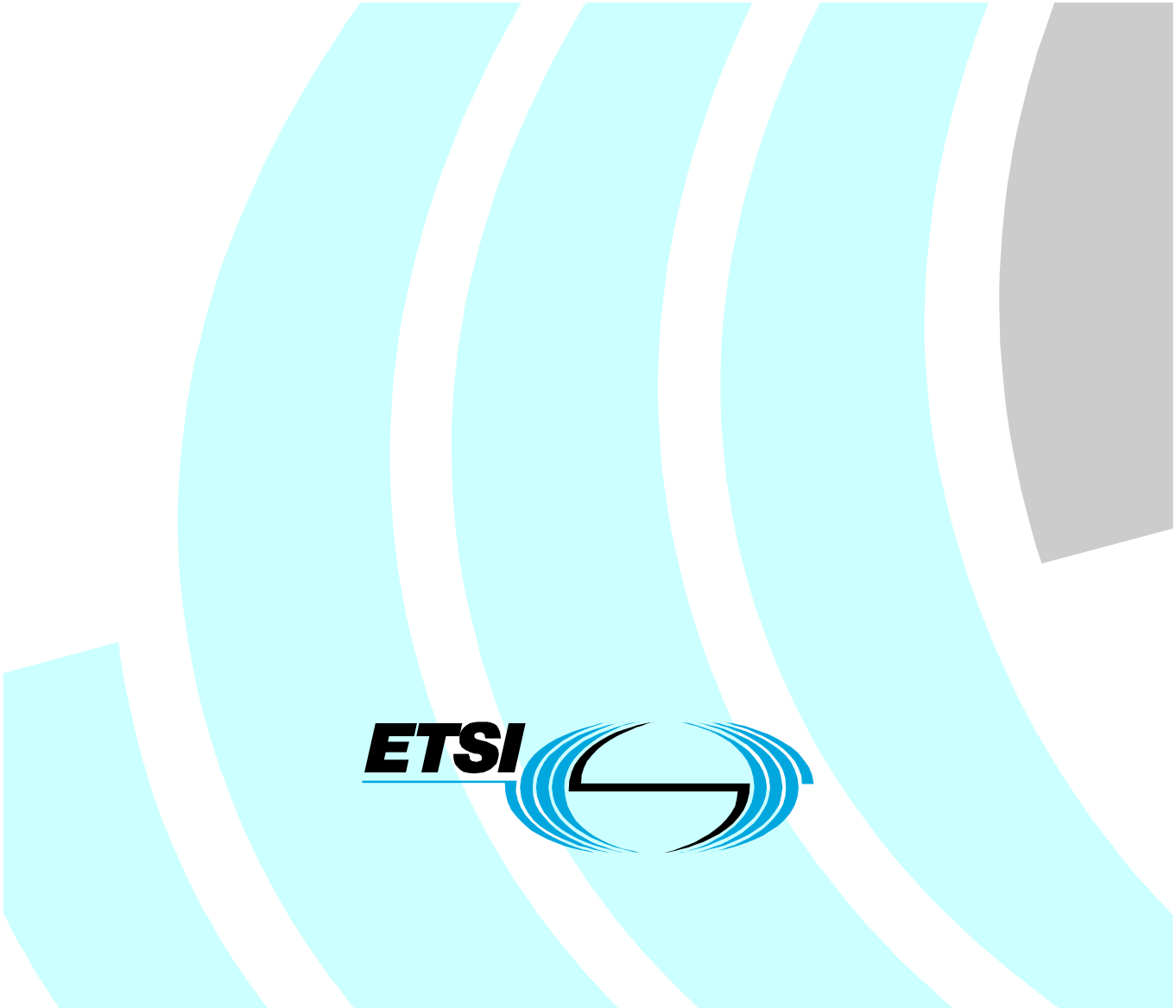
Final draft ETSI ES 202 388-4 V1.1.1 (2005-01)

ETSI Standard

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



Reference
DES/TISPAN-06004-04-OSA

Keywords
API, OSA, TSS&TP

ETSI

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

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

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

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 Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN), 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 [10].

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 202 915-4 sub-parts 1 [1] through to 5 [5] in compliance with the relevant requirements, and in accordance with the relevant guidance given in ISO/IEC 9646-2 [8] and ETS 300 406 [9].

2 References

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

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

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

[1] ETSI ES 202 915-4-1: "Open Service Access (OSA); Application Programming Interface (API); Part 4: Call Control; Sub-part 1: Call Control Common Definitions (Parlay 4)".

[2] ETSI ES 202 915-4-2: "Open Service Access (OSA); Application Programming Interface (API); Part 4: Call Control; Sub-part 2: Generic Call Control SCF (Parlay 4)".

[3] ETSI ES 202 915-4-3: "Open Service Access (OSA); Application Programming Interface (API); Part 4: Call Control; Sub-part 3: Multi-Party Call Control SCF (Parlay 4)".

[4] ETSI ES 202 915-4-4: "Open Service Access (OSA); Application Programming Interface (API); Part 4: Call Control; Sub-part 4: Multi-Media Call Control SCF (Parlay 4)".

[5] ETSI ES 202 915-4-5: "Open Service Access (OSA); Application Programming Interface (API); Part 4: Call Control; Sub-part 5: Conference Call Control SCF (Parlay 4)".

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

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

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

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

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

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in ES 202 915-4-1 [1], ES 202 915-4-2 [2], ES 202 915-4-3 [3], ES 202 915-4-4 [4], ES 202 915-4-5 [5], ISO/IEC 9646-1 [7], ISO/IEC 9646-2 [8] and the following apply:

abstract test case: Refer to ISO/IEC 9646-1 [7].

Abstract Test Method (ATM): Refer to ISO/IEC 9646-1 [7].

Abstract Test Suite (ATS): Refer to ISO/IEC 9646-1 [7].

Implementation Conformance Statement (ICS): Refer to ISO/IEC 9646-1 [7].

ICS proforma: Refer to ISO/IEC 9646-1 [7].

Implementation Under Test (IUT): Refer to ISO/IEC 9646-1 [7].

Implementation eXtra Information for Testing (IXIT): Refer to ISO/IEC 9646-1 [7].

IXIT proforma: Refer to ISO/IEC 9646-1 [7].

Lower Tester (LT): Refer to ISO/IEC 9646-1 [7].

Test Purpose (TP): Refer to ISO/IEC 9646-1 [7].

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
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
TP	Test Purpose
TSS	Test Suite Structure

4 Test Suite Structure (TSS) - SCF

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) - SCF

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:	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 202 915-4-1 [1], ES 202 915-4-2 [2], ES 202 915-4-3 [3], ES 202 915-4-4 [4] and ES 202 915-4-5 [5].

5.1.3 Test strategy

As the base standards ES 202 915-4-1 [1], ES 202 915-4-2 [2], ES 202 915-4-3 [3], ES 202 915-4-4 [4] and ES 202 915-4-5 [5] contain no explicit requirements for testing, the TPs were generated as a result of an analysis of the base standards and the ICS specification ES 202 363 [6].

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 [9]).

5.2 TPs for the Call Control SCF

All ICS items referred to in this clause are as specified in ES 202 363 [6] unless indicated otherwise by another numbered reference.

All parameters specified in method calls are valid unless specified.

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

5.2.1 Generic Call Control

The TPs in this clause are based on ES 202 915-4-2 [2].

5.2.1.1 IpCallControlManager

5.2.1.1.1 Mandatory, valid behaviour

Test GCC_IPCALLCONTROLMANAGER _01

Summary: IpCallControlManager, all mandatory methods, successful

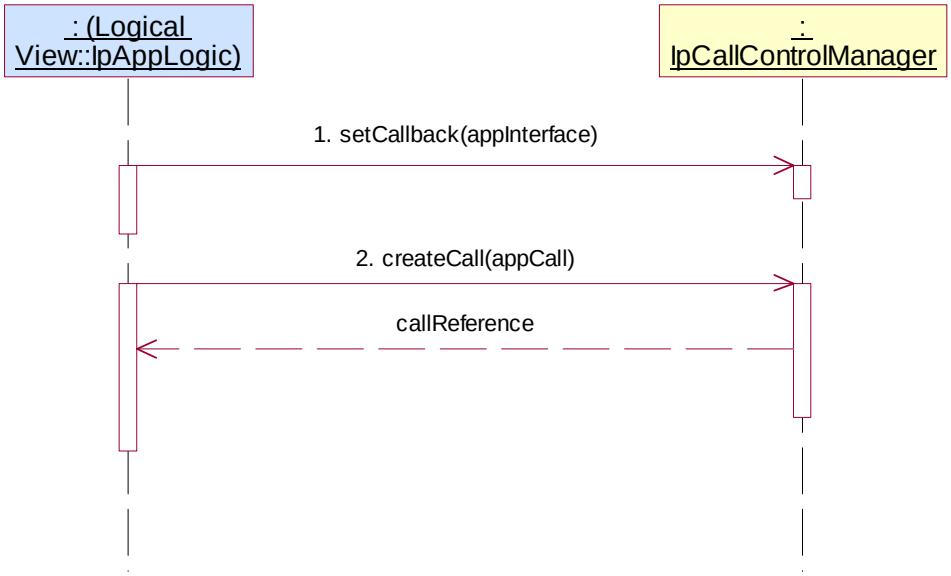
Reference: ES 202 915-4-2 [2], clause 6.1

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 202 915-4-2 [2], clause 6.1

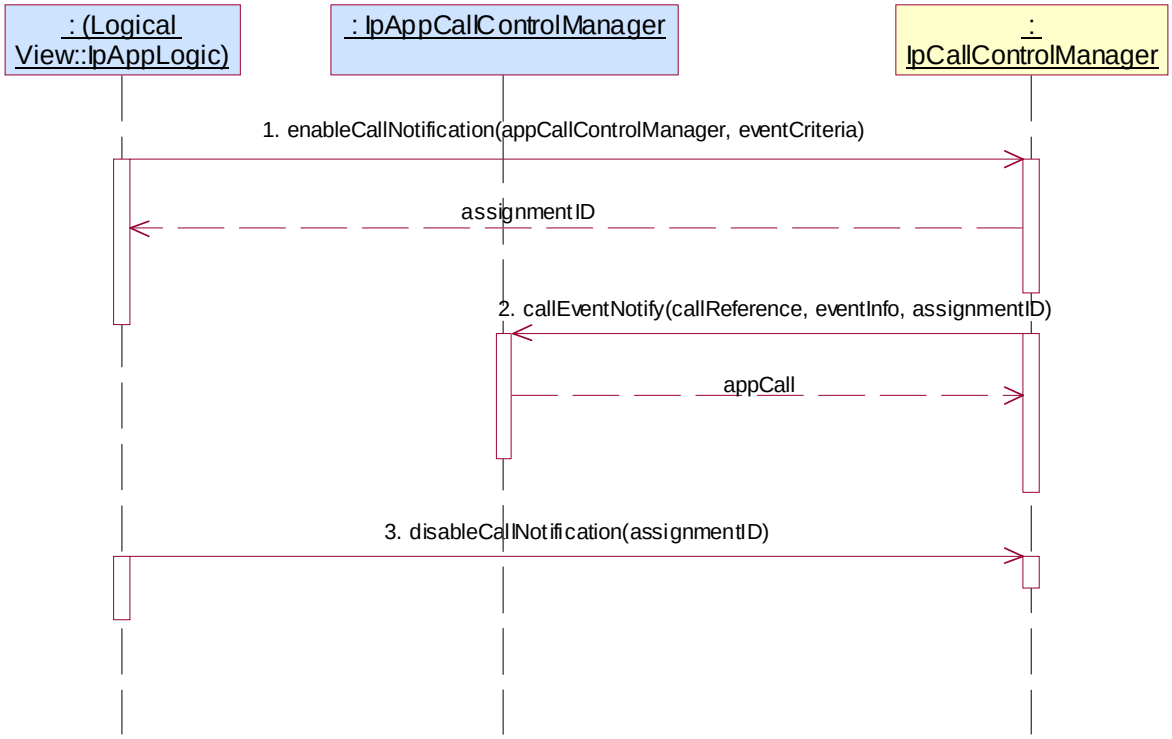
Condition: enableCallNotification method is supported.

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 202 915-4-2 [2], clause 6.1

Condition: enableCallNotification method is supported.

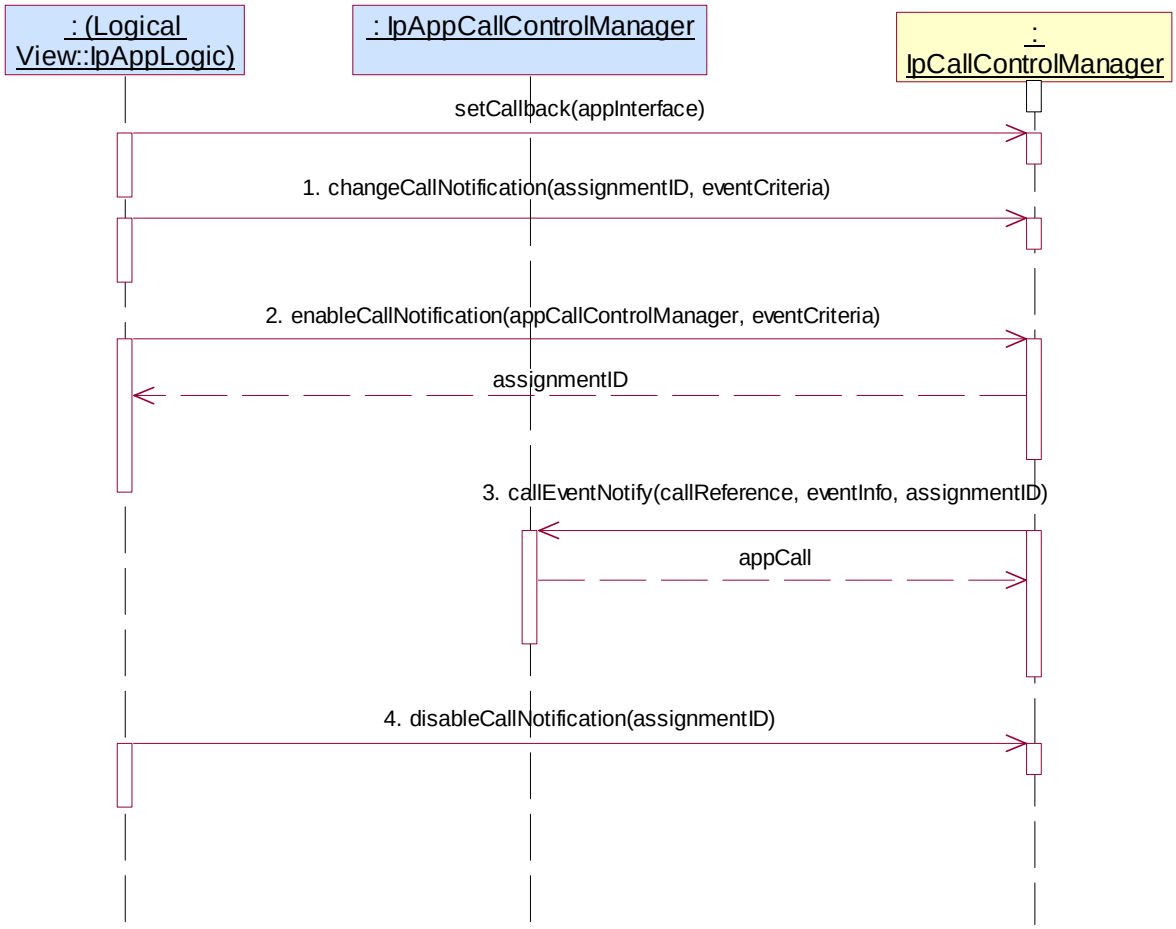
Test Sequence:

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

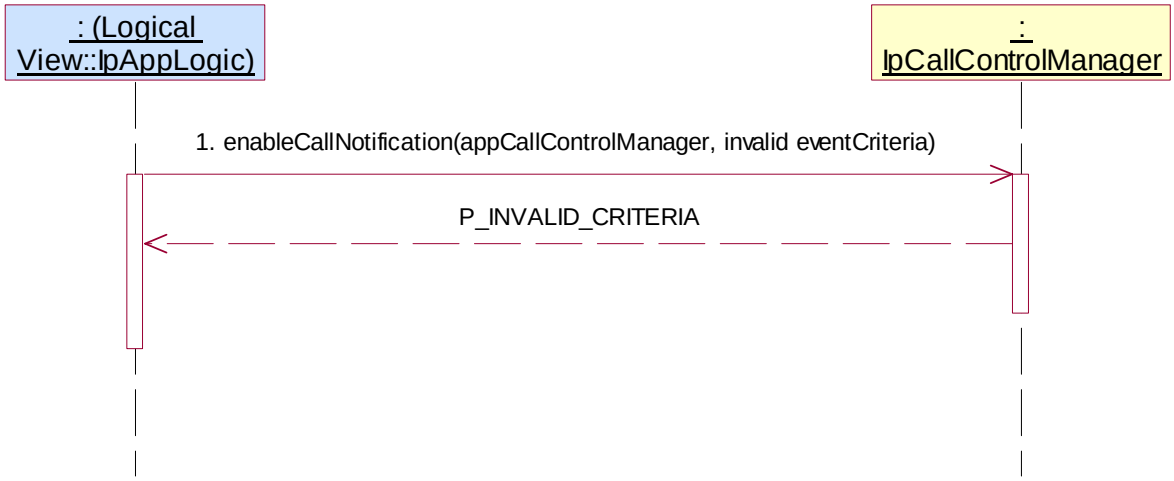
Summary: IpCallControlManager, enableCallNotification, P_INVALID_CRITERIA

Reference: ES 202 915-4-2 [2], clause 6.1

Condition: enableCallNotification method is supported.

Test Sequence:

1. Method call **enableCallNotification()**
   Parameters: valid appCallControlManager, invalid eventCriteria but with a valid event type
   Check: P_INVALID_CRITERIA is returned.



Test GCC_IPCALLCONTROLMANAGER _05

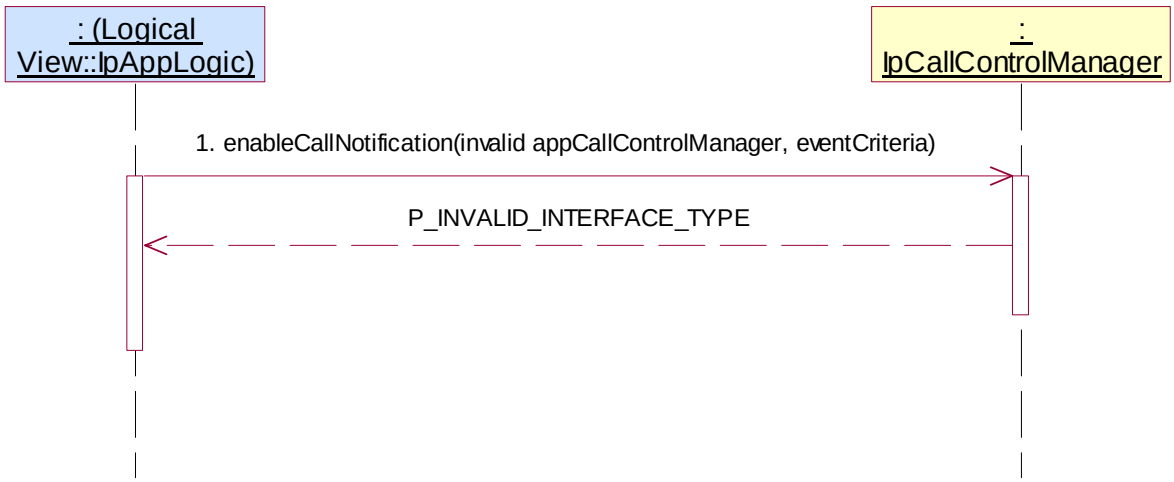
Summary: IpCallControlManager, enableCallNotification, P_INVALID_INTERFACE_TYPE

Reference: ES 202 915-4-2 [2], clause 6.1

Condition: enableCallNotification method is supported.

Test Sequence:

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



Test GCC_IPCALLCONTROLMANAGER _06

Summary: IpCallControlManager, disableCallNotification, P_INVALID_ASSIGNMENT_ID

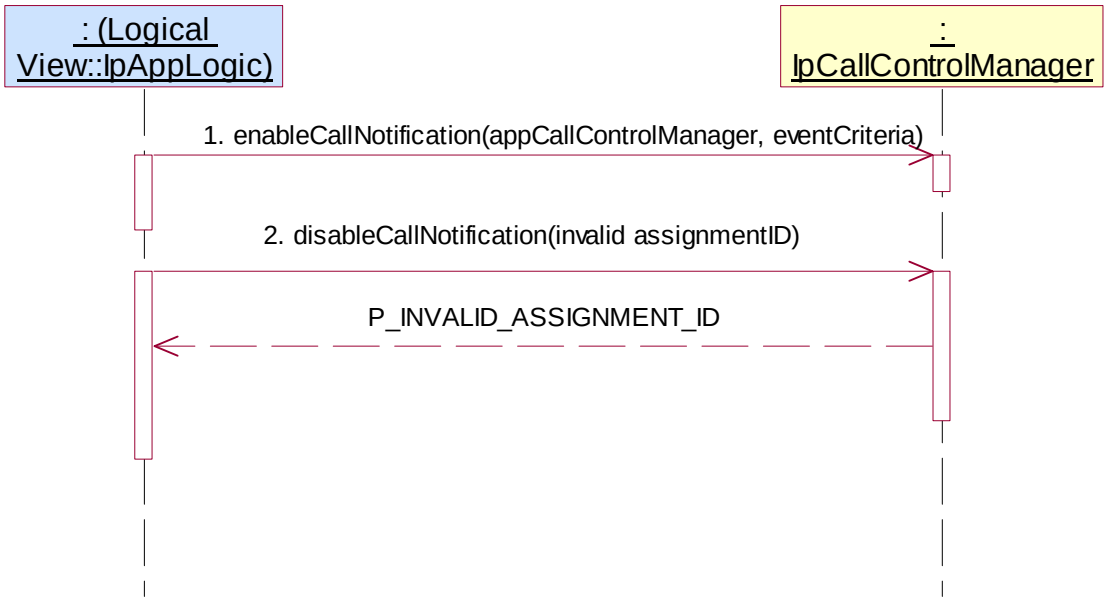
Reference: ES 202 915-4-2 [2], clause 6.1

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 is returned



Test GCC_IPCALLCONTROLMANAGER _07

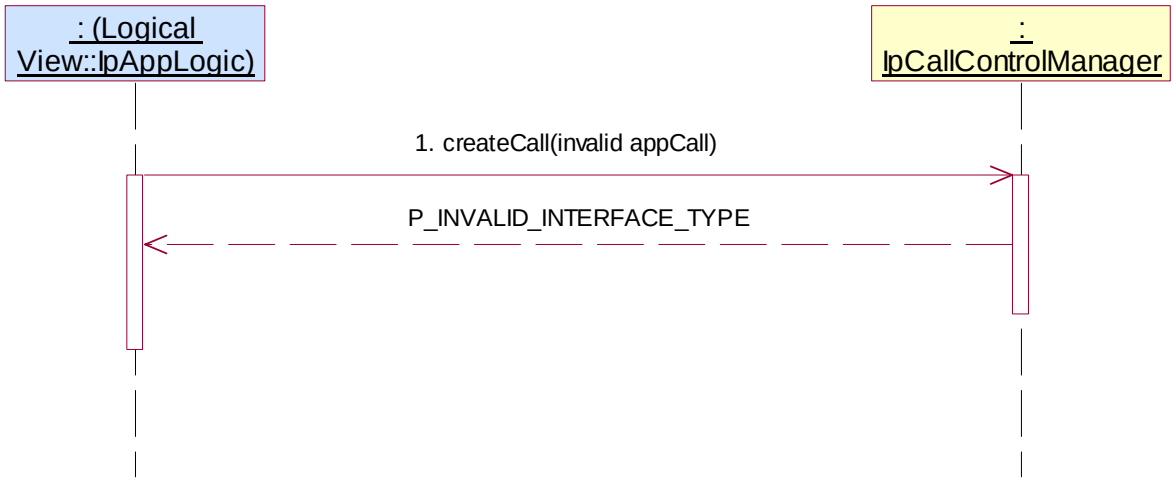
Summary: IpCallControlManager, createCall, P_INVALID_INTERFACE_TYPE

Reference: ES 202 915-4-2 [2], clause 6.1

Condition: createCall method is supported.

Test Sequence:

1. Method call **createCall()**
   Parameters: invalid value of appCall
   Check: P_INVALID_INTERFACE_TYPE is returned



5.2.1.1.3 Optional, valid behaviour

Test GCC_IPCALLCONTROLMANAGER _08

Summary: IpCallControlManager, getCriteria, successful

Reference: ES 202 915-4-2 [2], clause 6.1

Condition: enableCallNotification and getCriteria methods are supported.

