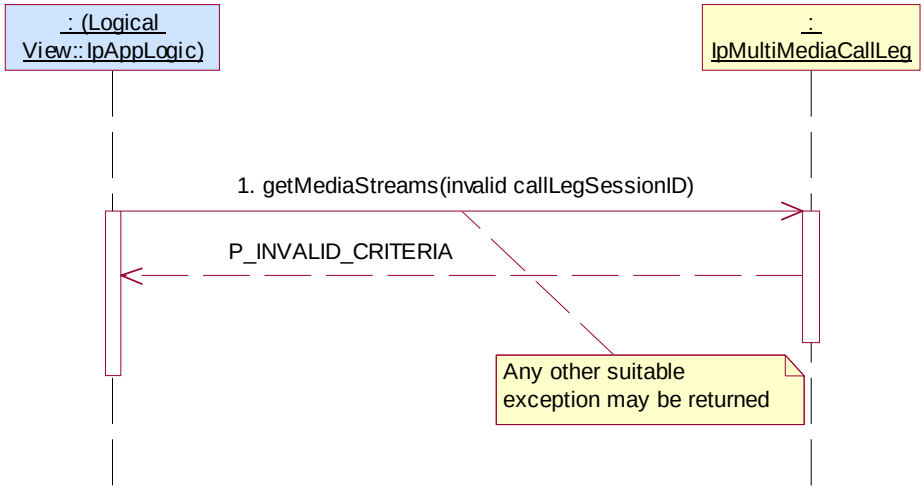
Summary: IpMultiMediaCallLeg, getMediaStreams, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 8.3.5.

Preamble: Same as CCC _ IpSubConfCall_01

Test Sequence:

1. Method call **getMediaStreams()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



5.2.4.5 IpMultiMediaStream

5.2.4.5.1 Mandatory, valid behaviour

Test CCC_ IpMultiMediaStream _01

Summary: IpMultiMediaStream, all methods mandatory, successful

Reference: ES 201 915-4 [1], clauses 7.3.1, 7.3.3, 8.3.5 and 8.3.7.

Preamble: Application has a valid callSessionID returned by one of the three following sequence:

1. Method call **createConference()** on IpConfCallControlManager
   Parameters: valid appConferenceCall, valid numberOfSubConferences, valid conferencePolicy, valid numberOfParticipants, valid duration
   Check: valid value of TpConfCallIdentifier is returned

2. Method call **getSubConferences()**
   Parameters: valid conferenceSessionID
   Check: valid value of TpSubConfCallIdentifierSet is returned

3. Method call **createCallLeg()** on IpSubConfCall
   Parameters: valid callSessionID returned in 2., valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

4. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 2, valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

or

3. Method call **createAndRouteCallLegReq()** on IpSubConfCall
   Parameters: valid callSessionID returned in 2., valid eventsRequested, valid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: valid value of TpCallLegIdentifier

4. Method call **mediaStreamMonitorReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 3., valid mediaStreamEventCriteria
   Check: no exception is returned

5. Triggered action: cause IUT to call Method **mediaStreamMonitorRes()** method on the tester's (application)
   Parameters: callLegSessionID, streams, type

6. Method call **mediaStreamAllow()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 3., valid mediaStreamList
   Check: no exception is returned

or

1. Method call **reserveResources()**
   Parameters: valid appInterface, valid startTime, valid numberOfParticipants, valid duration, valid conferencePolicy
   Check: valid value of TpResourceReservation is returned

2. Triggered action: cause IUT to call **conferenceCreated()** on Tester's (application's) IpAppConfCallControlManager interface
   Parameters: valid conferenceCall.

3. Method call **getSubConferences()**
   Parameters: valid conferenceSessionID
   Check: valid value of TpSubConfCallIdentifierSet is returned

4. Method call **createCallLeg()** on IpSubConfCall
   Parameters: valid callSessionID reported in 2., valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

5. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 2, valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

or

4. Method call **createAndRouteCallLegReq()** on IpSubConfCall
   Parameters: valid callSessionID returned in 2., valid eventsRequested, valid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: valid value of TpCallLegIdentifier

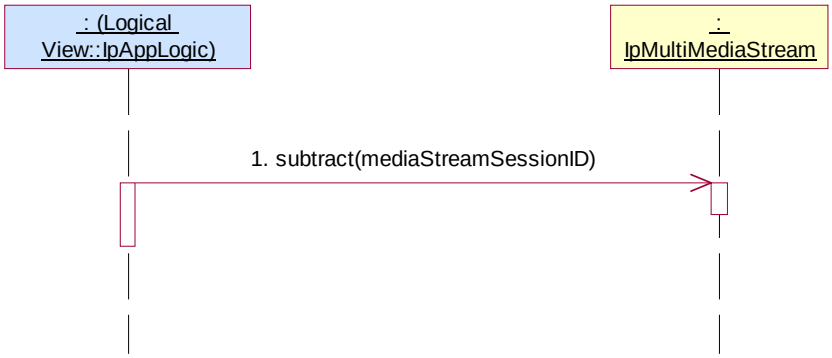
5. Method call **mediaStreamMonitorReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 3., valid mediaStreamEventCriteria
   Check: no exception is returned

6. Triggered action: cause IUT to call Method **mediaStreamMonitorRes()** method on the tester's (application)
   Parameters: callLegSessionID, streams, type

7. Method call **mediaStreamAllow()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 3., valid mediaStreamList
   Check: no exception is returned

Test Sequence:

1. Method call **subtract()** on IpMultiMediaStream
   Parameters: valid mediaStreamSessionID from TpMediaStreamSet returned in preamble.
   Check: no exception is returned



5.2.4.5.2 Mandatory, invalid behaviour

Test CCC_ IpMultiMediaStream _02

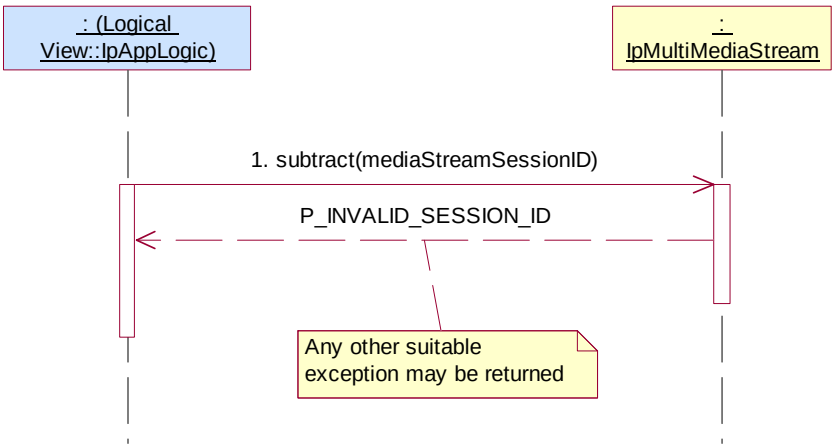
Summary: IpMultiMediaStream, subtract, P_INVALID_SESSION_ID

Reference: ES 201 915-4 [1], clause 8.3.7.

Preamble: Same as Test CCC_ IpMultiMediaStream _01

Test Sequence:

1. Method call **subtract()** on IpMultiMediaStream
   Parameters: invalid mediaStreamSessionID
   Check: P_INVALID_SESSION_ID, or another suitable exception, is returned



History

Document history			
V1.1.1	June 2003	Membership Approval Procedure	MV 20030801: 2003-06-03 to 2003-08-01