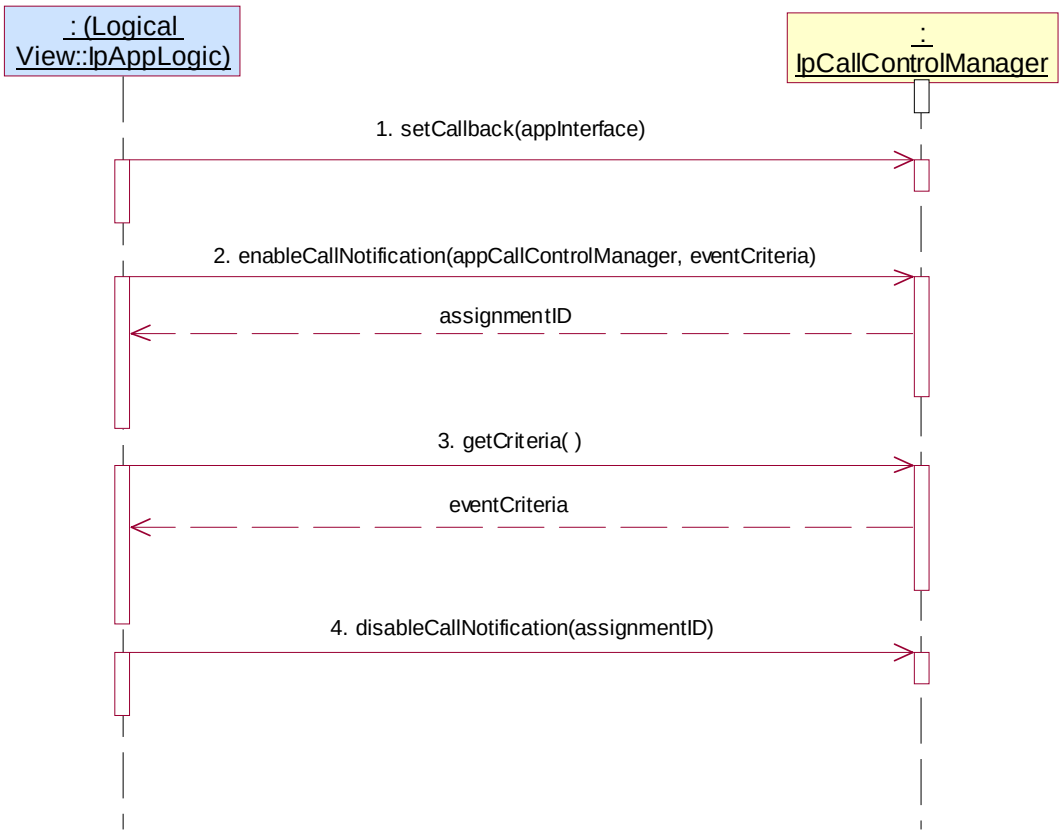
Test Sequence:

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 202 915-4-2 [2] clause 6.1

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

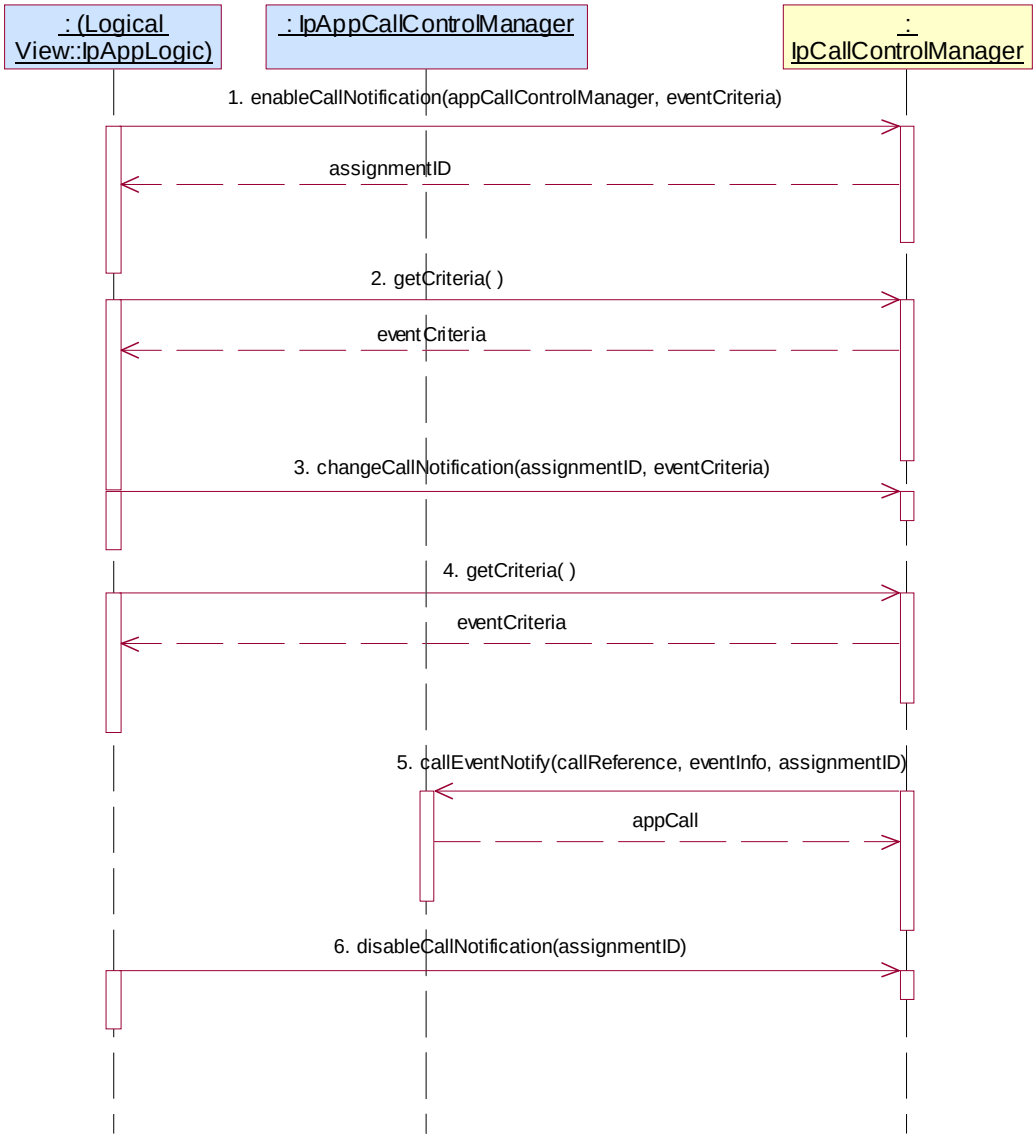
2. Method call **getCriteria()**
   Parameters: None
   Check: valid value of TpCallEventCriteriaResultSet is returned where eventCriteria given in 1. is included as a value of this TpCallEventCriteriaResultSet

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



Test GCC_IPCALLCONTROLMANAGER _10

Summary: IpCallControlManager, all methods, successful

Reference: ES 202 915-4-2 [2] clause 6.1

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

Test Sequence:

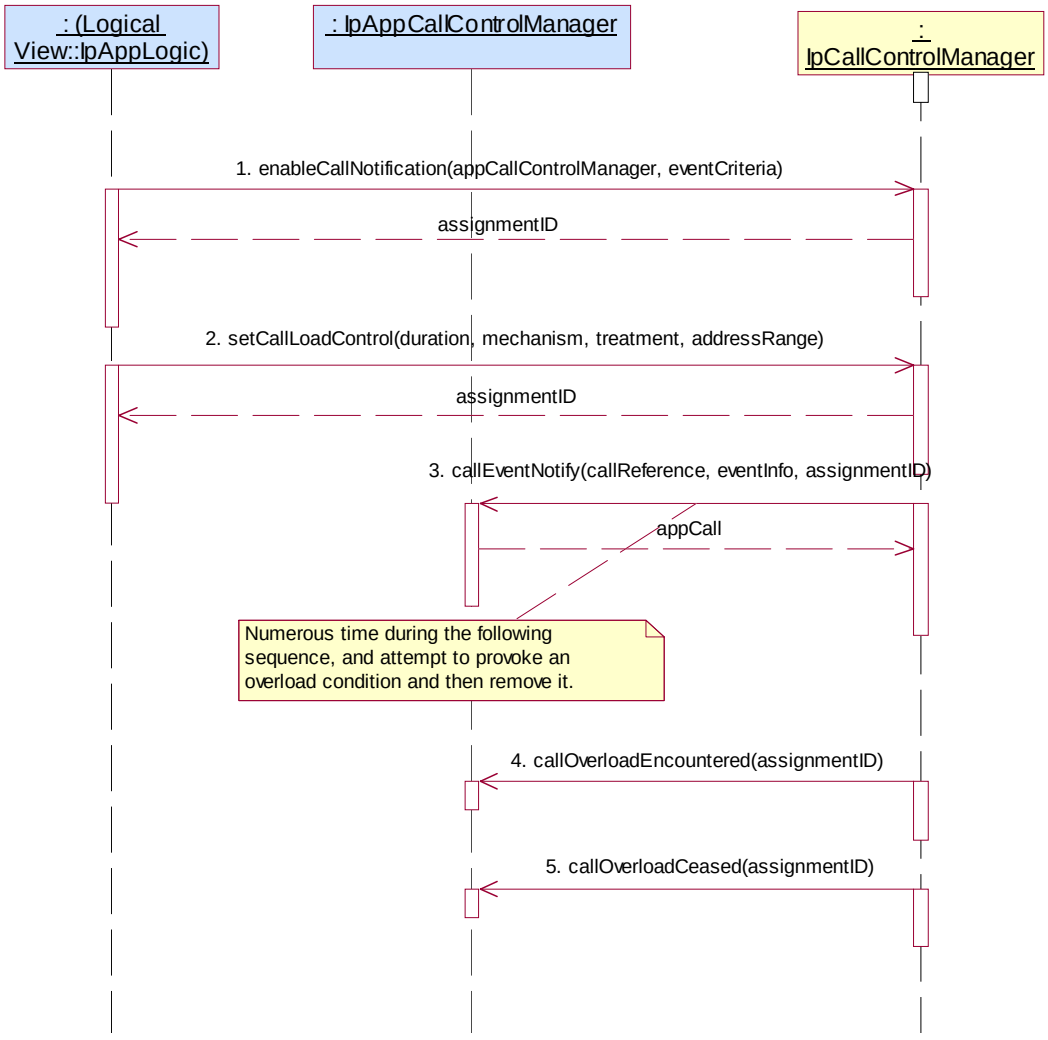
1. Method call **enableCallNotification()**
   Parameters: appCallControlManager with valid, not null, value, valid eventCriteria
   Check: valid value of TpAssignmentID is returned

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 202 915-4-2 [2] clause 6.1

Condition: enableCallNotification method is supported.

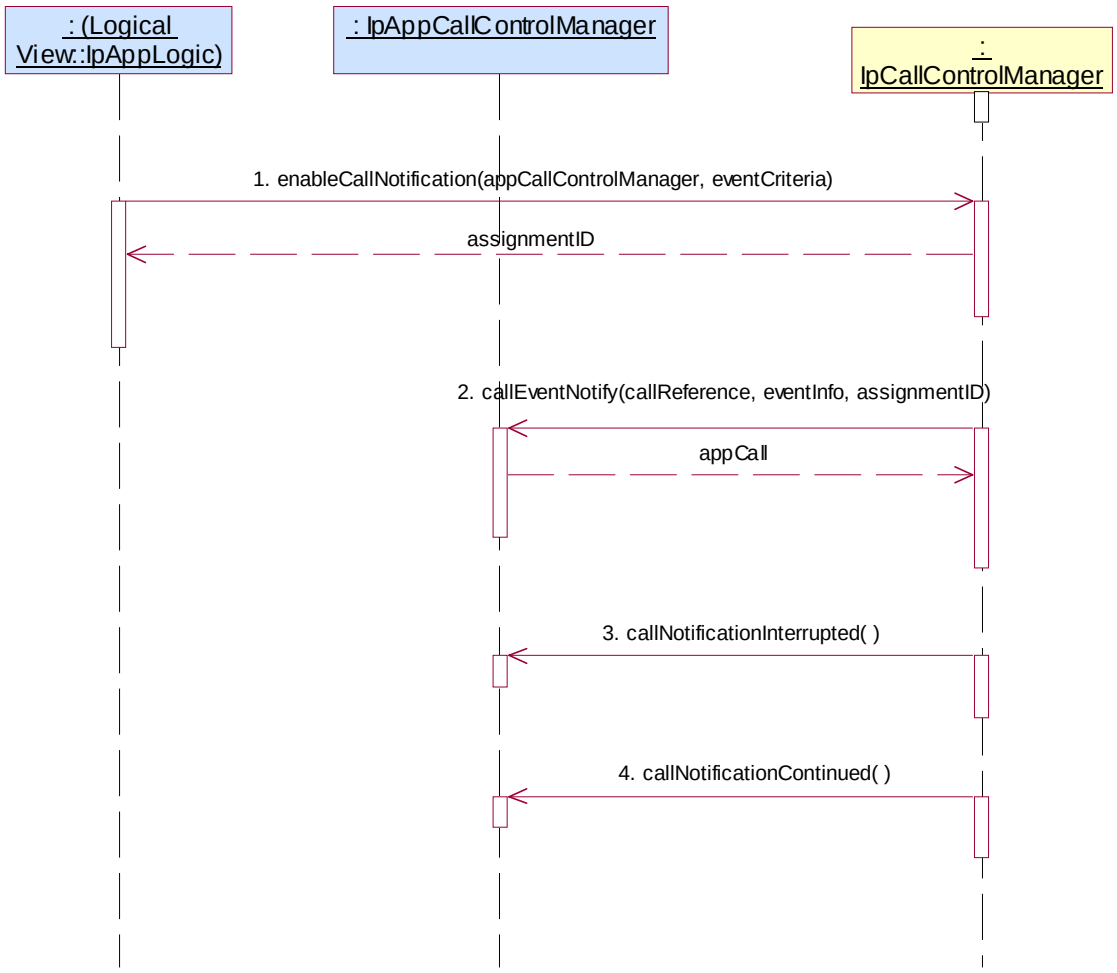
Test Sequence:

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 **callNotificationContinued()** method on the tester's (Application) **IpAppCallControlManager** interface.
   Parameters: None



Test GCC_IPCALLCONTROLMANAGER _12

Summary: IpCallControlManager, all methods, successful

Reference: ES 202 915-4-2 [2] clause 6.1

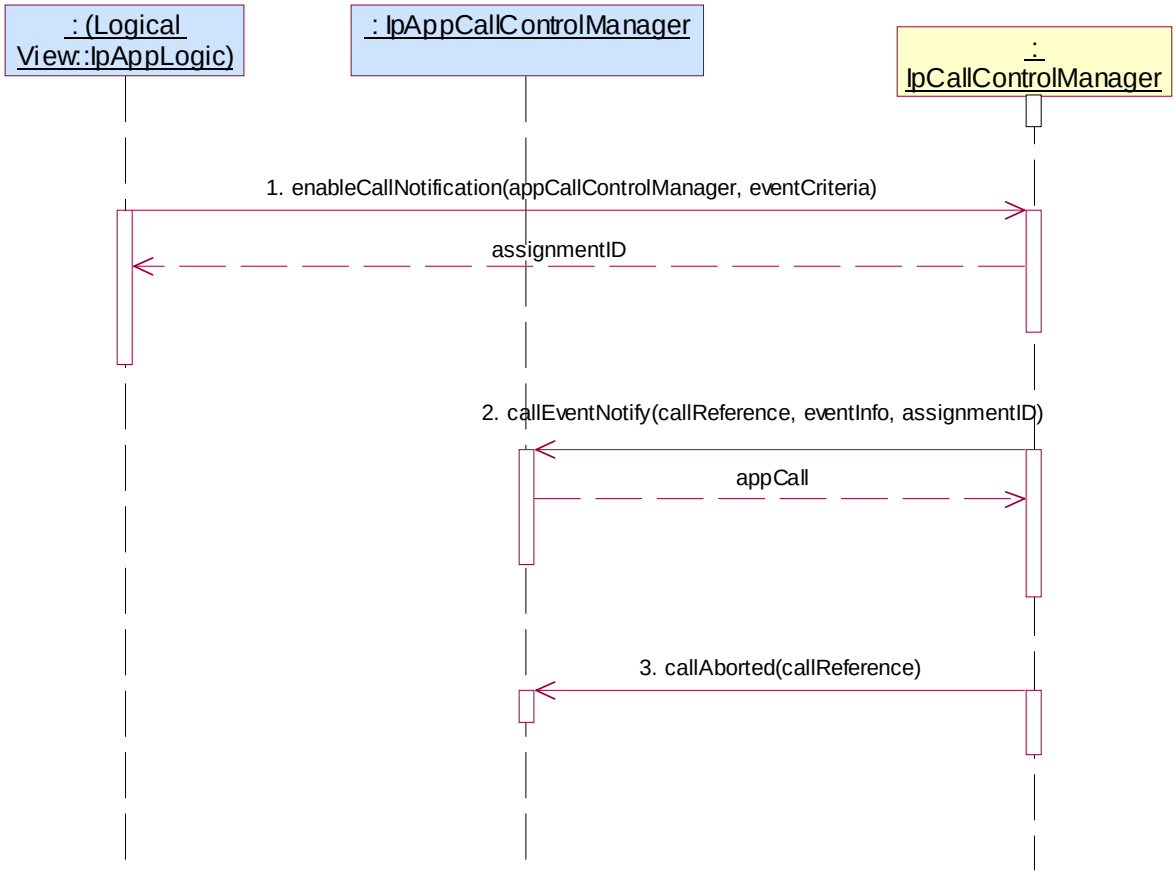
Condition: enableCallNotification and callAborted methods are supported.

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 callReference as reported in callEventNotify.



5.2.1.1.4 Optional, invalid behaviour

Test GCC_IPCALLCONTROLMANAGER _13

Summary: IpCallControlManager, changeCallNotification, P_INVALID_ASSIGNMENT_ID

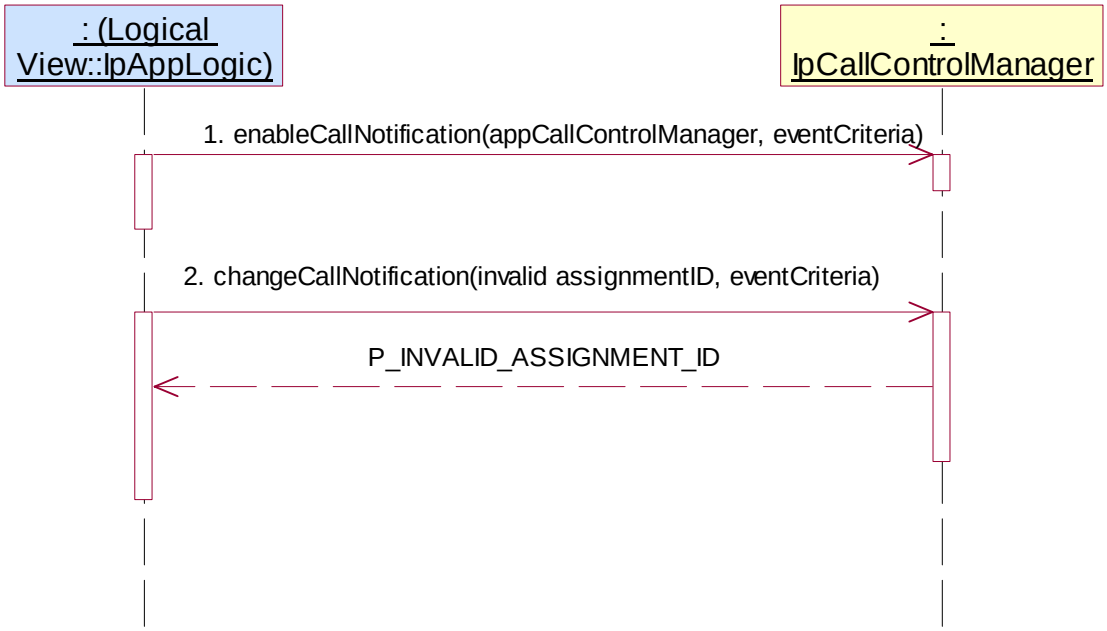
Reference: ES 202 915-4-2 [2] clause 6.1

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 is returned



Test GCC_IPCALLCONTROLMANAGER _14

Summary: IpCallControlManager, changeCallNotification, P_INVALID_CRITERIA

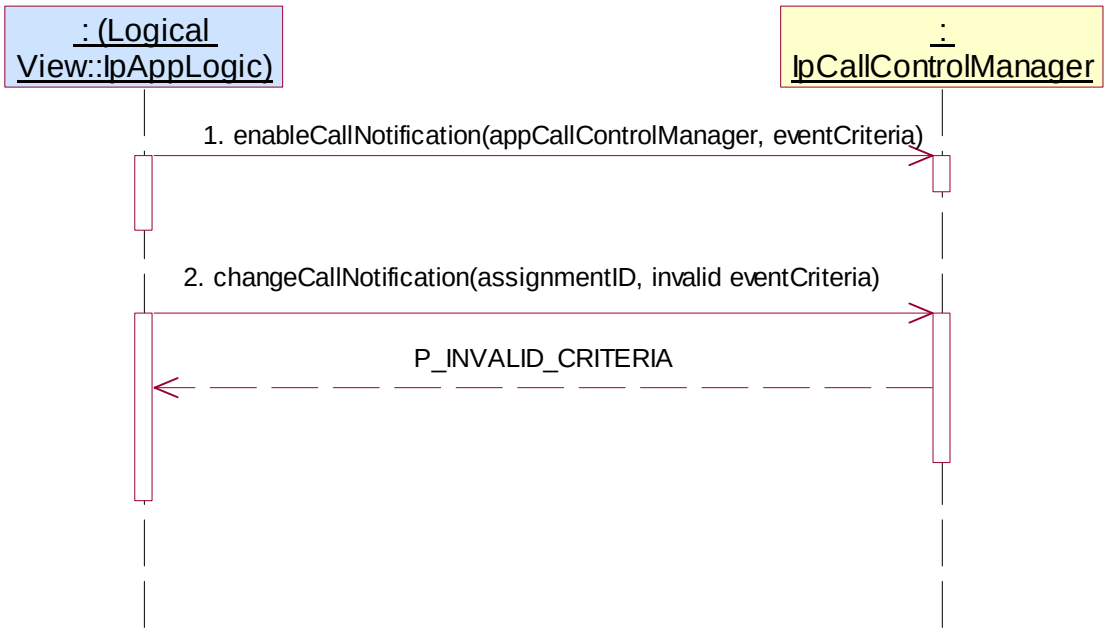
Reference: ES 202 915-4-2 [2] clause 6.1

Condition: enableCallNotification and changeCallNotification methods are supported.

Test Sequence:

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

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



Test GCC_IPCALLCONTROLMANAGER _15

Summary: IpCallControlManager, setCallLoadControl, P_INVALID_ADDRESS

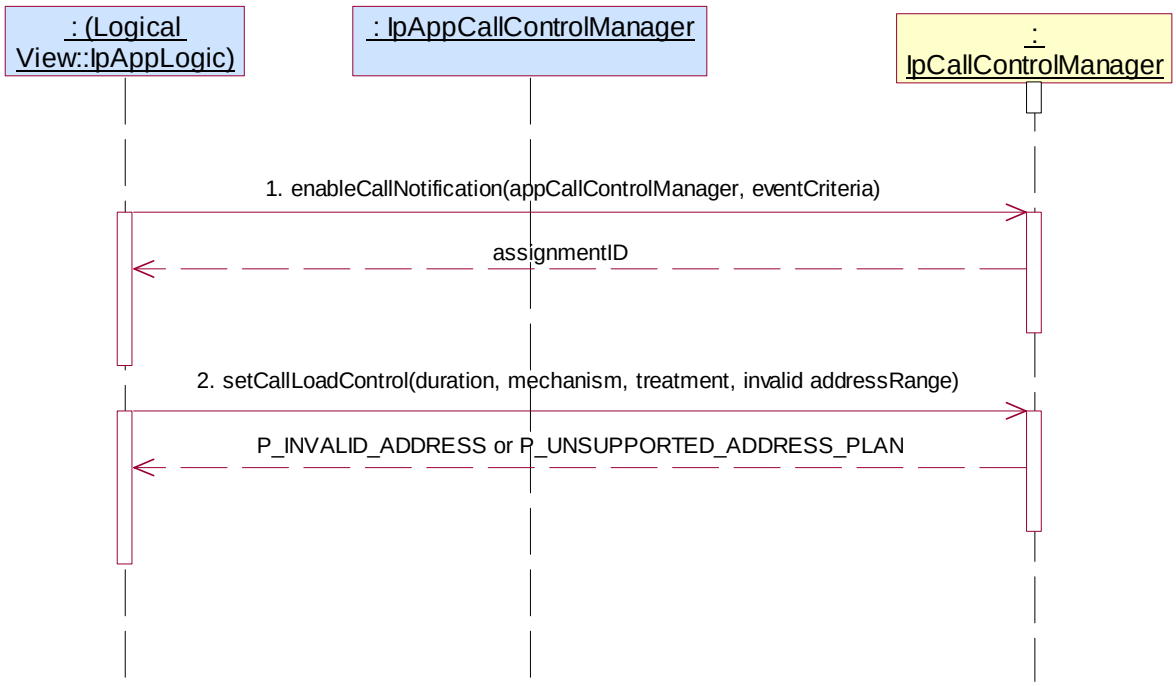
Reference: ES 202 915-4-2 [2] clause 6.1

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 or P_UNSUPPORTED_ADDRESS_PLAN 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 202 915-4-2 [2], clauses 6.1 and 6.3

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

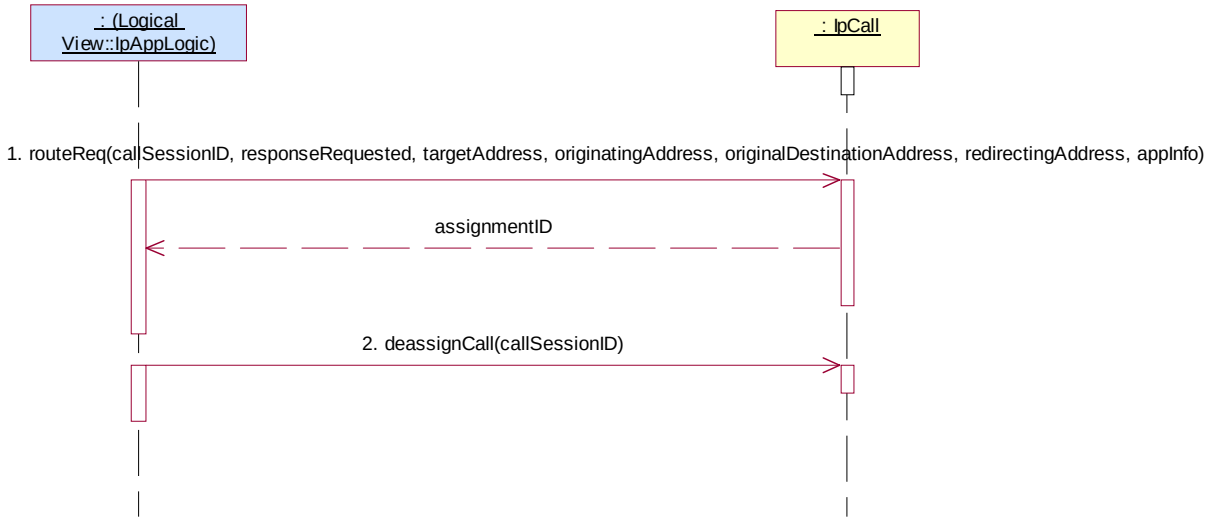
1. Method call **enableCallNotification()**
   Parameters: appCallControlManager with valid, not null, value, valid eventCriteria
   Check: valid value of TpAssignmentID is returned

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

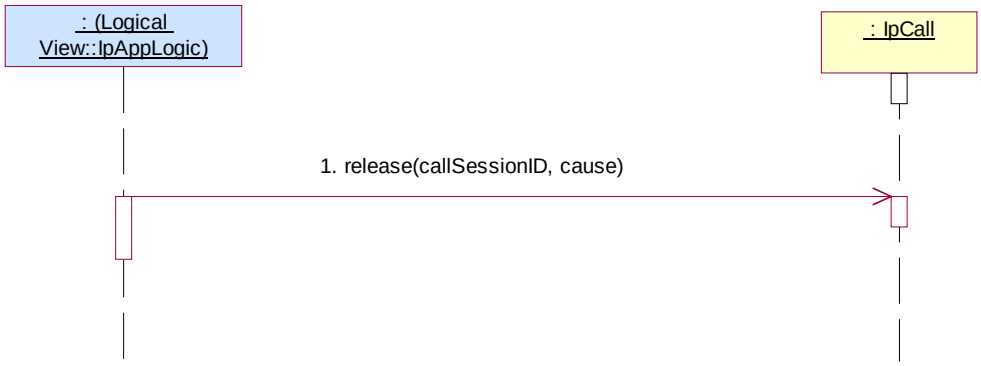
Summary: IpCall, all mandatory methods, successful

Reference: ES 202 915-4-2 [2], clauses 6.1 and 6.3

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

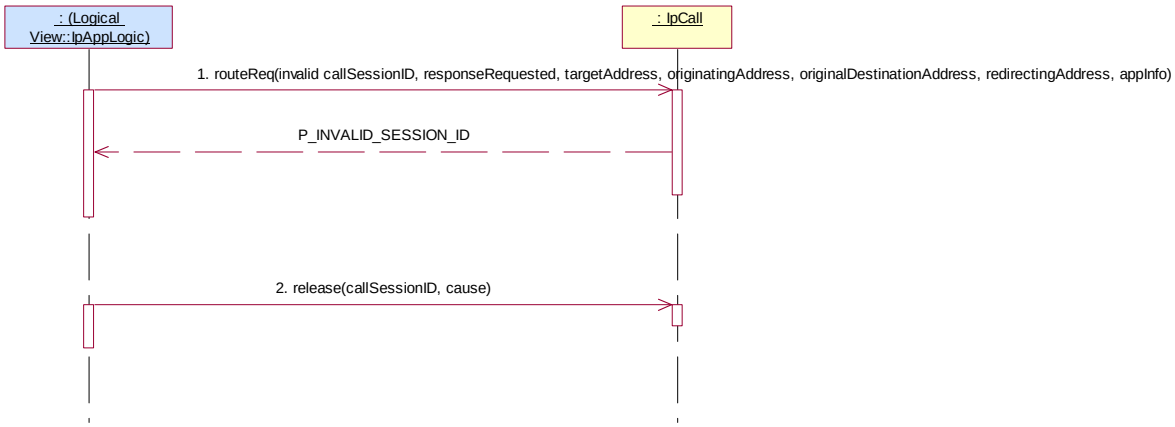
Summary: IpCall, routeReq, P_INVALID_SESSION_ID

Reference: ES 202 915-4-2 [2], clause 6.3

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 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

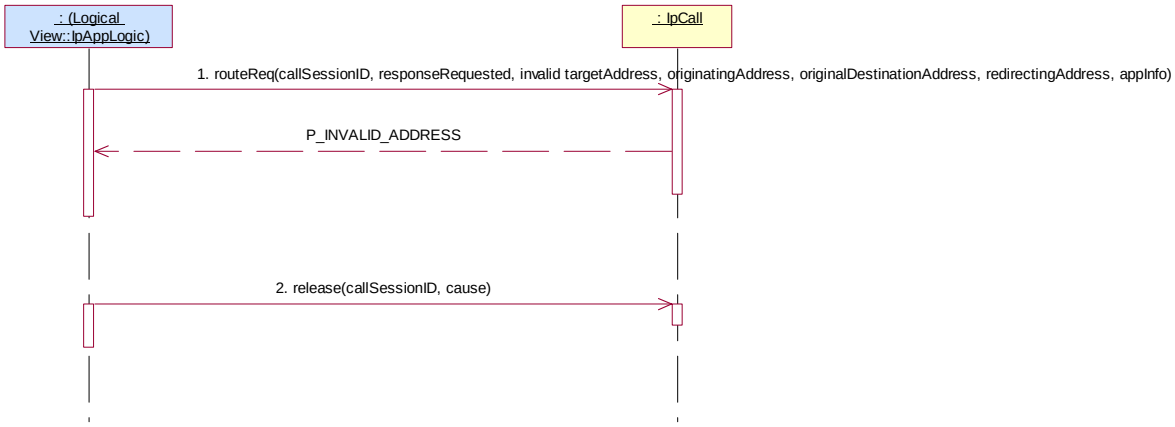
Reference: ES 202 915-4-2 [2], clauses 6.1 and 6.3

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 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

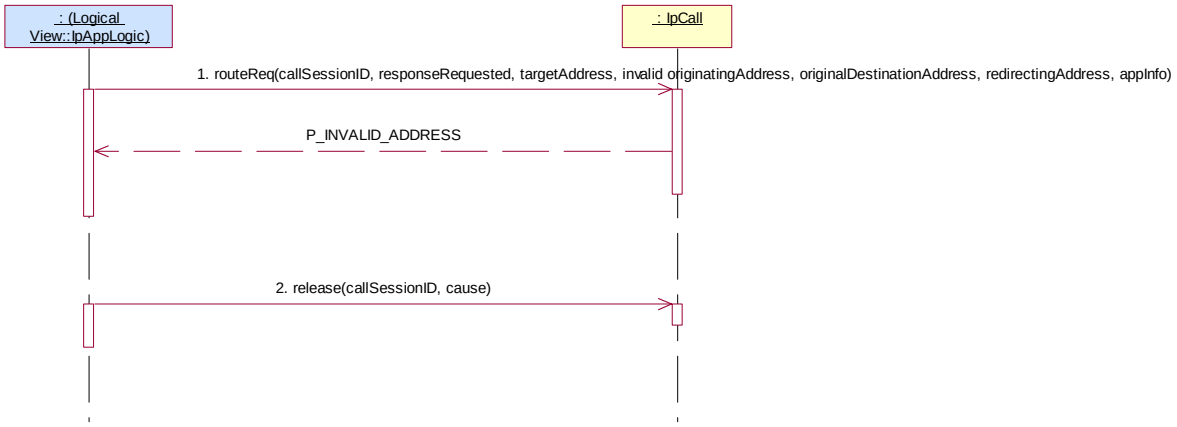
Reference: ES 202 915-4-2 [2], clauses 6.1 and 6.3

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 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

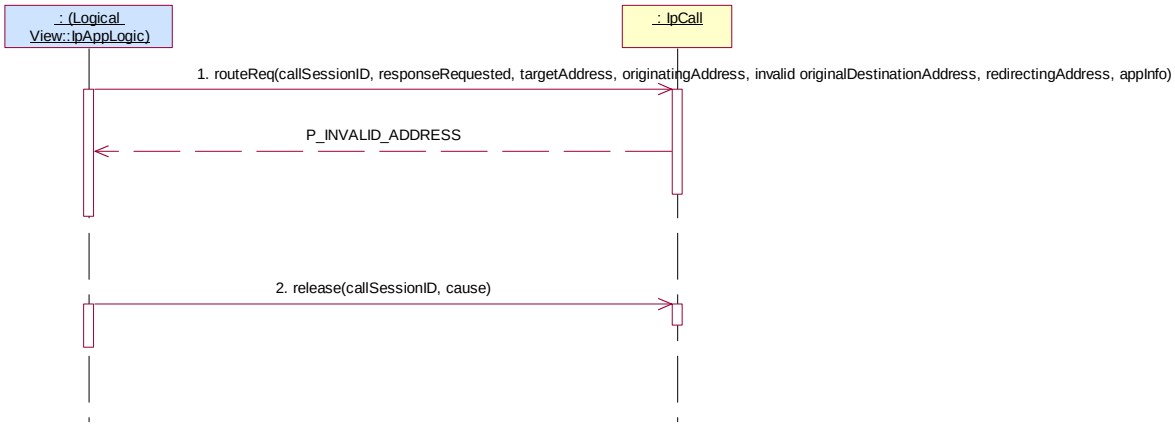
Reference: ES 202 915-4-2 [2], clauses 6.1 and 6.3

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 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

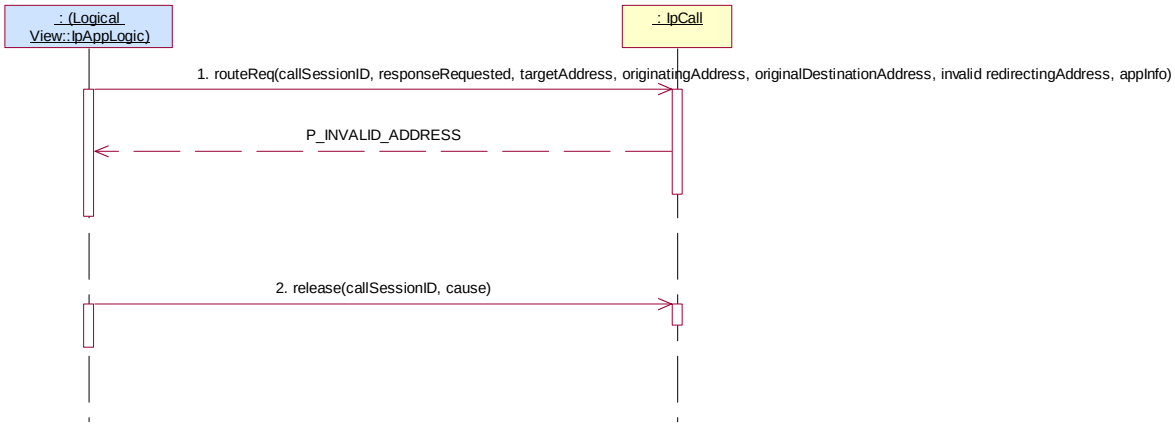
Reference: ES 202 915-4-2 [2], clauses 6.1 and 6.3

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 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

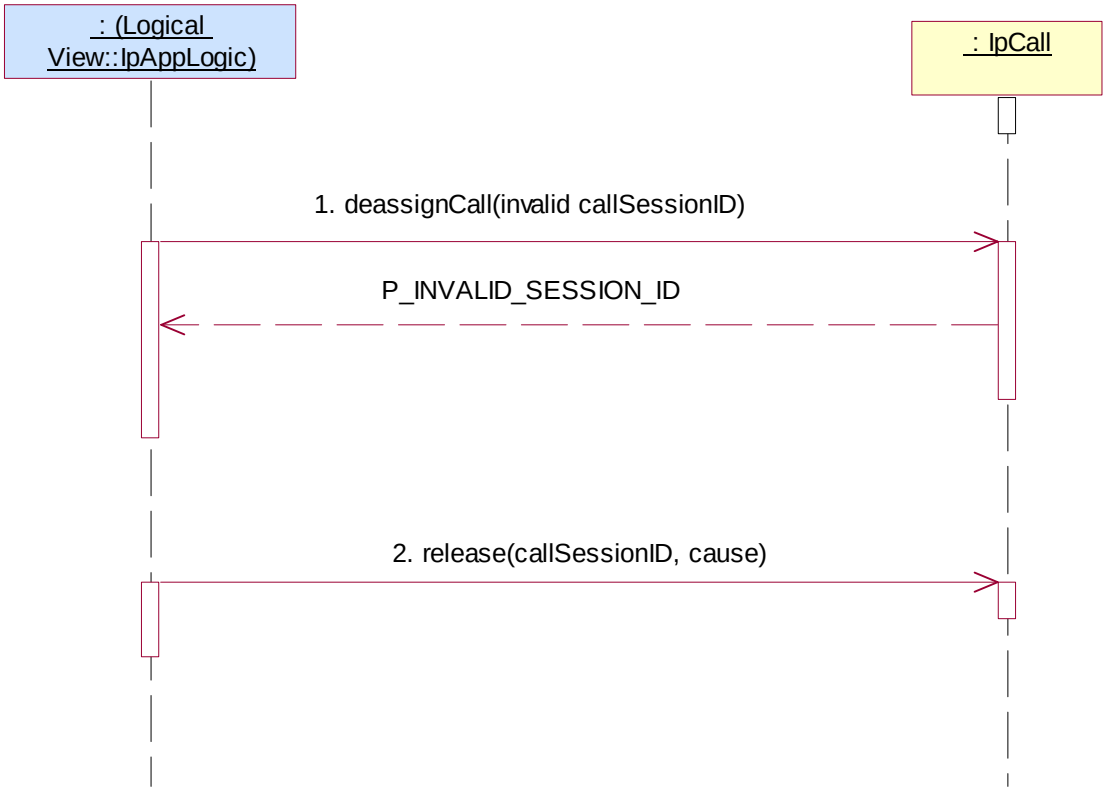
Reference: ES 202 915-4-2 [2], clause 6.3

Preamble: Same as GCC_IPCALL_01

Test Sequence:

1. Method call **deassignCall()**
   Parameters: invalid callSessionID
   Check: P_INVALID_SESSION_ID is returned

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


: (Logical View::IpAppLogic)
: IpCall
1. deassignCall(invalid callSessionID)
P_INVALID_SESSION_ID
2. release(callSessionID, cause)

Test GCC_IPCALL _09

Summary: IpCall, release, P_INVALID_SESSION_ID

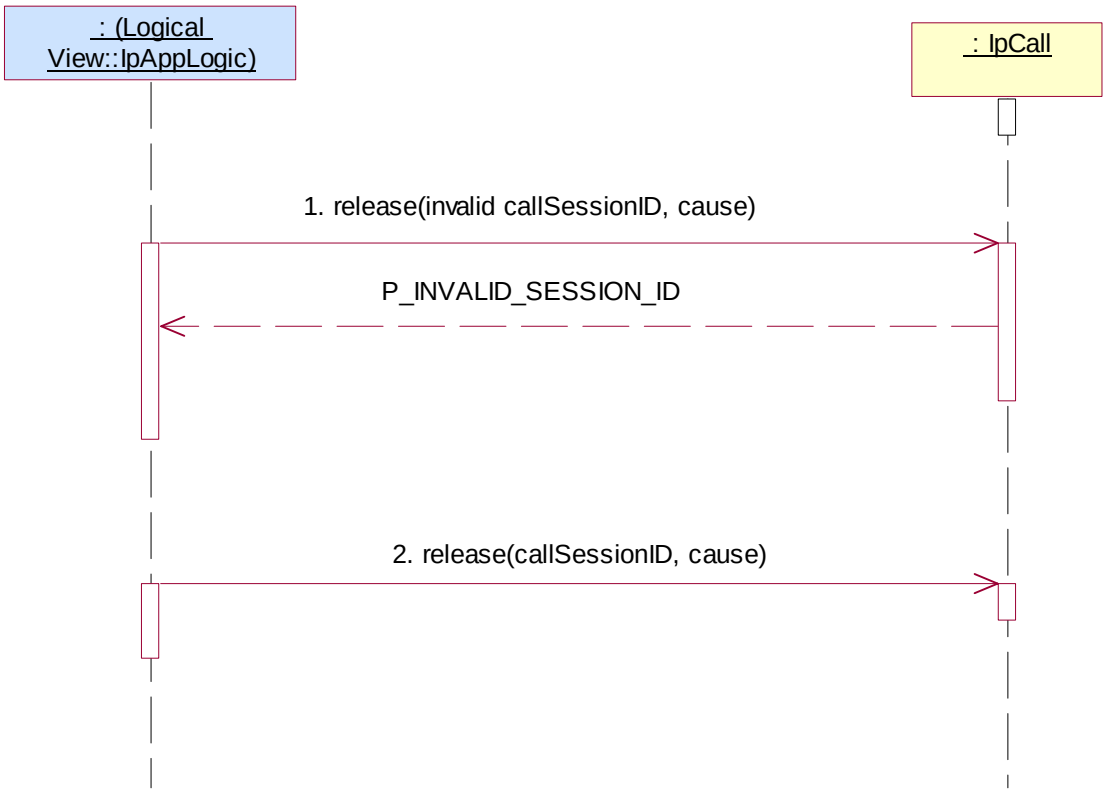
Reference: ES 202 915-4-2 [2], clause 6.3

Preamble: Same as GCC_IPCALL_01

Test Sequence:

1. Method call **release()**
   Parameters: invalid callSessionID
   Check: P_INVALID_SESSION_ID

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 202 915-4-2 [2], clauses 6.1 and 6.3

Preamble: Same as GCC_IPCALL_01

Condition: getCallInfoReq method is supported.

Test Sequence:

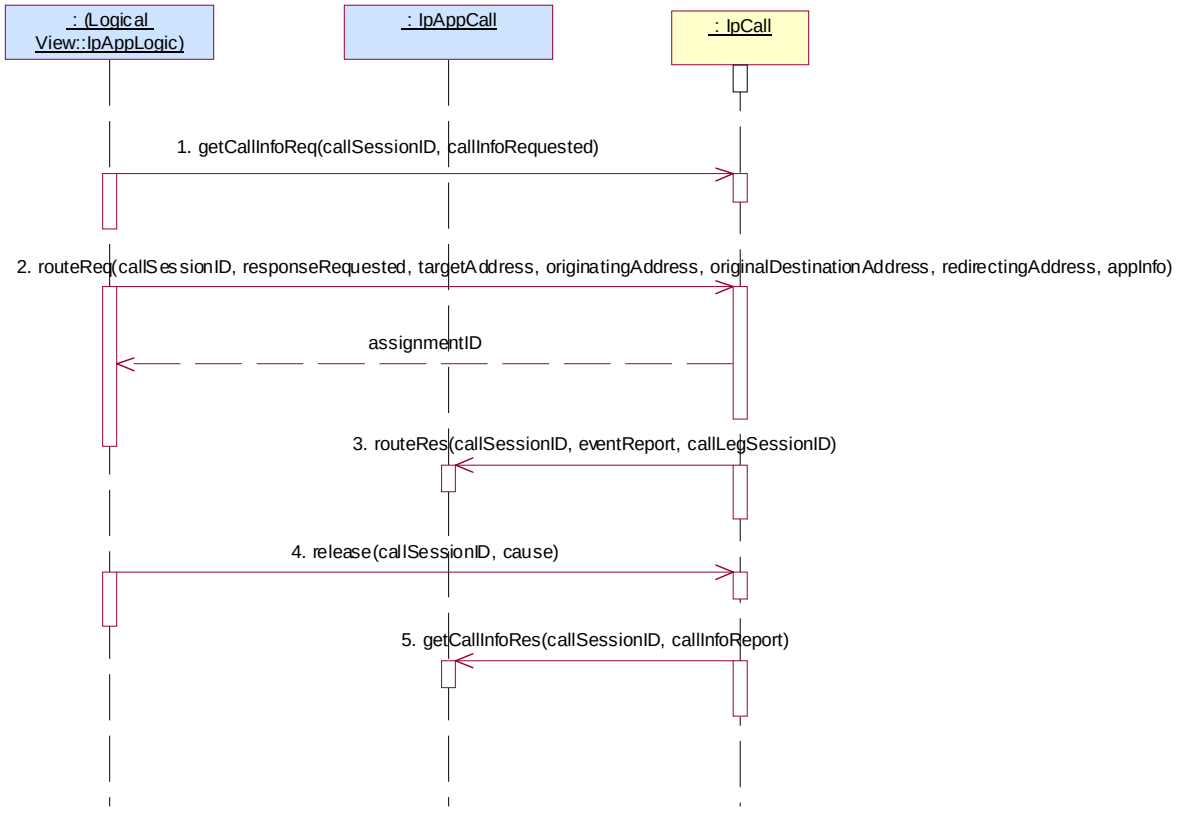
1. Method call **getCallInfoReq()**
   Parameters: valid callSessionID reported in preamble, valid callInforequested
   Check: no exception is returned

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 202 915-4-2 [2], clauses 6.1 and 6.3

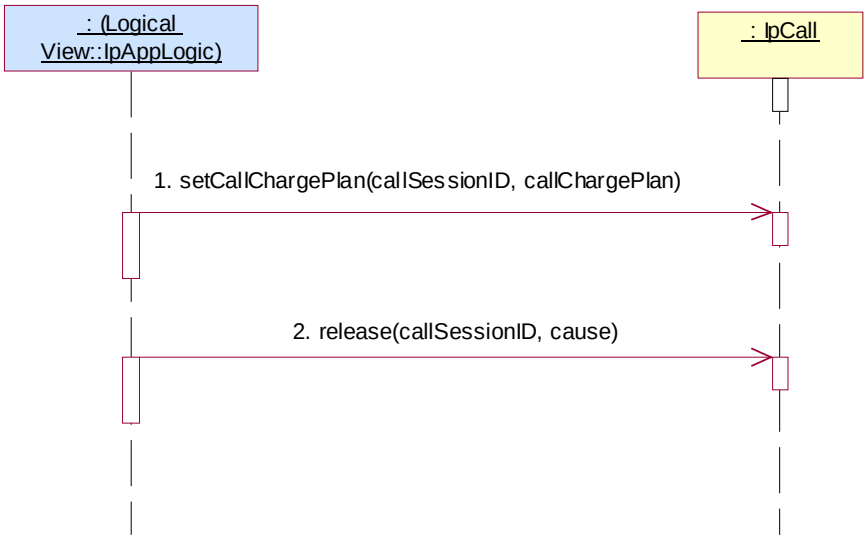
Preamble: Same as GCC_IPCALL_01

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 202 915-4-2 [2], clauses 6.1 and 6.3

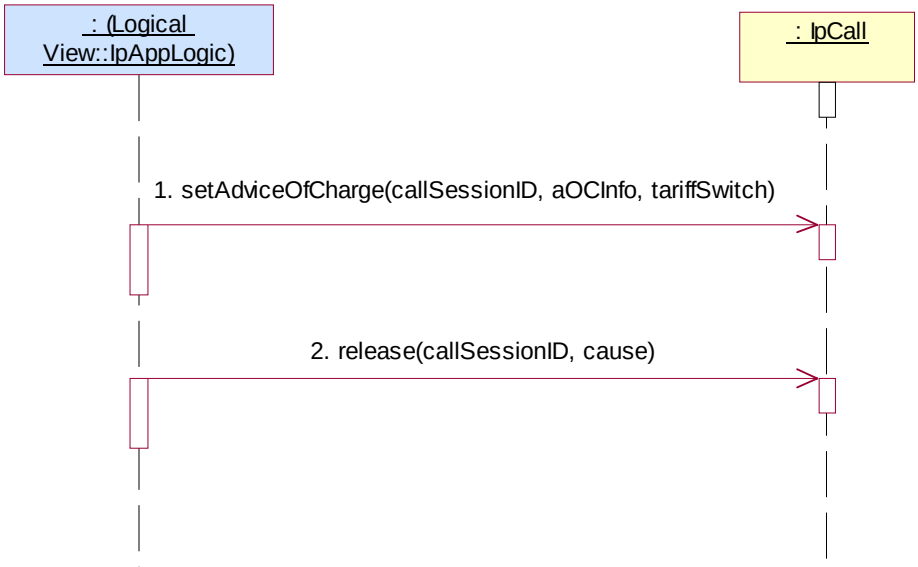
Preamble: Same as GCC_IPCALL_01

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 202 915-4-2 [2], clauses 6.1 and 6.3

Condition: getMoreDialledDigitsReq method is supported.

Test Sequence:

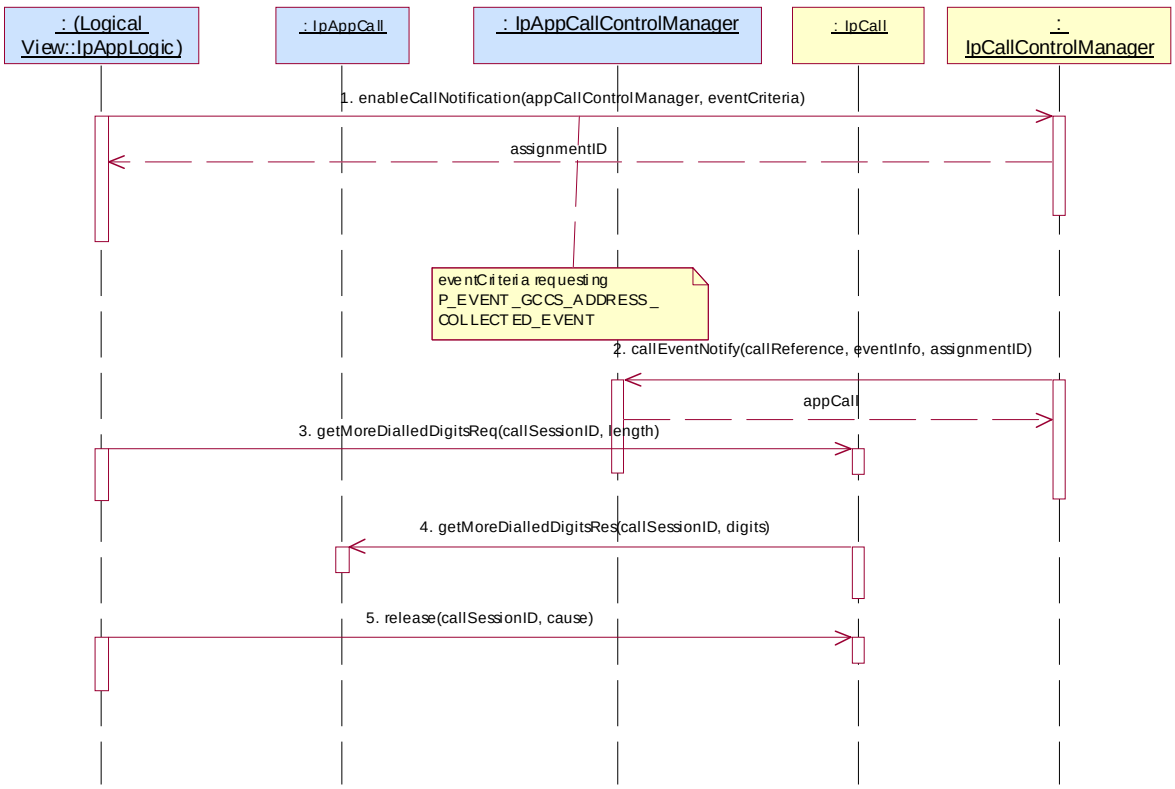
1. Method call **enableCallNotification()**
   Parameters: appCallControlManager with valid, not null, value, valid eventCriteria requesting P_EVENT_GCCS_ADDRESS_COLLECTED_EVENT.
   Check: valid value of TpAssignmentID is returned

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 in 2.
   Check: no exception is returned



Test GCC_IPCALL _14

Summary: IpCall, superviseCallReq, successful

Reference: ES 202 915-4-2 [2], clauses 6.1 and 6.3

Preamble: Same as GCC_IPCALL_01

Condition: superviseCallReq method is supported.

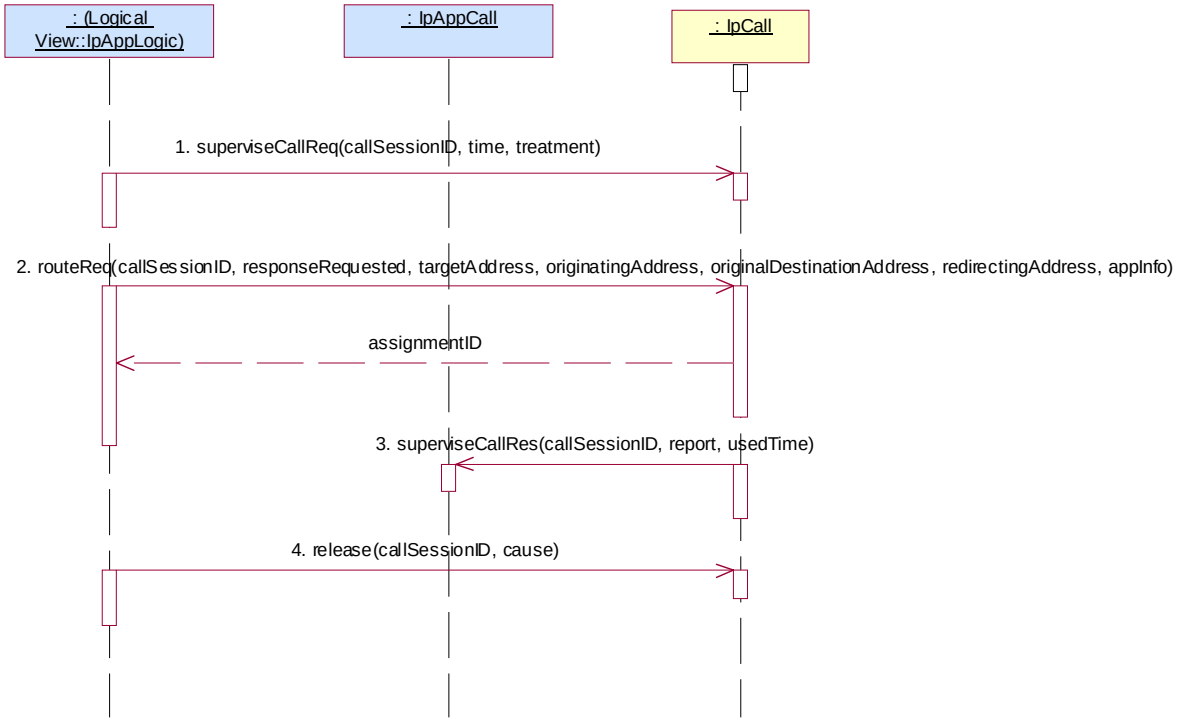
Test Sequence:

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



Test GCC_IPCALL _15

Summary: IpCall, continueProcessing, successful

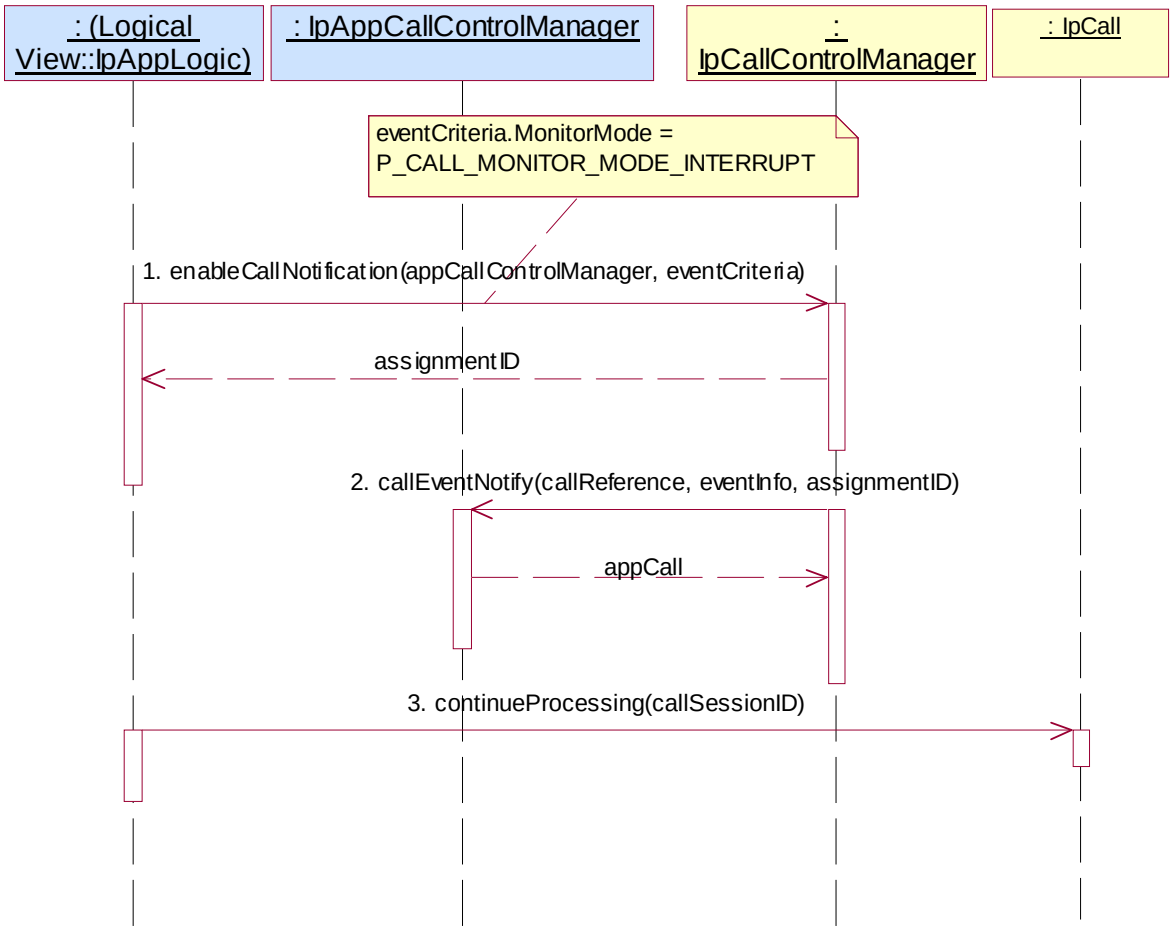
Reference: ES 202 915-4-2 [2], clauses 6.1 and 6.3

Test Sequence:

1. Method call **enableCallNotification()**
   Parameters: appCallControlManager with valid, not null, value, valid eventCriteria
   eventCriteria.MonitorMode = P_CALL_MONITOR_MODE_INTERRUPT
   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 **continueProcessing()**
   Parameters: callSessionID
   Check: no exception is returned



5.2.1.2.4 Optional, invalid behaviour

Test GCC_IPCALL _16

Summary: IpCall, getCallInfoReq, P_INVALID_SESSION_ID

Reference: ES 202 915-4-2 [2], clause 6.3

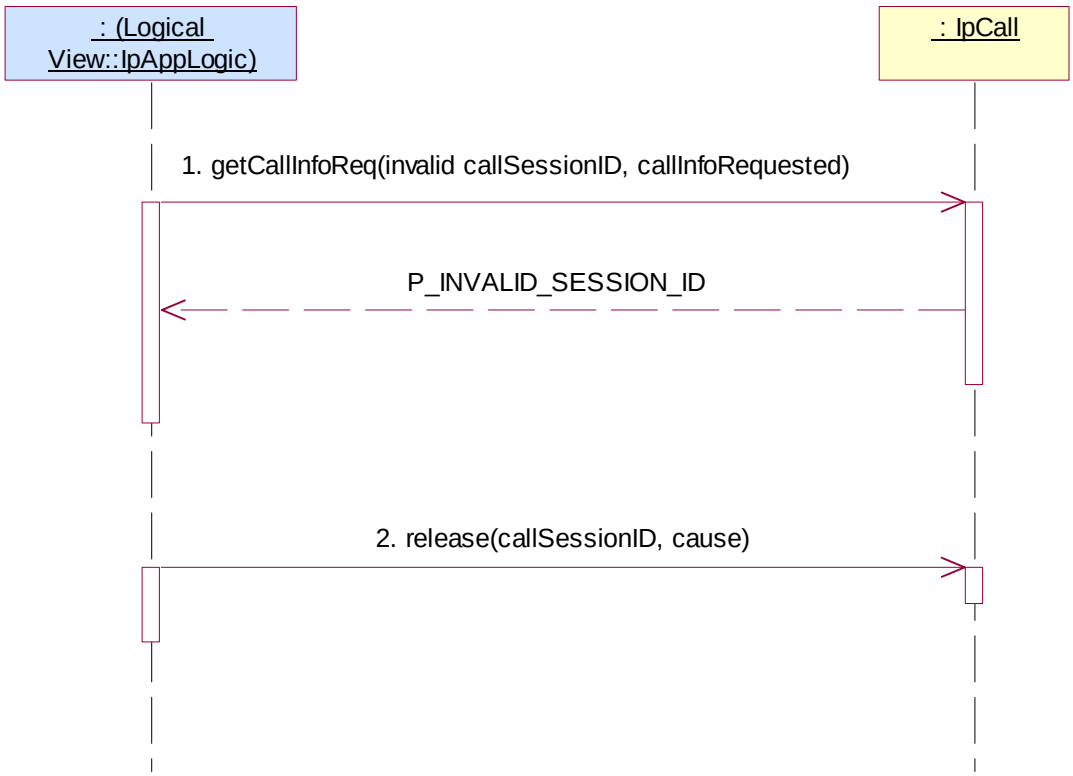
Preamble: Same as GCC_IPCALL_01

Condition: getCallInfoReq is supported.

Test Sequence:

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

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



Test GCC_IPCALL _17

Summary: IpCall, setCallChargePlan, P_INVALID_SESSION_ID

Reference: ES 202 915-4-2 [2], clause 6.3

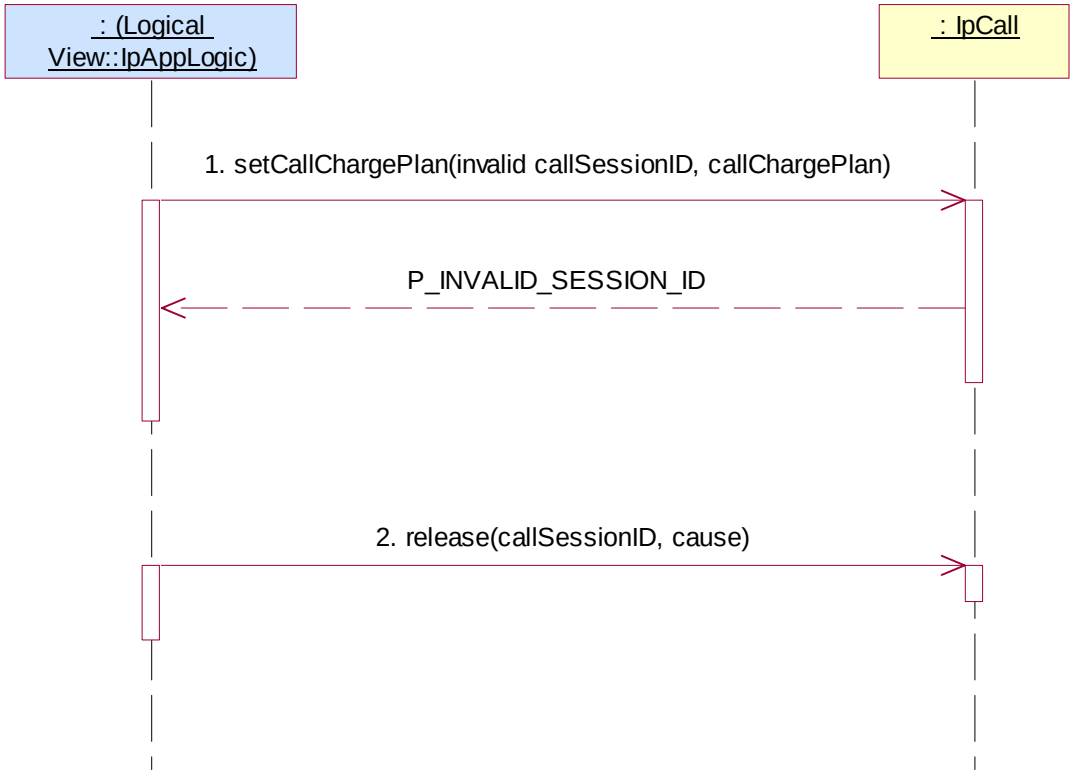
Preamble: Same as GCC_IPCALL_01

Condition: setCallChargePlan is supported.

Test Sequence:

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

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



Test GCC_IPCALL _18

Summary: IpCall, setAdviceOfCharge, P_INVALID_SESSION_ID

Reference: ES 202 915-4-2 [2], clause 6.3

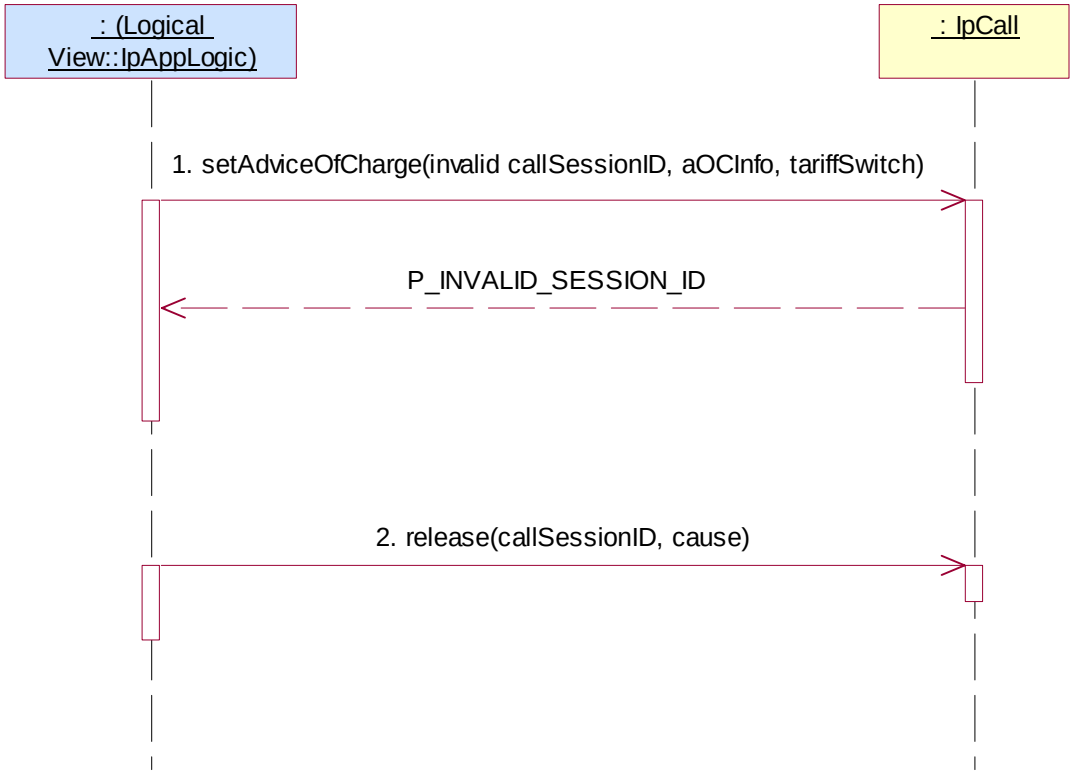
Preamble: Same as GCC_IPCALL_01

Condition: setAdviceOfCharge is supported.

Test Sequence:

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

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



Test GCC_IPCALL _19

Summary: IpCall, getMoreDialledDigitsReq, P_INVALID_SESSION_ID

Reference: ES 202 915-4-2 [2], clause 6.3

Condition: getMoreDialledDigitsReq is supported.

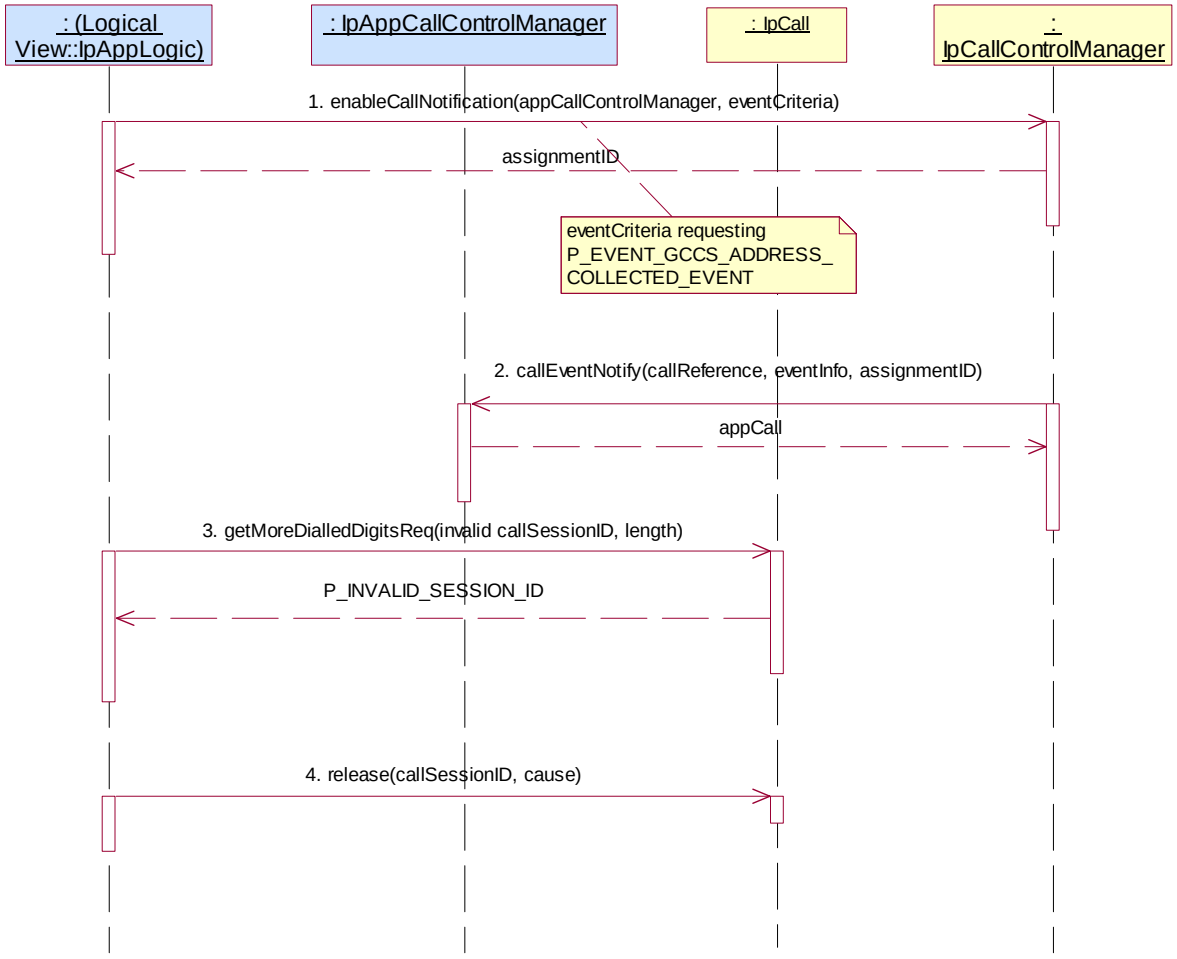
Test Sequence:

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 is returned

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



Test GCC_IPCALL _20

Summary: IpCall, superviseCallReq, P_INVALID_SESSION_ID

Reference: ES 202 915-4-2 [2], clause 6.3

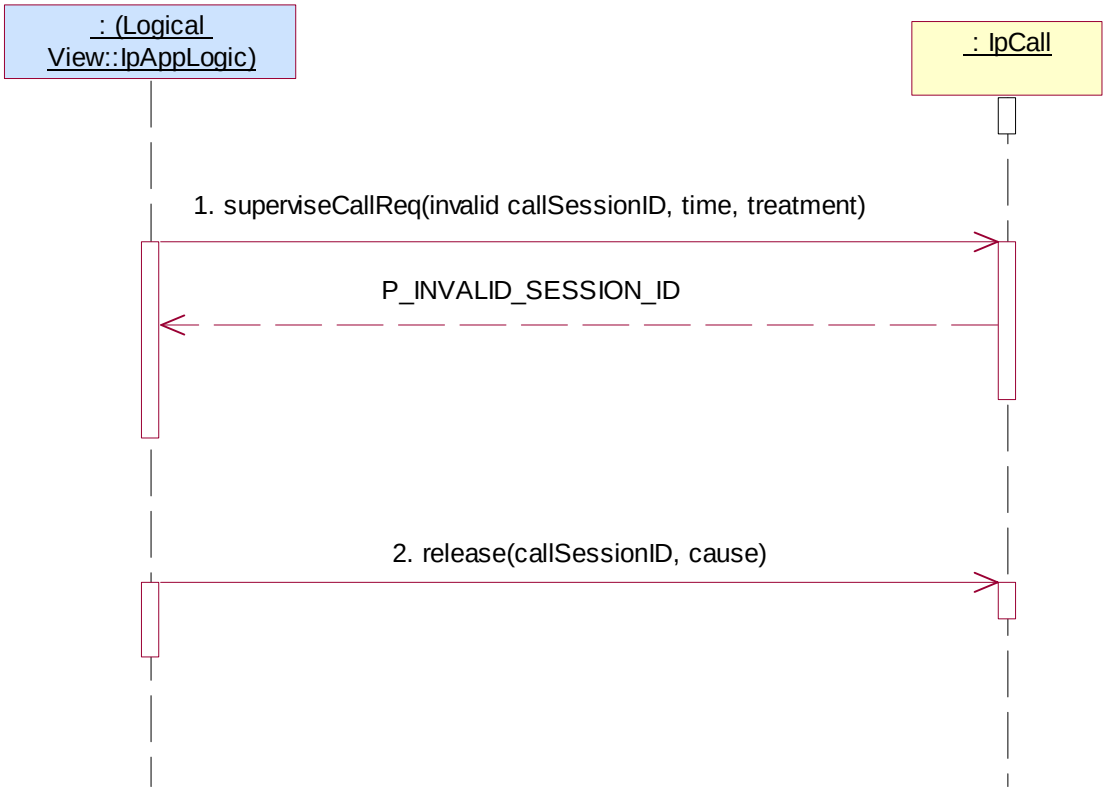
Preamble: Same as GCC_IPCALL_01

Condition: superviseCallReq is supported.

Test Sequence:

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

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



Test GCC_IPCALL _21

Summary: IpCall, continueProcessing, P_INVALID_SESSION_ID

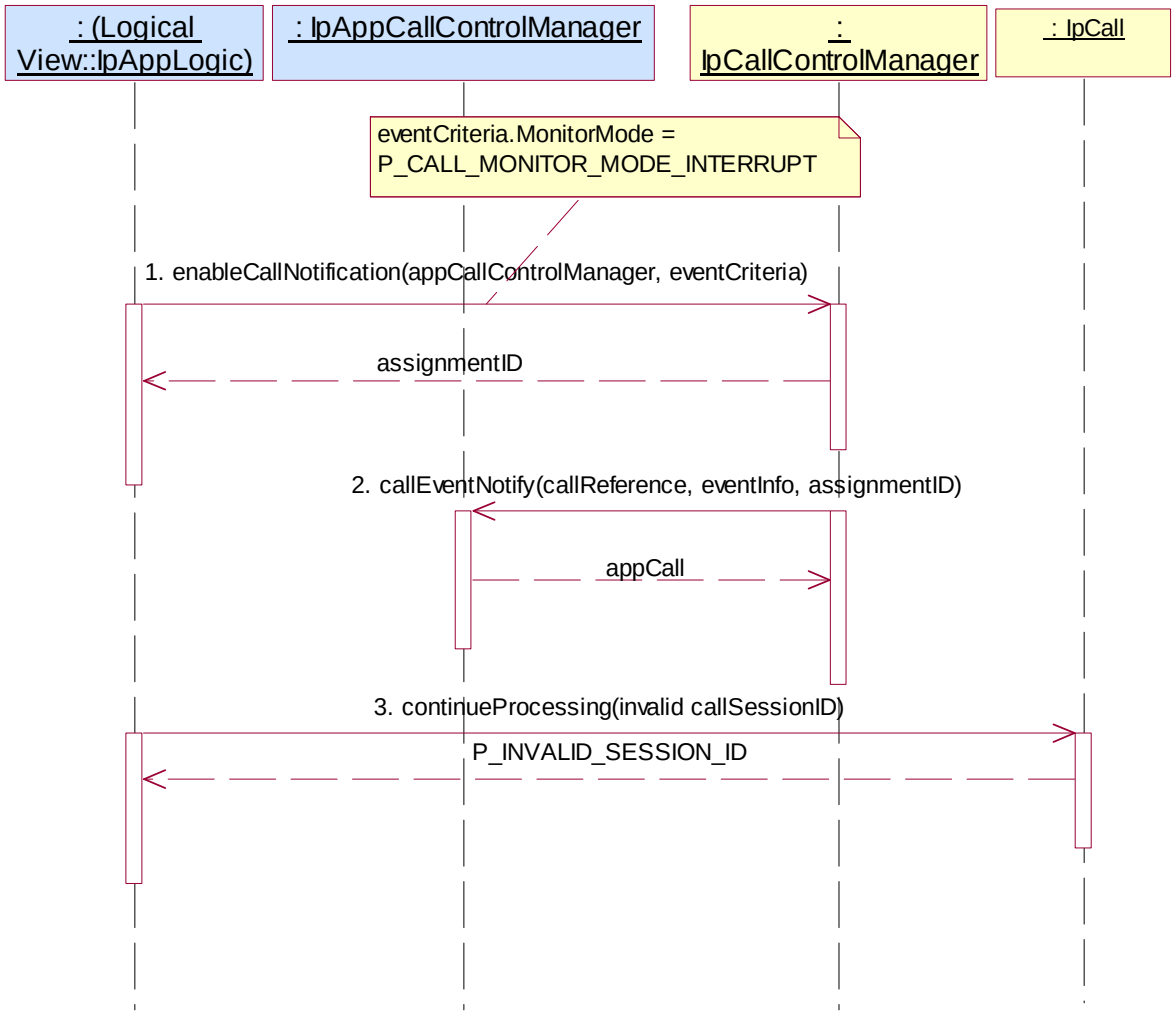
Reference: ES 202 915-4-2 [2], clauses 6.1 and 6.3

Test Sequence:

1. Method call **enableCallNotification()**
   Parameters: appCallControlManager with valid, not null, value, valid eventCriteria
   eventCriteria.MonitorMode = P_CALL_MONITOR_MODE_INTERRUPT
   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 **continueProcessing()**
   Parameters: invalid callSessionID
   Check: P_INVALID_SESSION_ID is returned



5.2.2 MultiParty Call Control Service (MPCC)

The TPs in this clause are based on ES 202 915-4-3 [3].

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 202 915-4-3 [3], clause 6.1

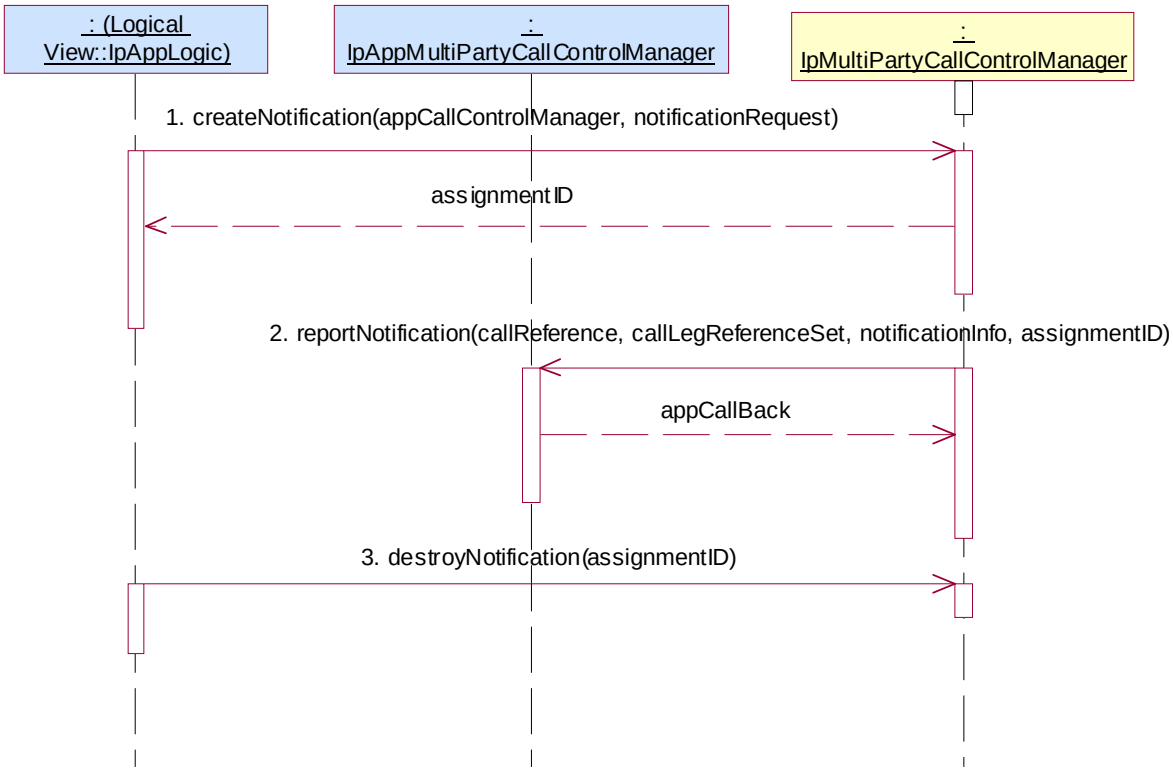
Condition: createNotification method is supported.

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 202 915-4-3 [3], clause 6.1

Condition: createNotification method is supported.

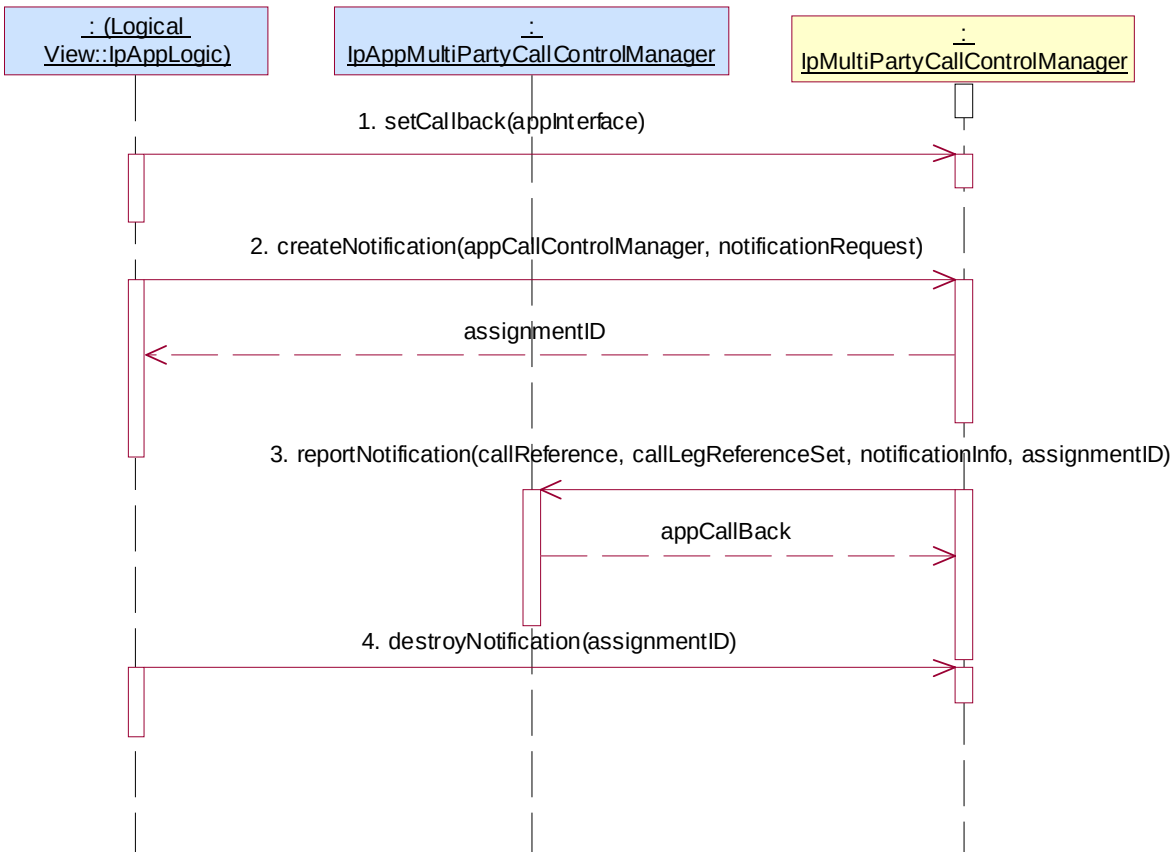
Test Sequence:

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

2. Method call **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


: (Logical View::IpAppLogic)
: IpAppMultiPartyCallControlManager
: IpMultiPartyCallControlManager
1. setCallback(appInterface)
2. createNotification(appCallControlManager, notificationRequest)
assignmentID
3. reportNotification(callReference, callLegReferenceSet, notificationInfo, assignmentID)
appCallBack
4. destroyNotification(assignmentID)

Test MPCC_ IpMultiPartyCallControlManager _03

Summary: IpMultiPartyCallControlManager, all mandatory methods, successful

Reference: ES 202 915-4-3 [3], clause 6.1

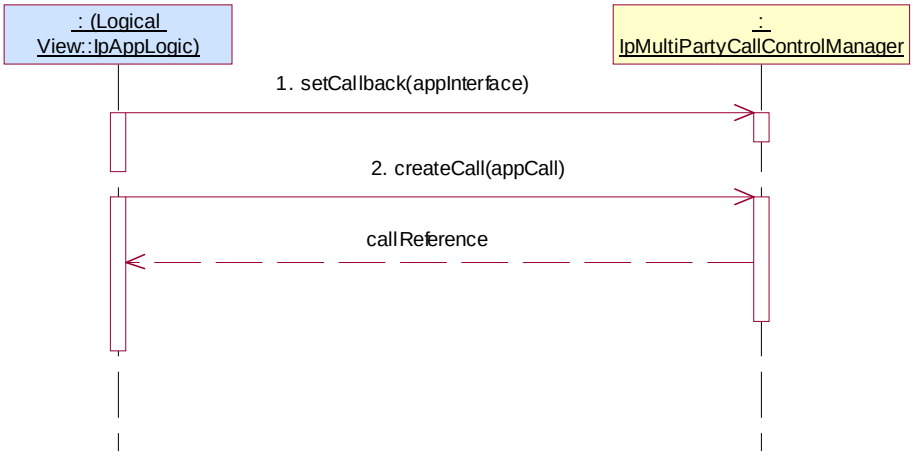
Preamble: Application has a reference interface used for callbacks.

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



Test MPCC_ IpMultiPartyCallControlManager _04

Summary: IpMultiPartyCallControlManager, all mandatory methods, successful

Reference: ES 202 915-4-3 [3], clause 6.1

Condition: enableNotifications method is supported.

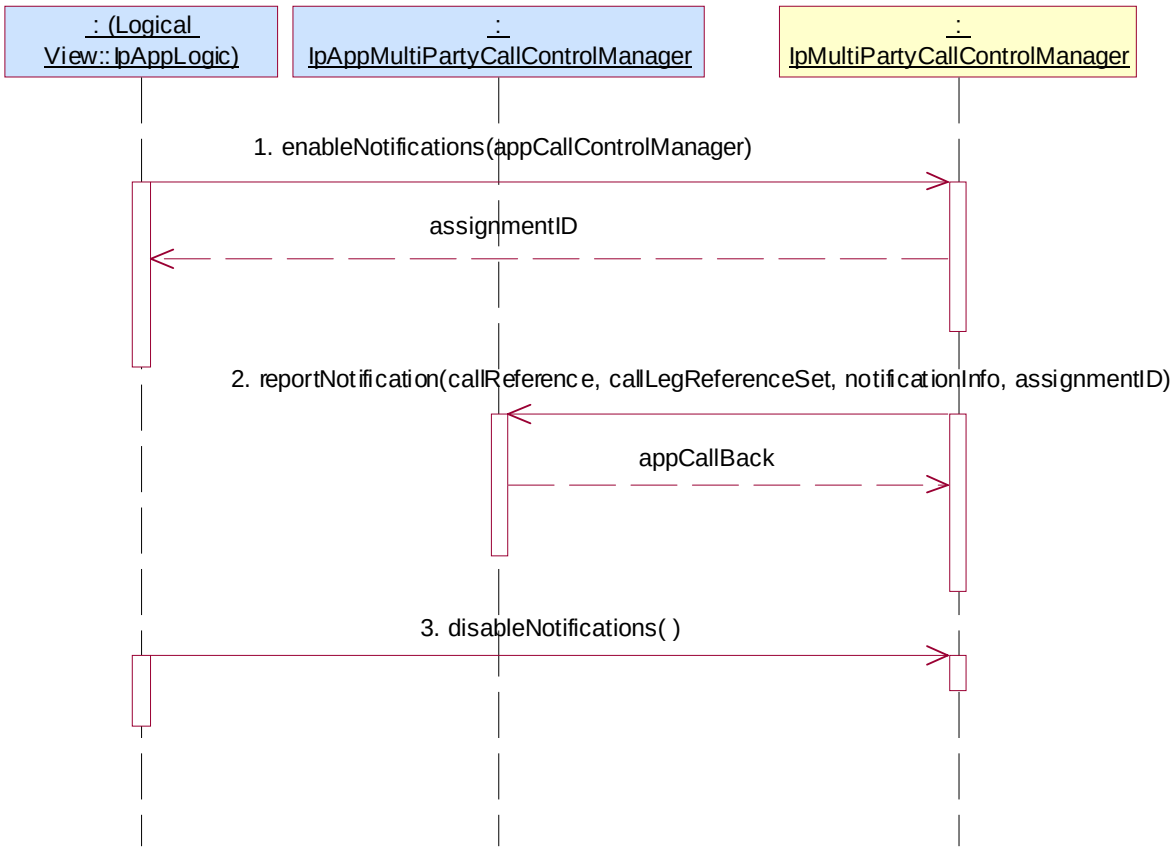
Test Sequence:

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

2. Method call **enableNotifications()**
   Parameters: appCallControlManager with null value
   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 **disableNotifications()**
   Parameters: none
   Check: no exception is returned



5.2.2.1.2 Mandatory, invalid behaviour

Test MPCC_ IpMultiPartyCallControlManager _05

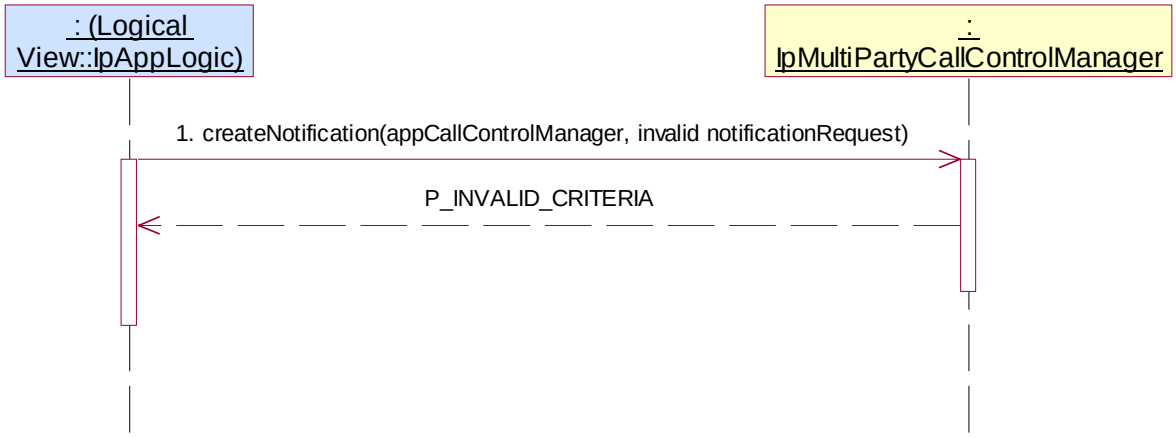
Summary: IpMultiPartyCallControlManager, createNotification, P_INVALID_CRITERIA

Reference: ES 202 915-4-3 [3], clause 6.1

Condition: createNotification method is supported.

Test Sequence:

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



Test MPCC_ IpMultiPartyCallControlManager _06

Summary: IpMultiPartyCallControlManager, destroyNotification, P_INVALID_ASSIGNMENT_ID

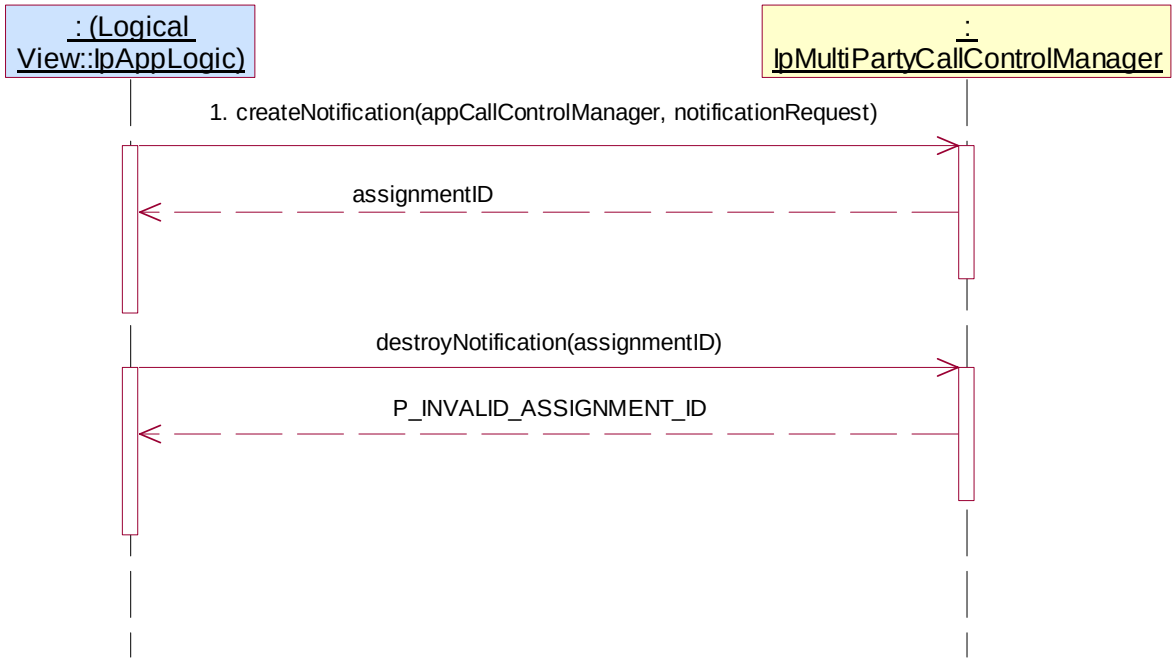
Reference: ES 202 915-4-3 [3], clause 6.1

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_ID is returned



Test MPCC_ IpMultiPartyCallControlManager _07

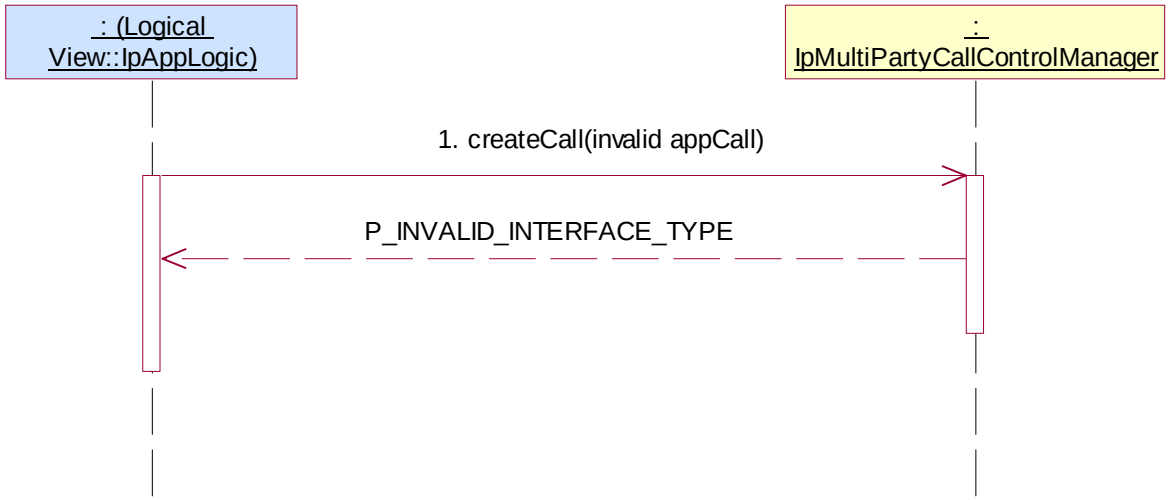
Summary: IpMultiPartyCallControlManager, createCall, P_INVALID_INTERFACE_TYPE

Reference: ES 202 915-4-3 [3], clause 6.1

Condition: createCall method is supported.

Test Sequence:

1. Method call **createCall()**
   Parameters: invalid (not Null) value of appCall
   Check: P_INVALID_INTERFACE_TYPE is returned



5.2.2.1.3 Optional, valid behaviour

Test MPCC_ IpMultiPartyCallControlManager _08

Summary: IpMultiPartyCallControlManager, getNotification, successful

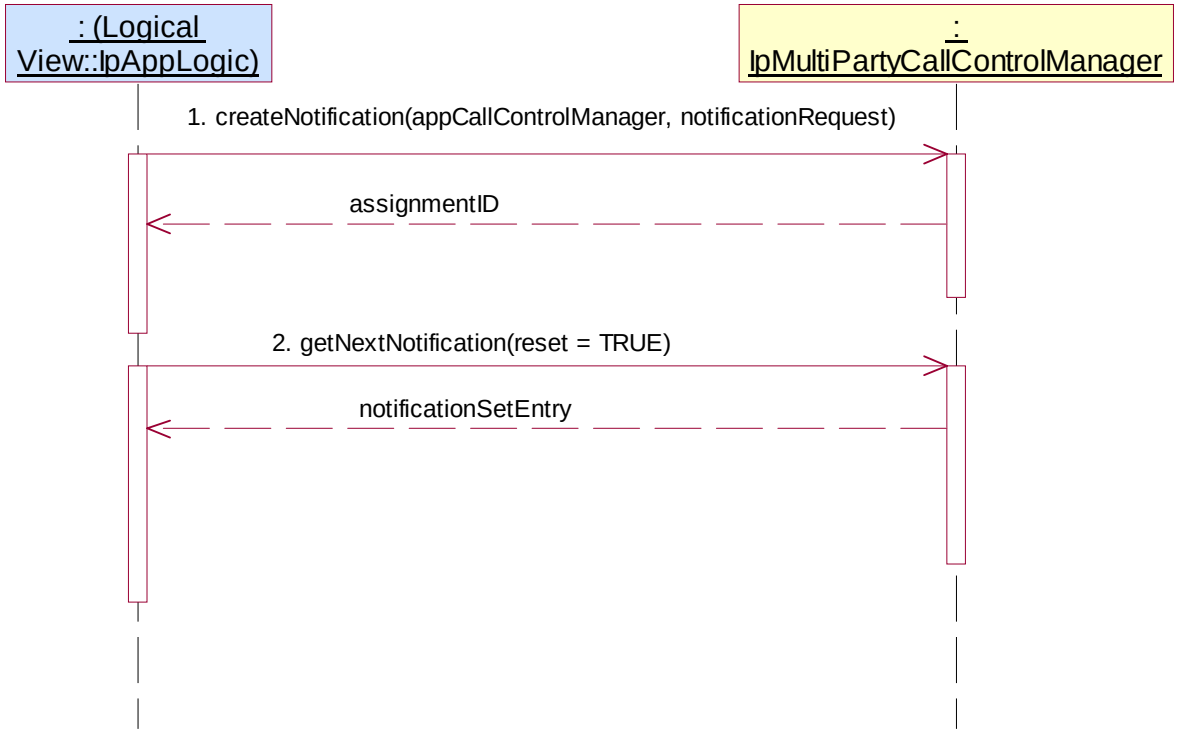
Reference: ES 202 915-4-3 [3], clause 6.1

Condition: getNextNotification 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 **getNextNotification()**
   Parameters: reset = TRUE
   Check: valid value of TpNotificationRequestedSetEntry is returned where notificationRequest given in 1. is included as a value of this TpCallEventCriteriaResult



Test MPCC_ IpMultiPartyCallControlManager _09

Summary: IpMultiPartyCallControlManager, changeNotification, successful

Reference: ES 202 915-4-3 [3], clause 6.1

Condition: createNotification, getNextNotification and changeNotification methods are supported.

Test Sequence:

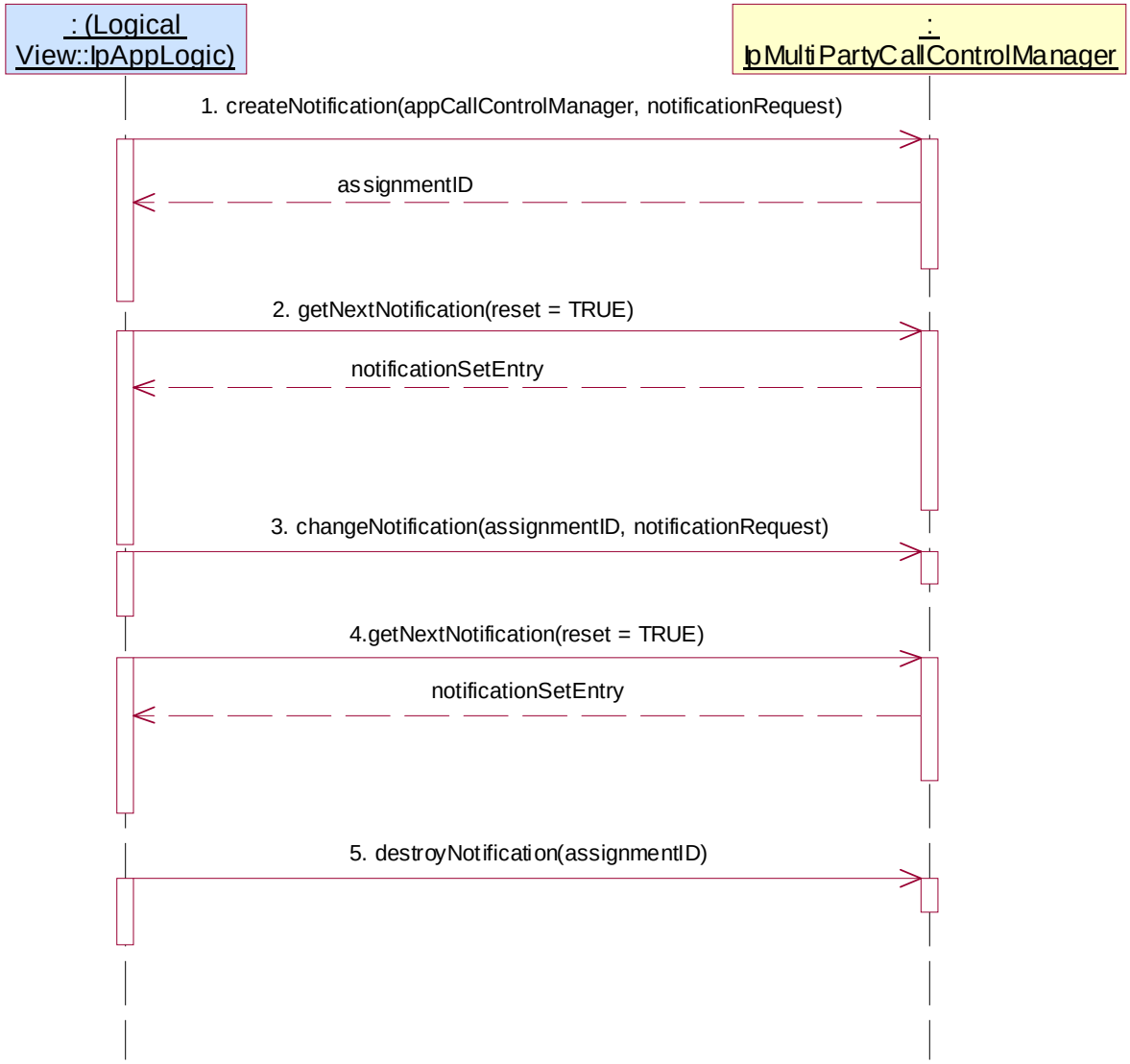
1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, non-null, value, valid notificationRequest
   Check: valid value of TpAssignmentID is returned

2. Method call **getNextNotification()**
   Parameters: reset =TRUE
   Check: valid value of TpNotificationRequestedSetEntry 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 **getNextNotification()**
   Parameters: reset = TRUE
   Check: valid value of TpNotificationRequestedSetEntry is returned where notificationRequest given in 3. 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. getNextNotification(reset = TRUE)
notificationSetEntry
3. changeNotification(assignmentID, notificationRequest)
4.getNextNotification(reset = TRUE)
notificationSetEntry
5. destroyNotification(assignmentID)

Test MPCC_ IpMultiPartyCallControlManager _10

Summary: IpMultiPartyCallControlManager, all methods, successful

Reference: ES 202 915-4-3 [3], clause 6.1

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

Test Sequence:

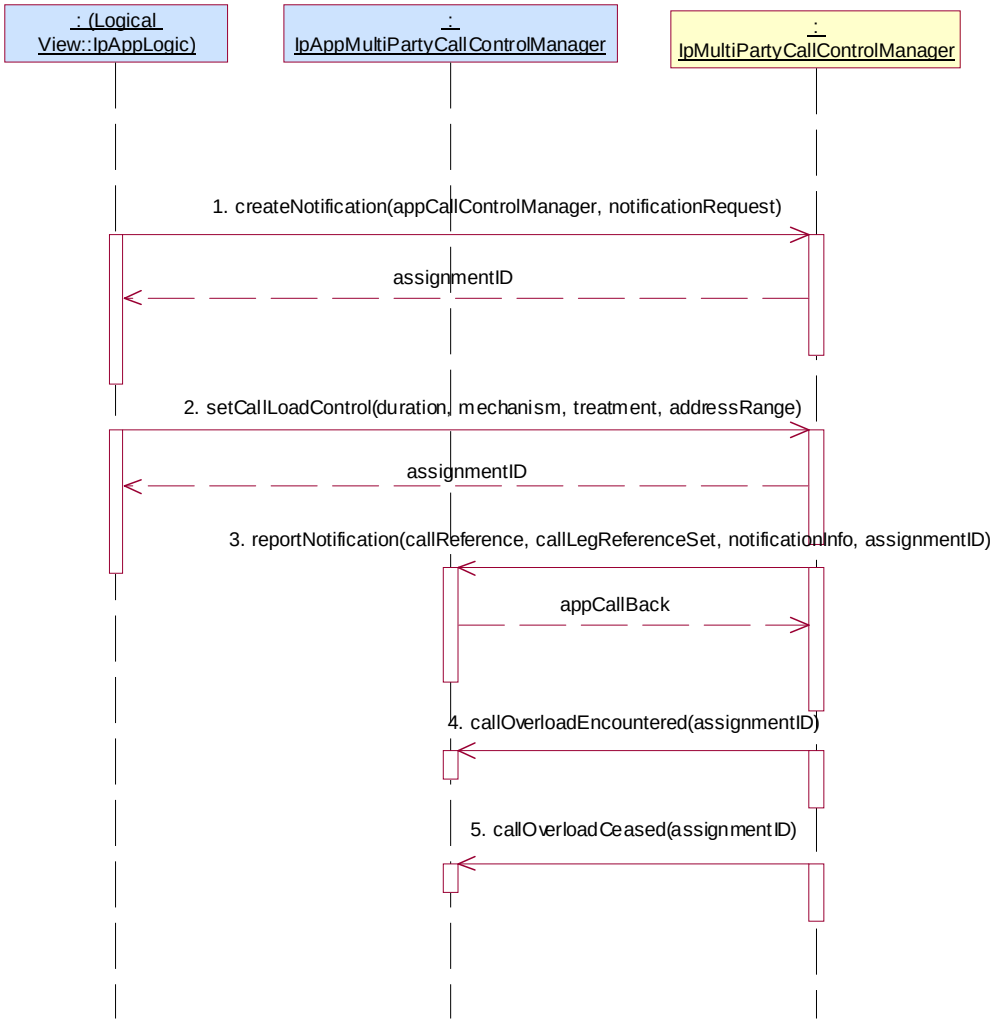
1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, non-null, value, valid notificationRequest
   Check: valid value of TpAssignmentID is returned

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) **IpAppMultiPartyCallControlManager** interface.
   Parameters: valid assignmentID

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



Test MPCC_ IpMultiPartyCallControlManager _11

Summary: IpMultiPartyCallControlManager, all methods, successful

Reference: ES 202 915-4-3 [3], clause 6.1

Condition: createNotification, managerInterrupted methods are supported.

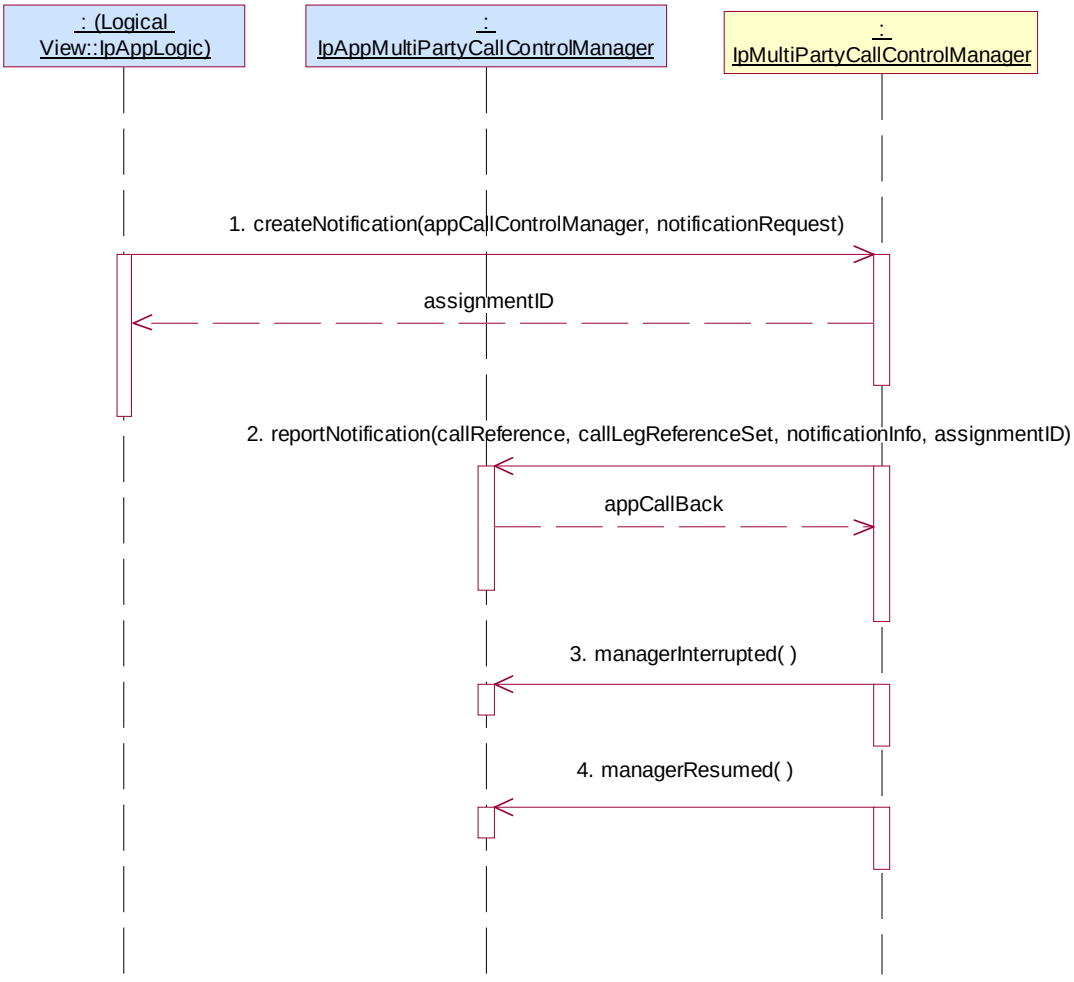
Test Sequence:

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 _12

Summary: IpMultiPartyCallControlManager, all methods, successful

Reference: ES 202 915-4-3 [3], clause 6.1

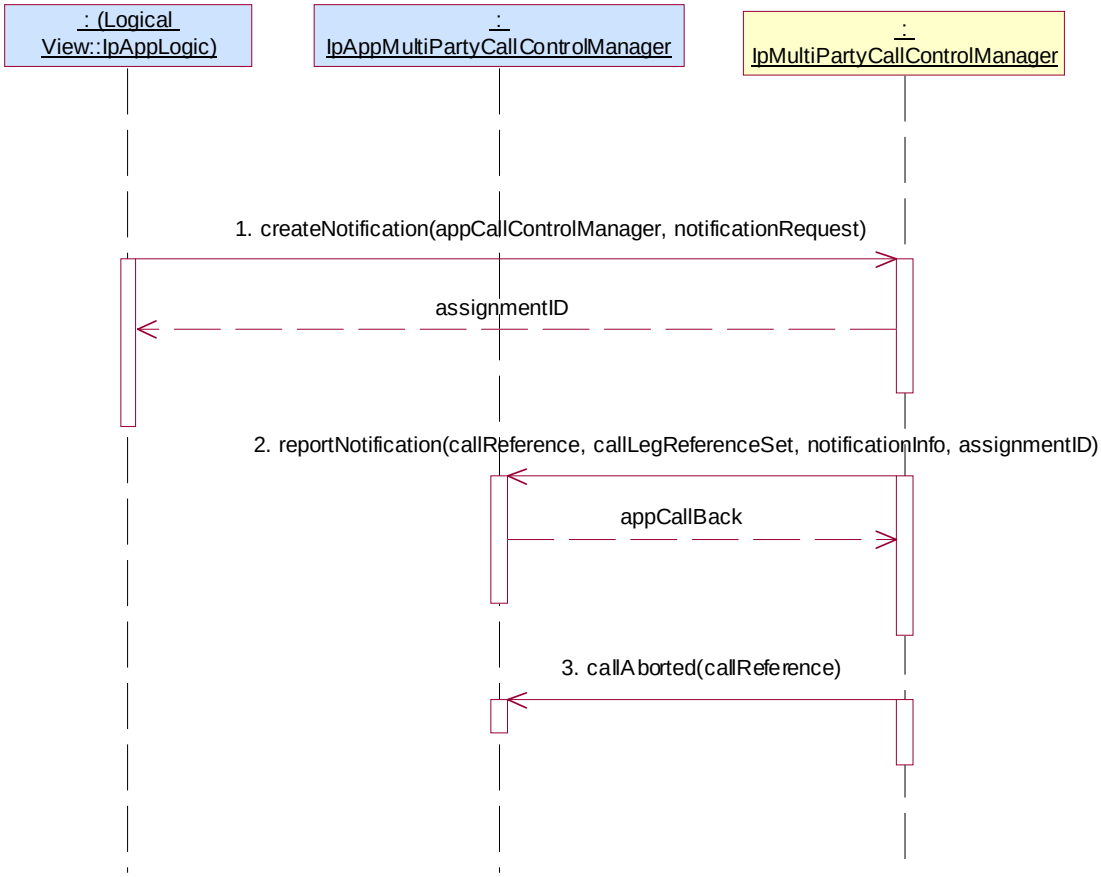
Condition: createNotification and callAborted methods are supported.

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 callReference as reported in reportNotification.



5.2.2.1.4 Optional, invalid behaviour

Test MPCC_ IpMultiPartyCallControlManager _13

Summary: IpMultiPartyCallControlManager, changeNotification, P_INVALID_ASSIGNMENT_ID

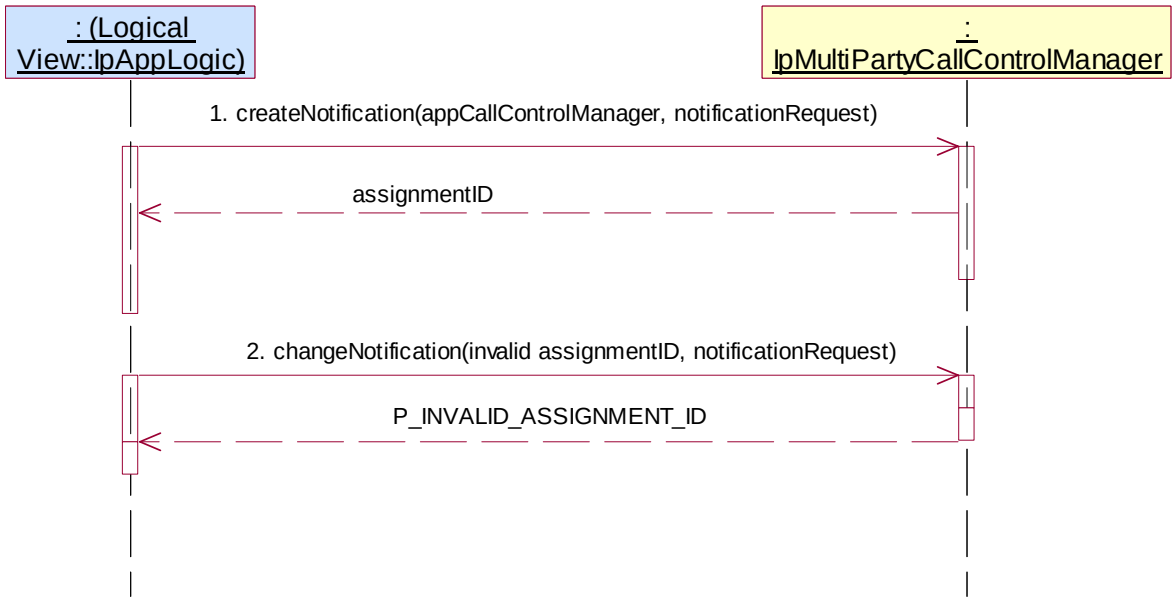
Reference: ES 202 915-4-3 [3], clause 6.1

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 is returned



Test MPCC_ IpMultiPartyCallControlManager _14

Summary: IpMultiPartyCallControlManager, changeNotification, P_INVALID_CRITERIA

Reference: ES 202 915-4-3 [3], clause 6.1

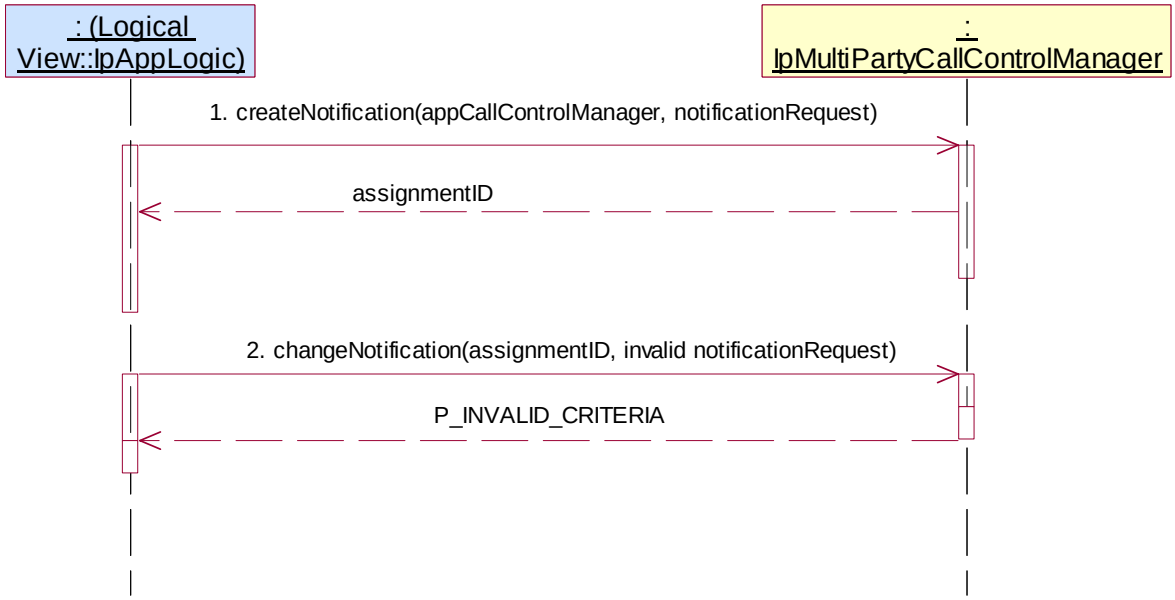
Preamble: Application has a reference interface used for callbacks.

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 is returned



Test MPCC_ IpMultiPartyCallControlManager _15

Summary: IpMultiPartyCallControlManager, setCallLoadControl, P_INVALID_ADDRESS

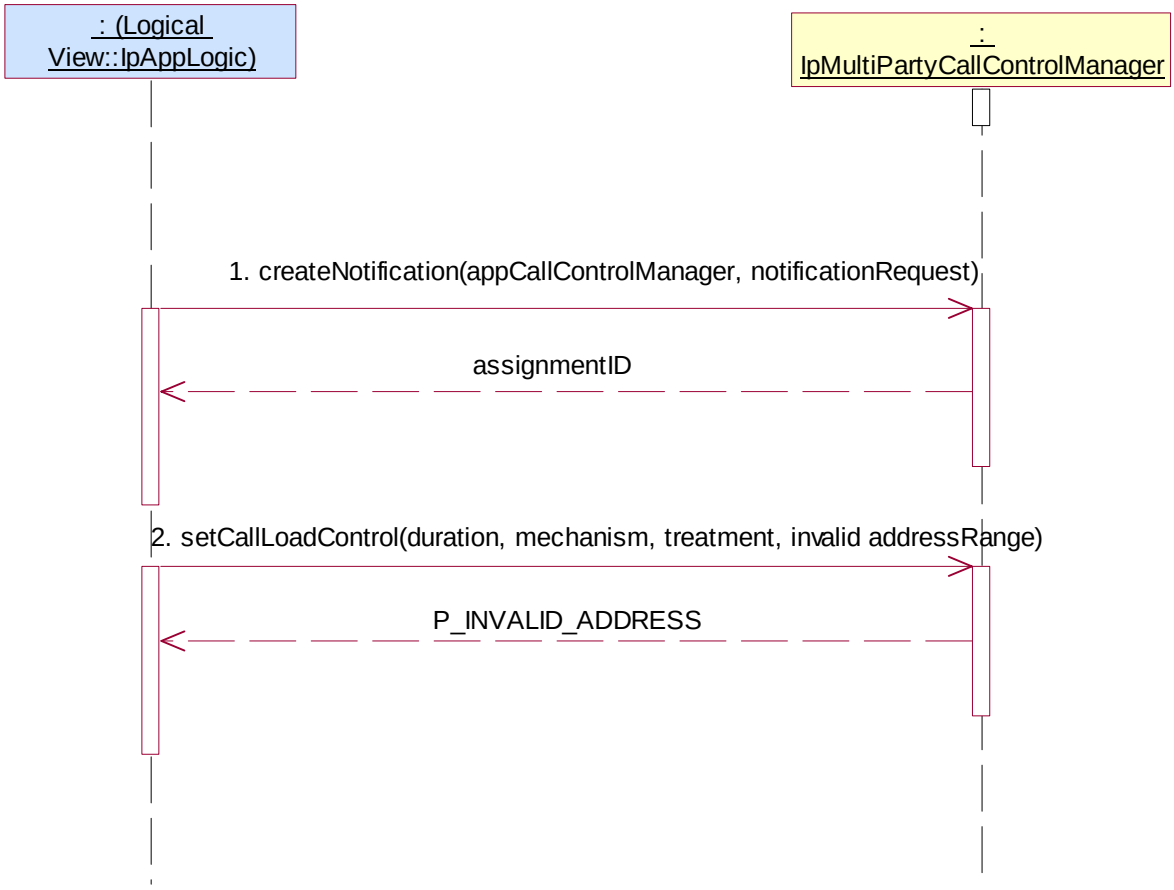
Reference: ES 202 915-4-3 [3], clause 6.1

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 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 202 915-4-3 [3], clauses 6.1 and 6.3

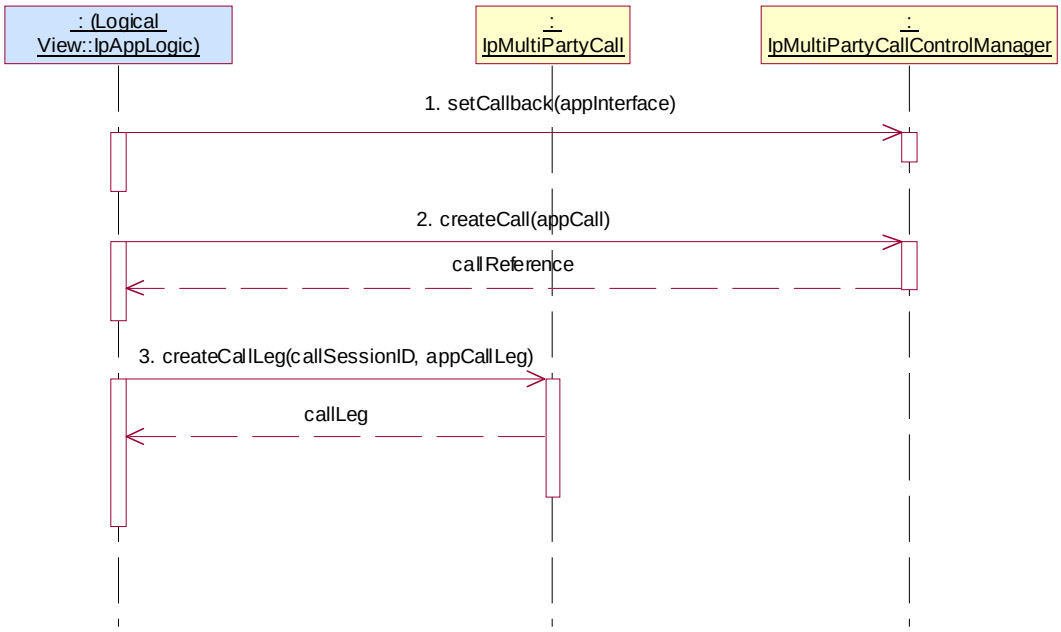
Condition: createCall and CreateCallLeg methods are supported.

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 202 915-4-3 [3], clauses 6.1 and 6.3

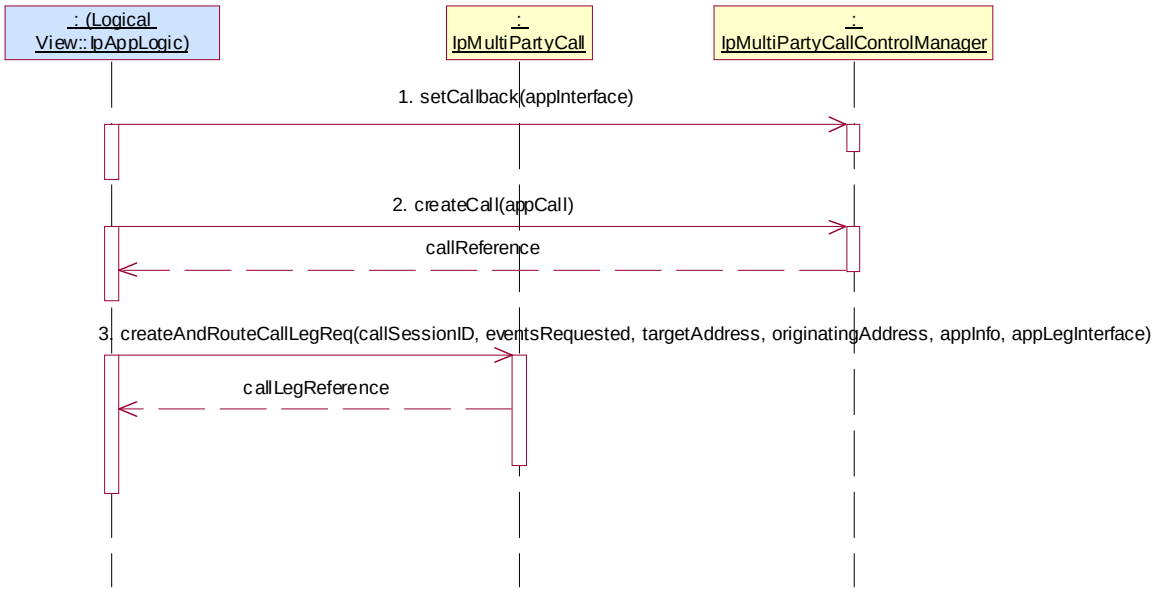
Condition: createCall and CreateAndRouteCallLeg methods are supported.

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 202 915-4-3 [3], clauses 6.1 and 6.3

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 **eventReportReq()**
   Parameters: valid callLegSessionID returned in 1., valid eventsRequested with an Interrupt event
   Check: no exception is returned

5. 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 with Interrupt event, 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 **eventReportReq()**
   Parameters: valid callLegSessionID returned in 1., valid eventsRequested with an Interrupt event
   Check: no exception is returned

5. 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 with Interrupt event, valid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: valid value of TpCallLegIdentifier

or

1. Method call **enableNotifications()**
   Parameters: appCallControlManager with valid, non-null, value
   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

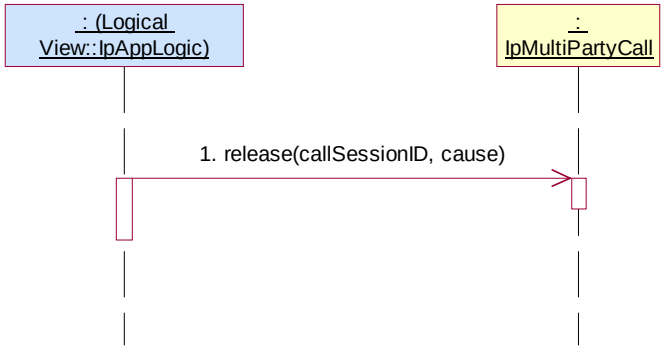
4. Method call **routeReq()**
   Parameters: valid callLegSessionID returned in 3., valid targetAddress, valid originatingAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

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 in preamble, valid cause
   Check: no exception is returned



Test MPCC_ IpMultiPartyCall _04

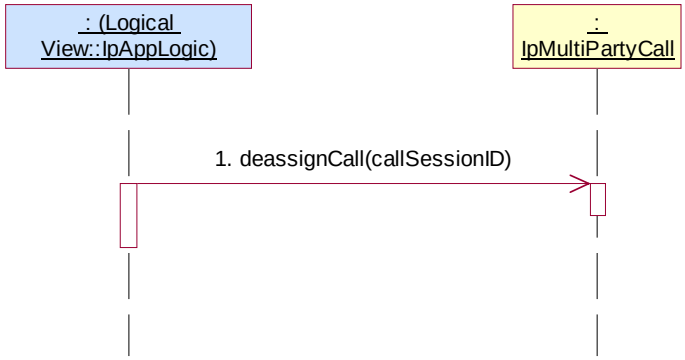
Summary: IpMultiPartyCall, all methods mandatory, successful

Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.3

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

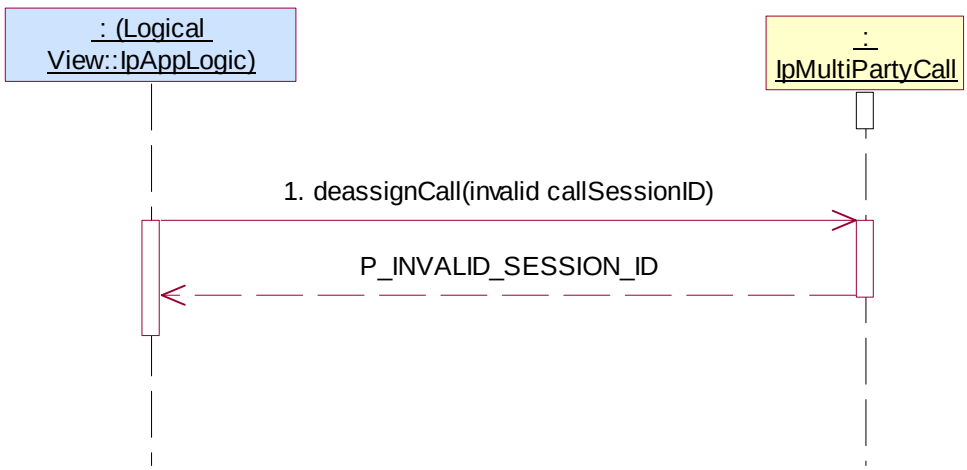
Summary: IpMultiPartyCall, deassignCall, P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.3

Preamble: Same as MPCC_ IpMultiPartyCall _03

Test Sequence:

1. Method call **deassignCall()**
   Parameters: invalid callSessionID
   Check: P_INVALID_SESSION_ID is returned



Test MPCC_ IpMultiPartyCall _06

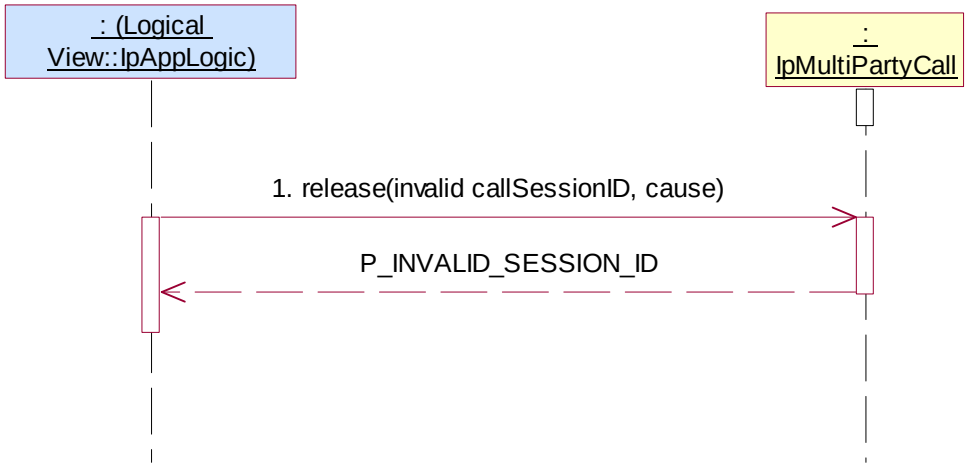
Summary: IpMultiPartyCall, release, P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.3

Preamble: Same as MPCC_ IpMultiPartyCall _03

Test Sequence:

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



Test MPCC_ IpMultiPartyCall _07

Summary: IpMultiPartyCall, createCallLeg, P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.3

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

or

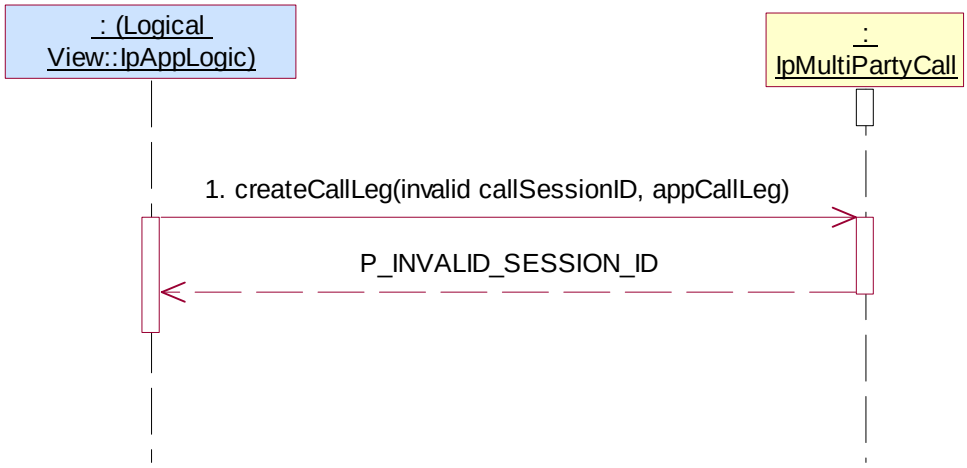
1. Method call **enableNotifications()**
   Parameters: appCallControlManager with valid, non-null, value
   Check: valid value of TpAssignmentID is returned

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 is returned



Test MPCC_ IpMultiPartyCall _08

Summary: IpMultiPartyCall, createCallLeg, P_INVALID_INTERFACE_TYPE

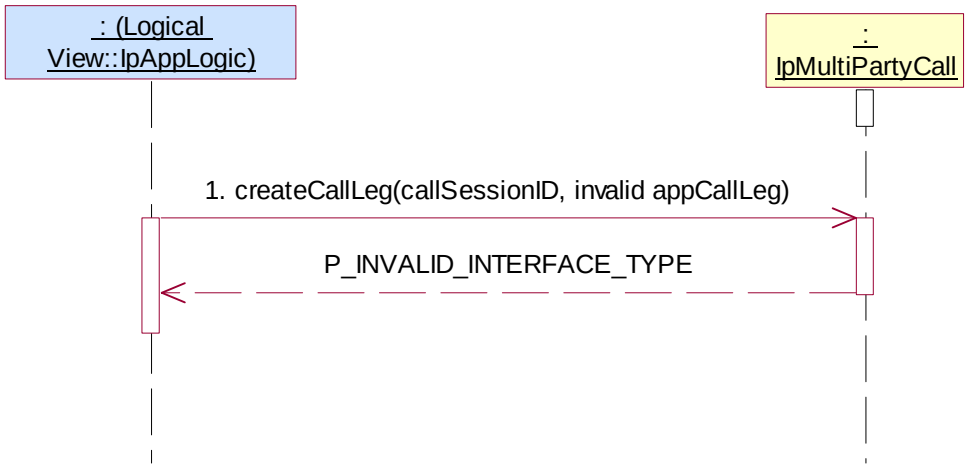
Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.3

Preamble: Same as MPCC_ IpMultiPartyCall _07

Condition: createCallLeg method is supported.

Test Sequence:

1. Method call **createCallLeg()**
   Parameters: valid callSessionID reported in preamble, invalid (not Null) appCallLeg
   Check: P_INVALID_INTERFACE_TYPE is returned



Test MPCC_ IpMultiPartyCall _09

Summary: IpMultiPartyCall, createAndRouteCallLegReq, P_INVALID_SESSION_ID

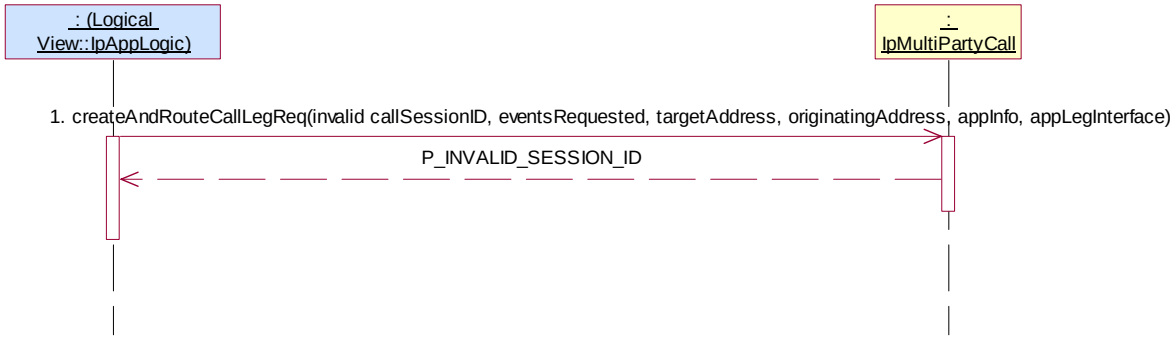
Reference: ES 202 915-4-3 [3], clause 6.3

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 is returned



Test MPCC_ IpMultiPartyCall _10

Summary: IpMultiPartyCall, createAndRouteCallLegReq, P_INVALID_INTERFACE_TYPE

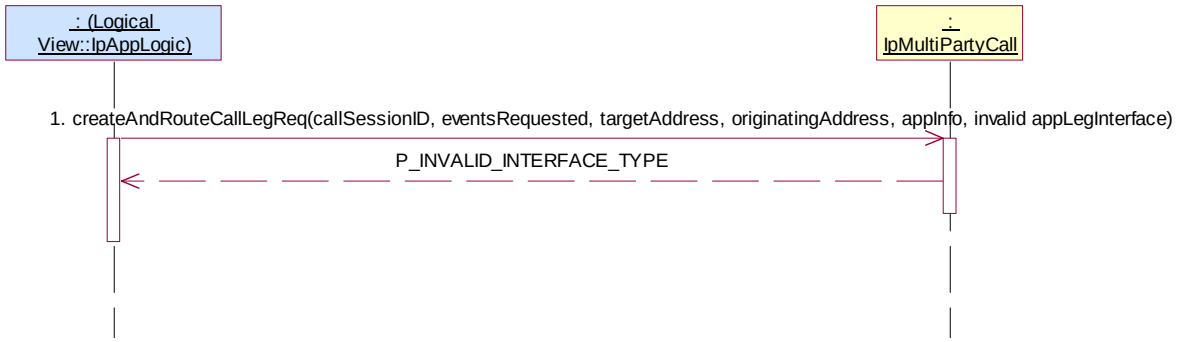
Reference: ES 202 915-4-3 [3], clause 6.3

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 (not Null) appLegInterface
   Check: P_INVALID_INTERFACE_TYPE is returned



Test MPCC_ IpMultiPartyCall _11

Summary: IpMultiPartyCall, createAndRouteCallLegReq, P_INVALID_ADDRESS

Reference: ES 202 915-4-3 [3], clause 6.3

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 is returned

Test MPCC_ IpMultiPartyCall _12

Summary: IpMultiPartyCall, createAndRouteCallLegReq, P_INVALID_ADDRESS

Reference: ES 202 915-4-3 [3], clause 6.3

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 is returned

Test MPCC_ IpMultiPartyCall _13

Summary: IpMultiPartyCall, createAndRouteCallLegReq, P_INVALID_CRITERIA

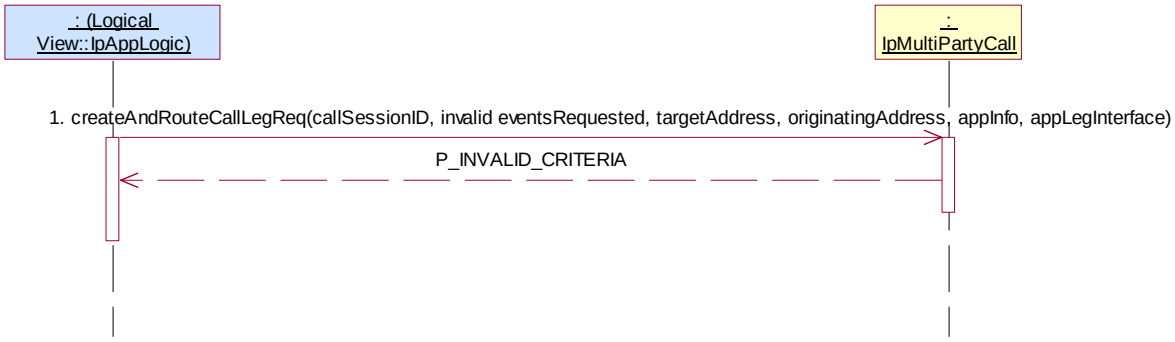
Reference: ES 202 915-4-3 [3], clause 6.3

Preamble: Same as MPCC_ IpMultiPartyCall _07

Condition: createAndRouteCallLegReq method is supported.

Test Sequence:

1. Method call **createAndRouteCallLegReq()**
   Parameters: valid callSessionID, invalid eventsRequested but with valid event type, valid targetAddress, valid originatingAddress, valid appInfo, valid appLegInterface
   Check: P_INVALID_CRITERIA is returned



5.2.2.2.3 Optional, valid behaviour

Test MPCC_ IpMultiPartyCall _14

Summary: IpMultiPartyCall, getInfoReq, successful

Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.3

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

or

1. Method call **enableNotifications()**
   Parameters: appCallControlManager with valid, non-null, value
   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

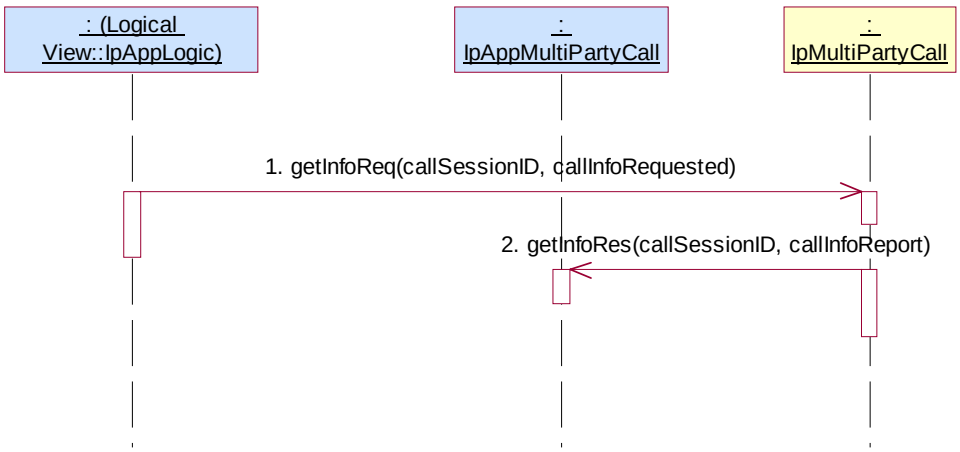
3. Method call **createCallLeg ()**
   Parameters: valid callSessionID reported in 2., valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

Condition: createCallLeg and getInfoReq methods are supported.

Test Sequence:

1. Method call **getInfoReq()**
   Parameters: valid callSessionID reported in preamble, 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

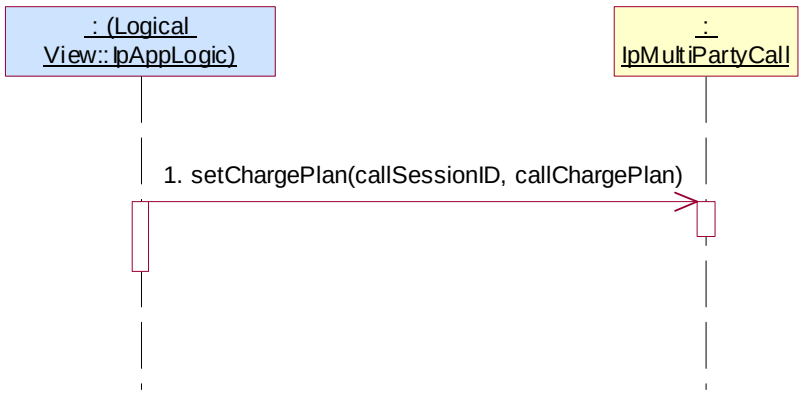
Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.3

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

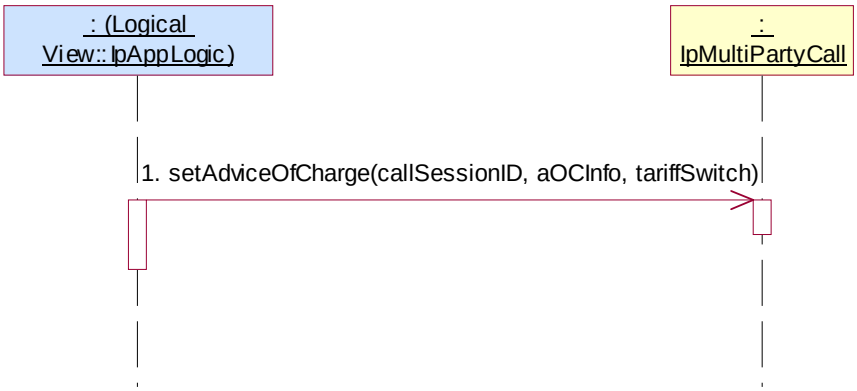
Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.3

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 202 915-4-3 [3], clauses 6.1 and 6.3

Preamble: Same as MPCC_ IpMultiPartyCall _14

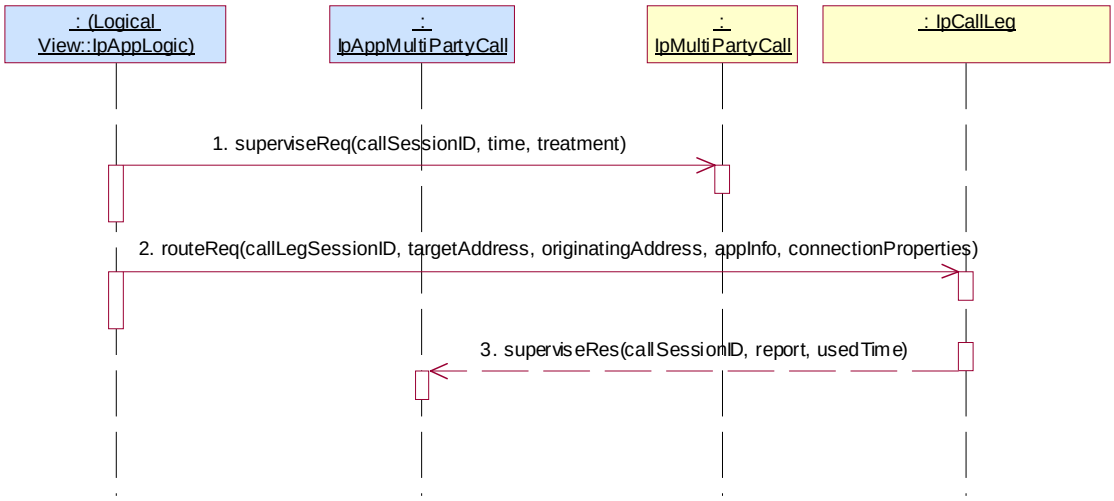
Condition: createCallLeg and superviseReq methods are supported.

Test Sequence:

1. Method call **superviseReq()**
   Parameters: valid callSessionID reported in preamble, 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

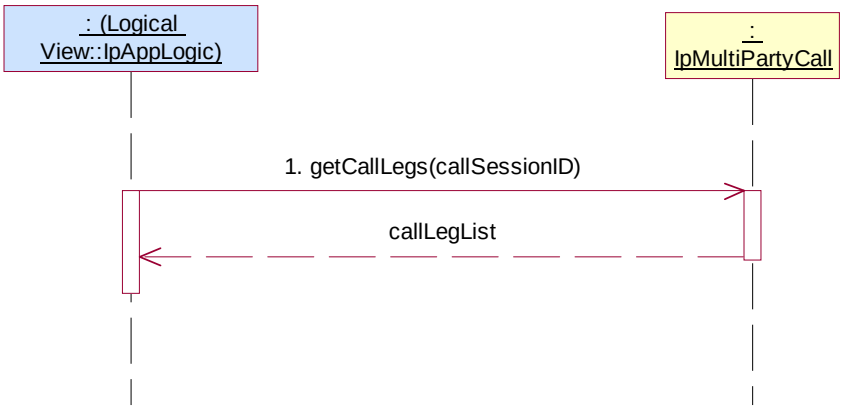
Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.3

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

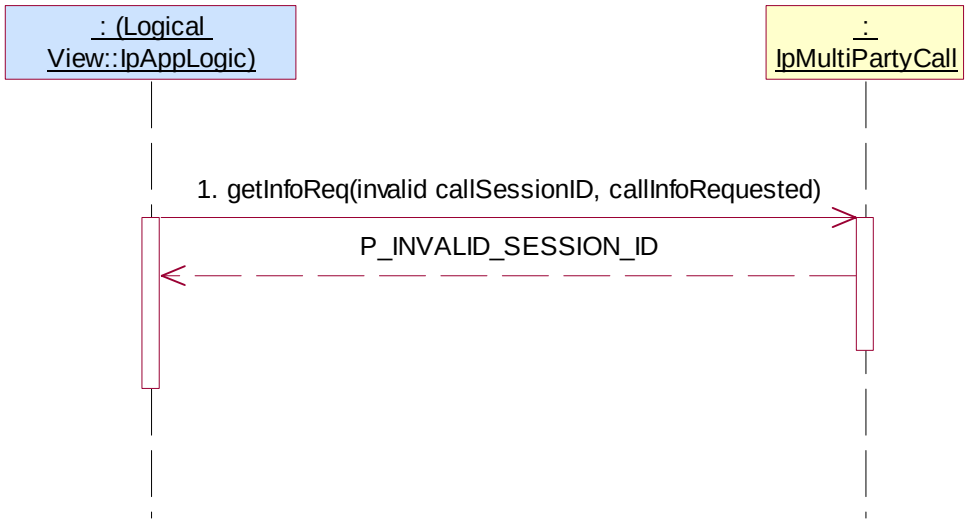
Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.3

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 is returned



Test MPCC_ IpMultiPartyCall _20

Summary: IpMultiPartyCall, setChargePlan, P_INVALID_SESSION_ID

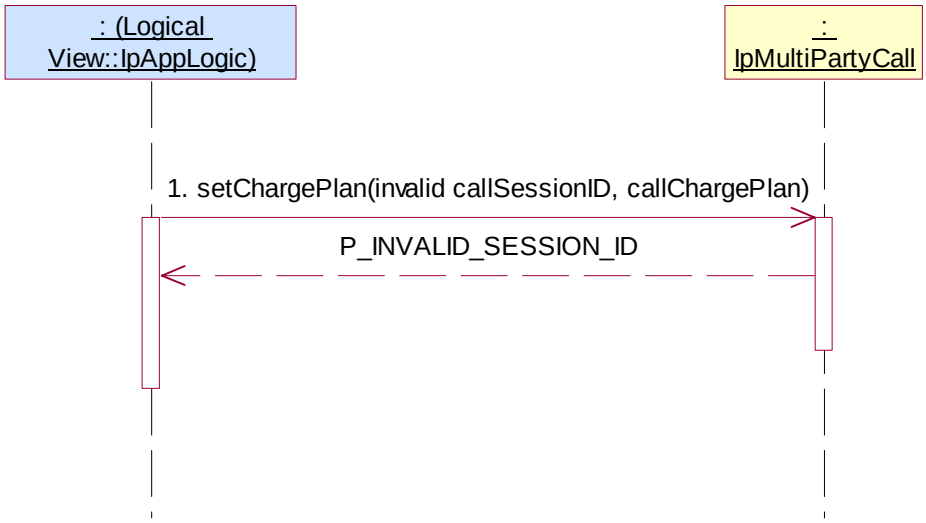
Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.3

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 is returned



Test MPCC_ IpMultiPartyCall _21

Summary: IpMultiPartyCall, setAdviceOfCharge, P_INVALID_SESSION_ID

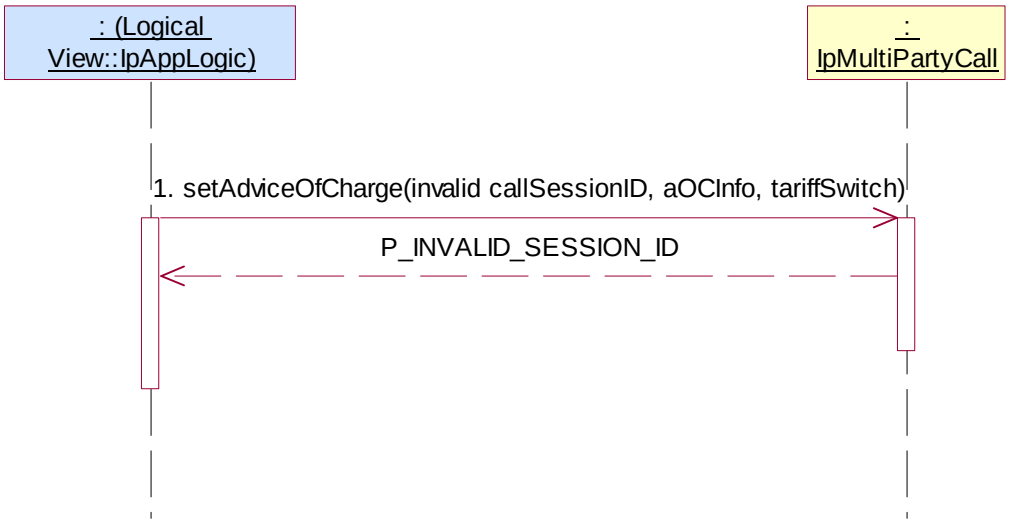
Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.3

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 is returned



Test MPCC_ IpMultiPartyCall _22

Summary: IpMultiPartyCall, setAdviceOfCharge, P_INVALID_CURRENCY

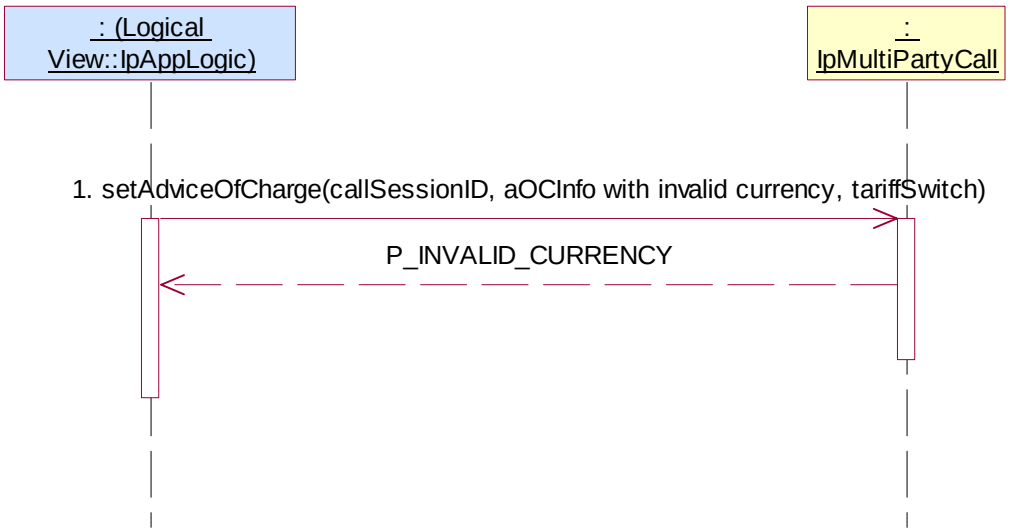
Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.3

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 is returned.



Test MPCC_ IpMultiPartyCall _23

Summary: IpMultiPartyCall, setAdviceOfCharge, P_INVALID_AMOUNT

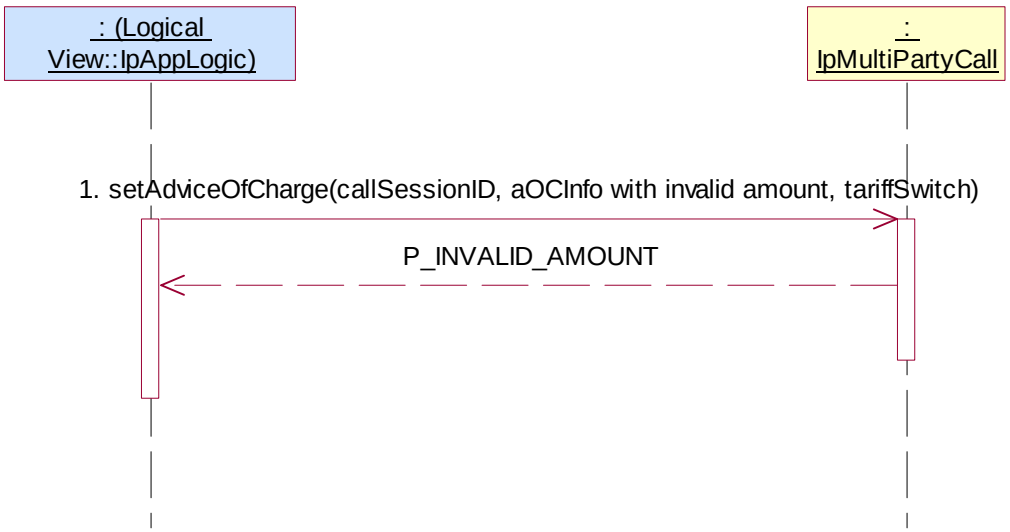
Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.3

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 is returned.



Test MPCC_ IpMultiPartyCall _24

Summary: IpMultiPartyCall, superviseReq, P_INVALID_SESSION_ID

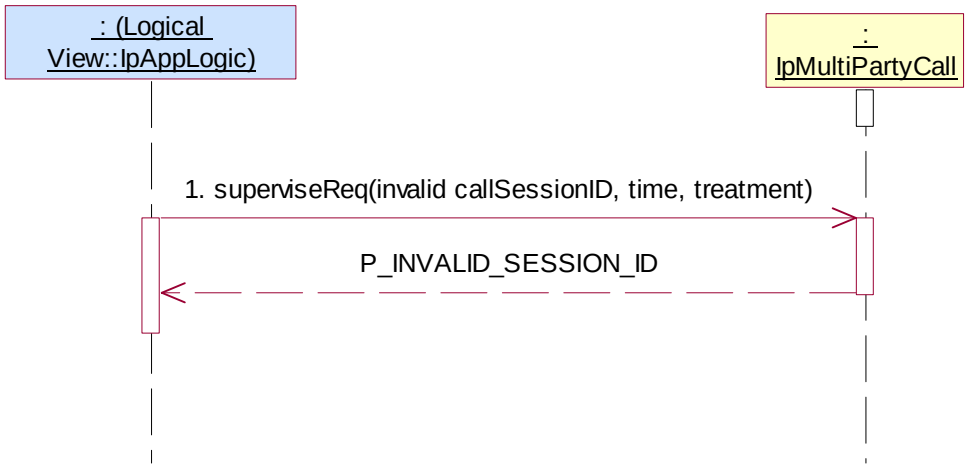
Reference: ES 202 915-4-3 [3], clause 6.3

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 is returned



Test MPCC_ IpMultiPartyCall _25

Summary: IpMultiPartyCall, getCallLegs, P_INVALID_SESSION_ID

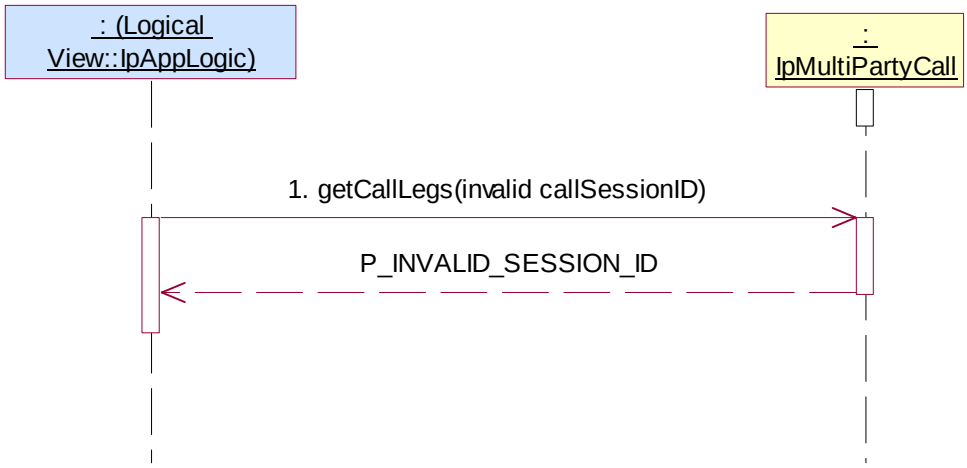
Reference: ES 202 915-4-3 [3], clause 6.3

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 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 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

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

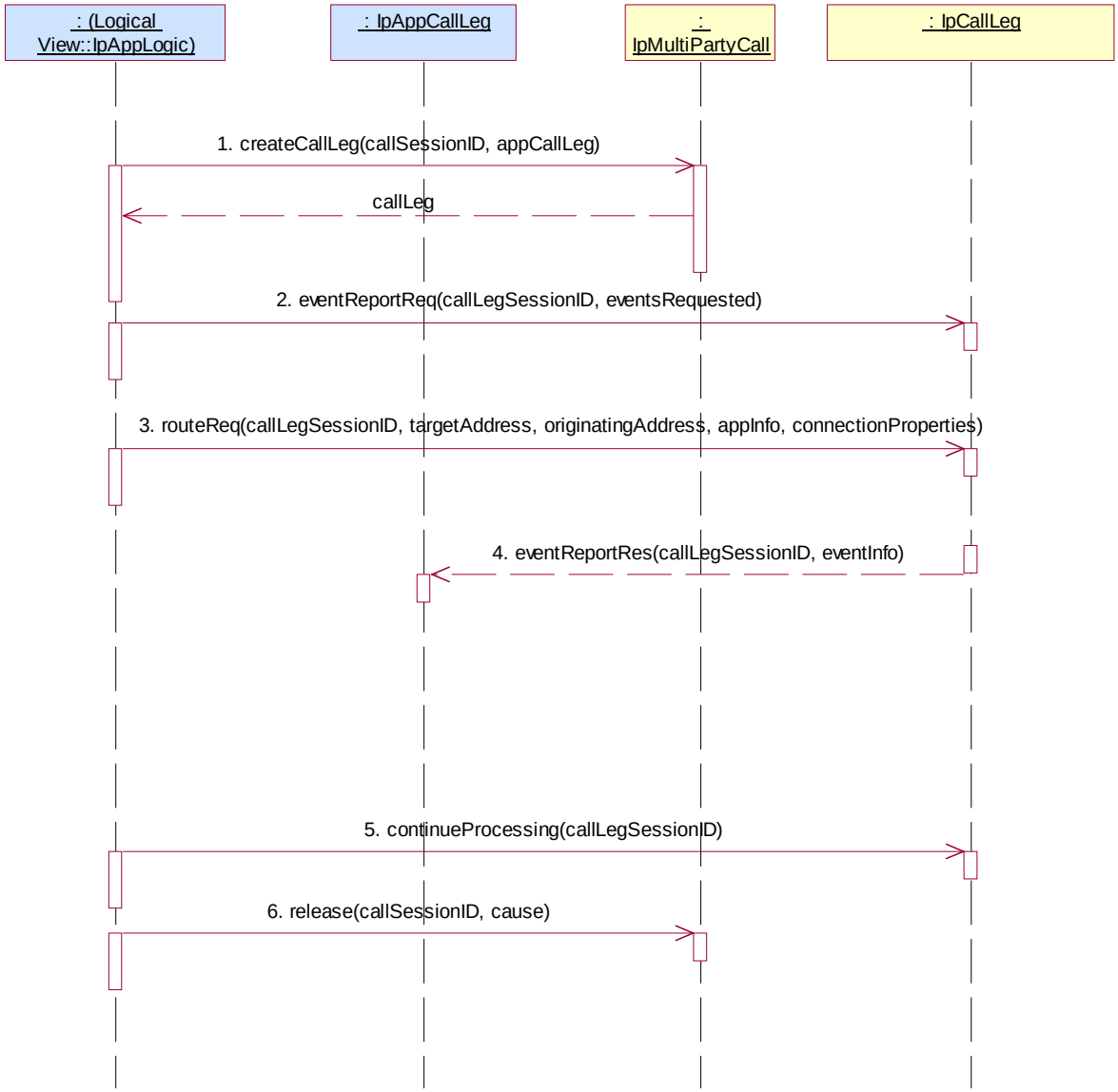
2. Method call **eventReportReq()**
   Parameters: valid callLegSessionID returned in 1., valid eventsRequested with an Interrupt event
   Check: no exception is returned

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, eventInfo

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)
5. continueProcessing(callLegSessionID)
6. release(callSessionID, cause)

Test MPCC_ IpCallLeg _02

Summary: IpCallLeg, all mandatory methods, successful

Reference: ES 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

Preamble: Same as MPCC_ IpMultiPartyCall _07

Condition: createAndRouteCallLeg method is supported.

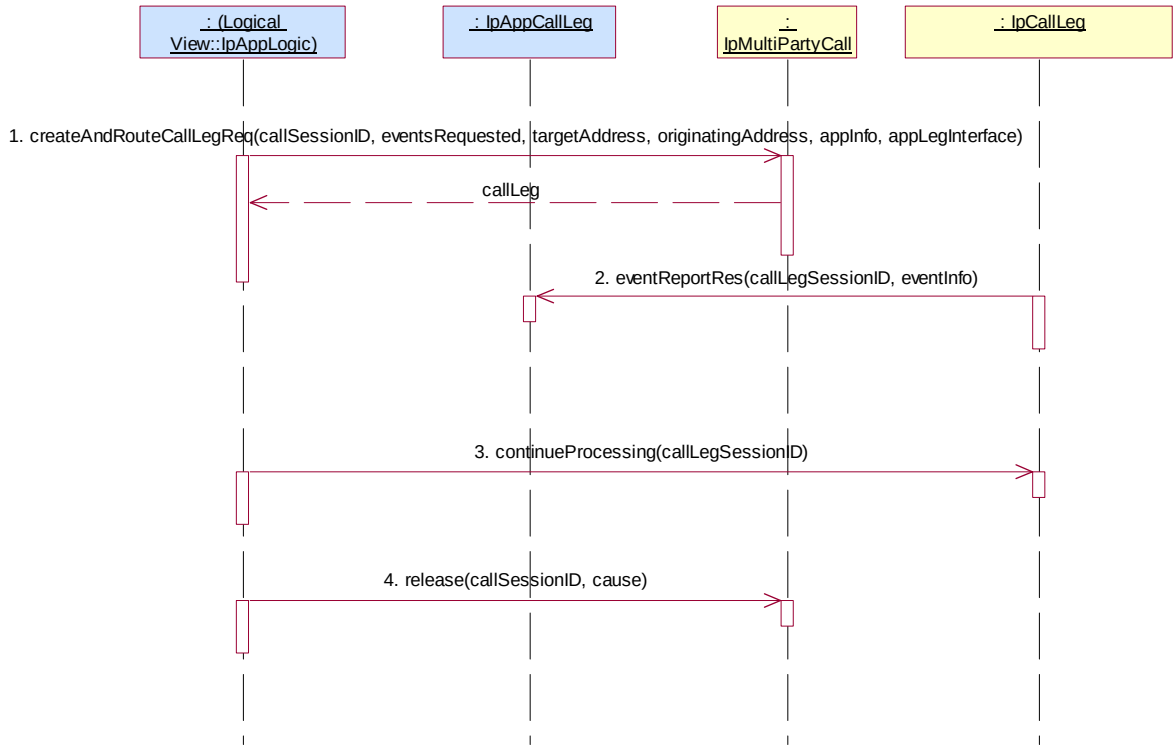
Test Sequence:

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

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 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

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

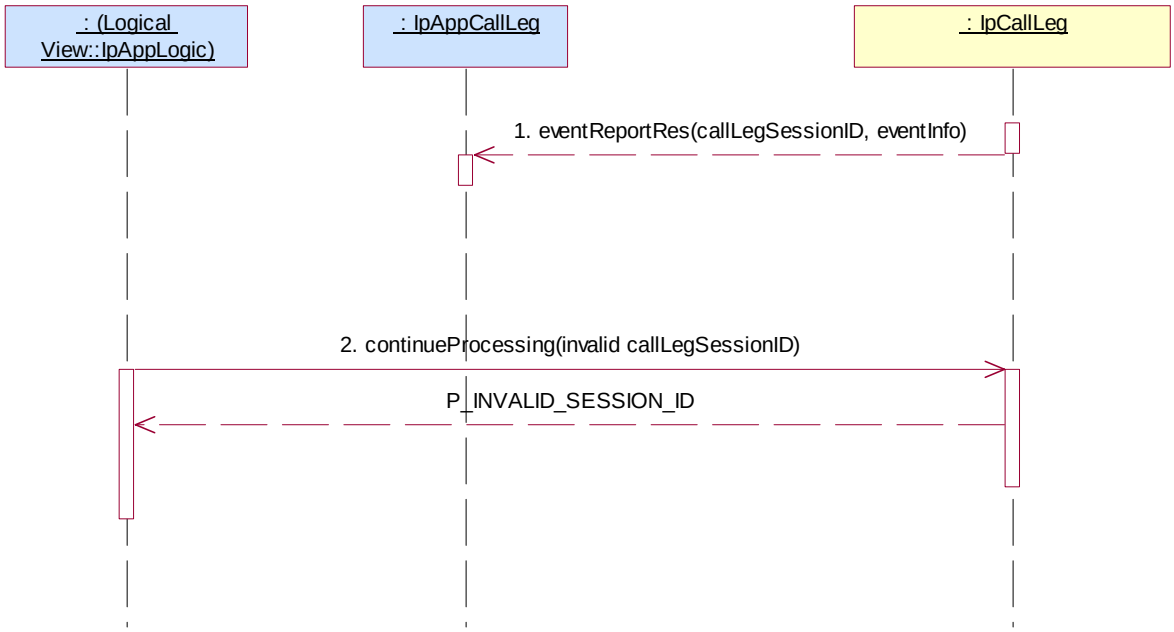
Reference: ES 202 915-4-3 [3], clause 6.5

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, eventInfo

2. Method call **continueProcessing()**
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID is returned



Test MPCC_ IpCallLeg _05

Summary: IpCallLeg, routeReq, P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.5

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 is returned

Test MPCC_ IpCallLeg _06

Summary: IpCallLeg, routeReq, P_INVALID_ADDRESS

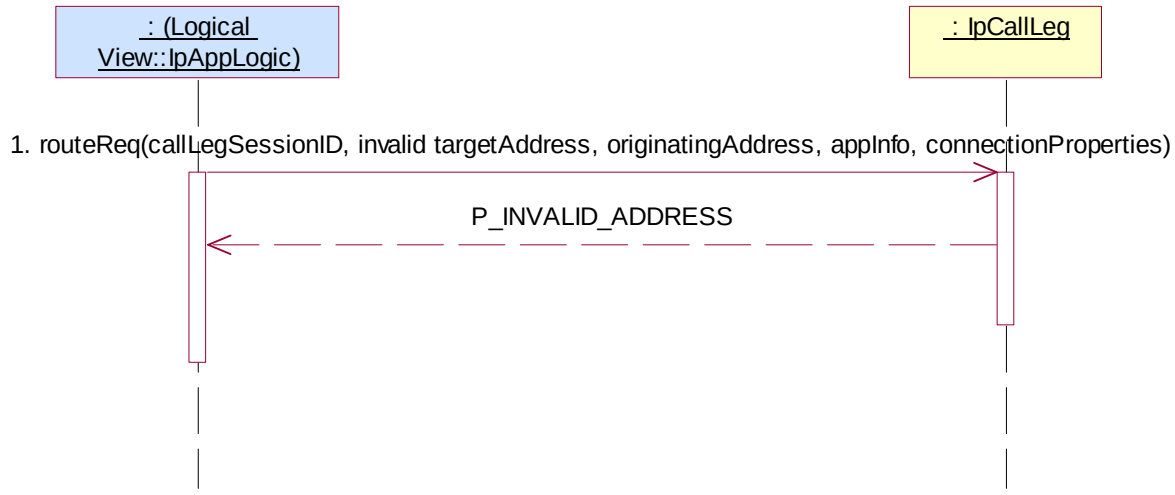
Reference: ES 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

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 is returned



Test MPCC_ IpCallLeg _07

Summary: IpCallLeg, routeReq, P_INVALID_ADDRESS

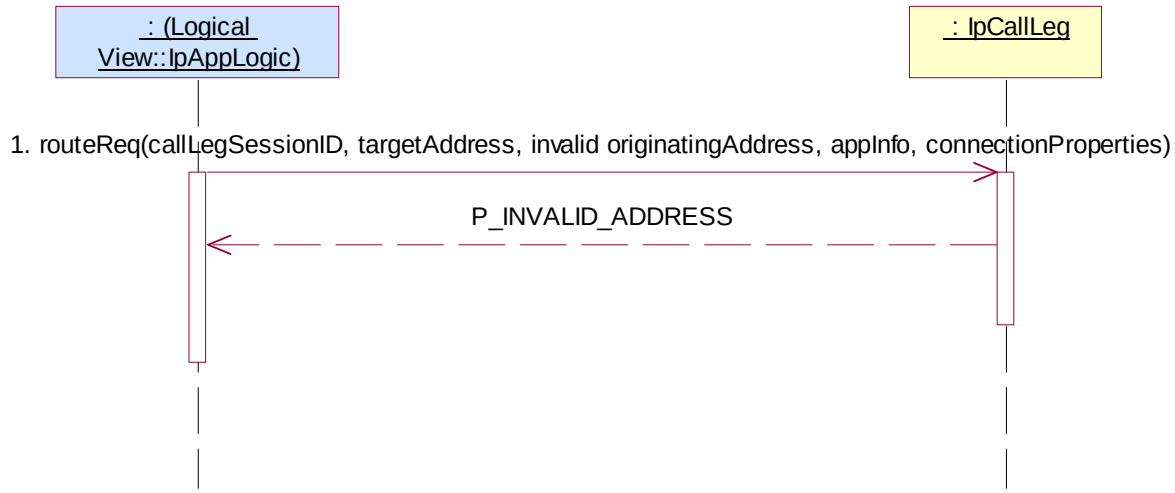
Reference: ES 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

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 is returned



Test MPCC_ IpCallLeg _08

Summary: IpCallLeg, eventReportReq**,** P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.5

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 is returned

Test MPCC_ IpCallLeg _09

Summary: IpCallLeg, eventReportReq, P_INVALID_CRITERIA

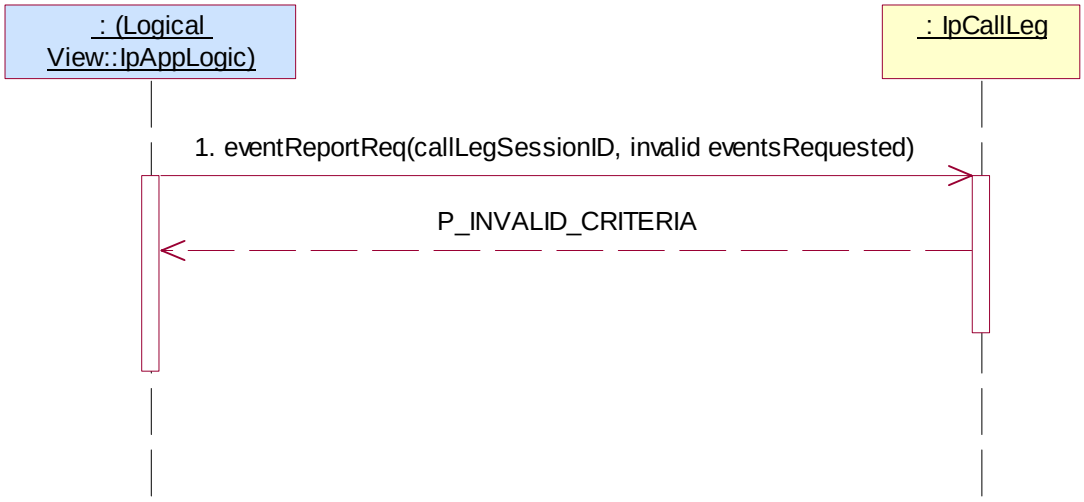
Reference: ES 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

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 but with valid event type
   Check: P_INVALID_CRITERIA is returned



Test MPCC_ IpCallLeg _10

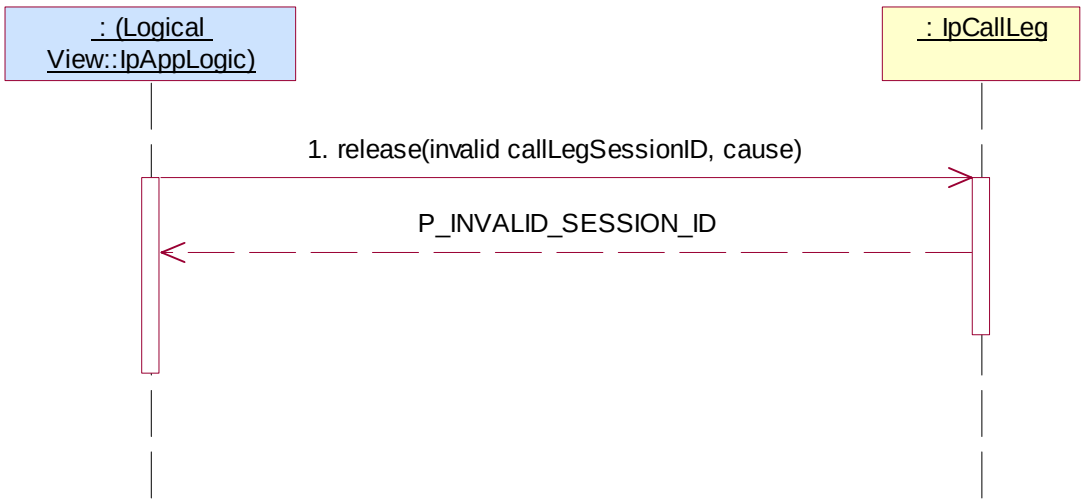
Summary: IpCallLeg, release**,** P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.5

Preamble: Same as MPCC_ IpMultiPartyCall _03

Test Sequence:

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



Test MPCC_ IpCallLeg _11

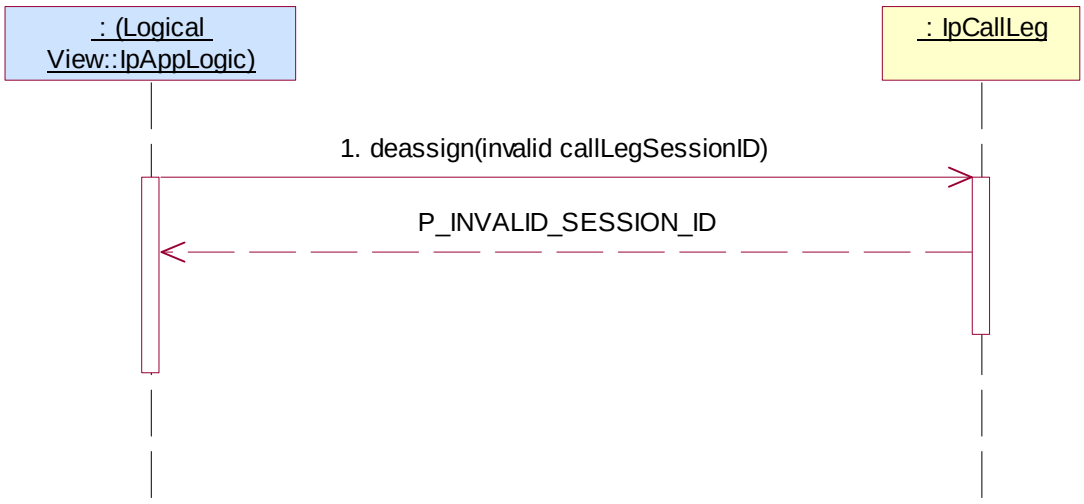
Summary: IpCallLeg, deassign**,** P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.5

Preamble: Same as MPCC_ IpMultiPartyCall _03

Test Sequence:

1. Method call **deassign()**
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID is returned.



5.2.2.3.3 Optional, valid behaviour

Test MPCC_ IpCallLeg _12

Summary: IpCallLeg, getInfoReq, successful

Reference: ES 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

Preamble: Same as MPCC_ IpMultiPartyCall _14

Condition: createCallLeg and getInfoReq methods are supported.

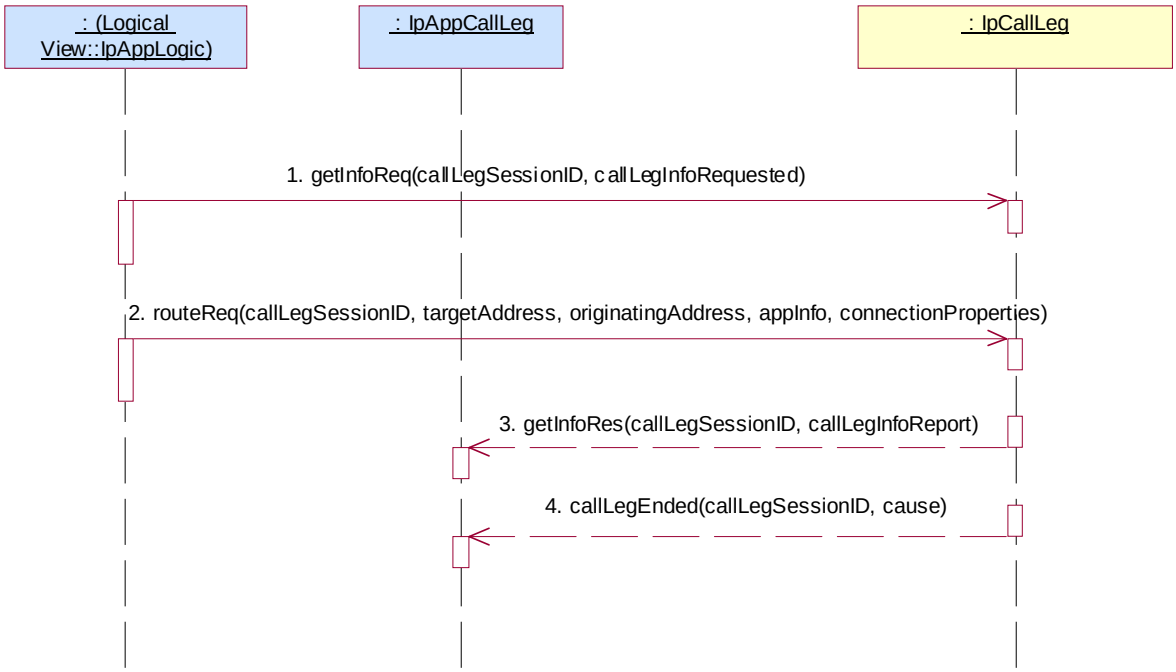
Test Sequence:

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

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

4. 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 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

Preamble: Same as MPCC_ IpMultiPartyCall _07

Condition: createCallLeg and attachMediaReq methods are supported.

Test Sequence:

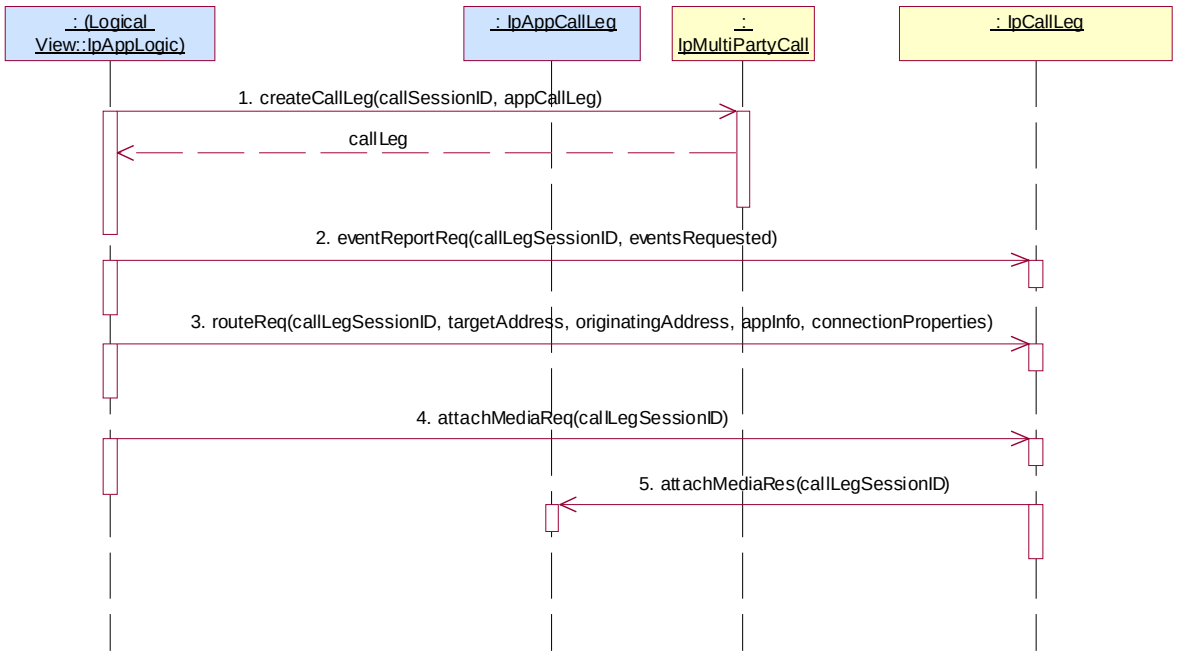
1. Method call **createCallLeg()**
   Parameters: valid callSessionID reported in preamble, valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

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 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

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

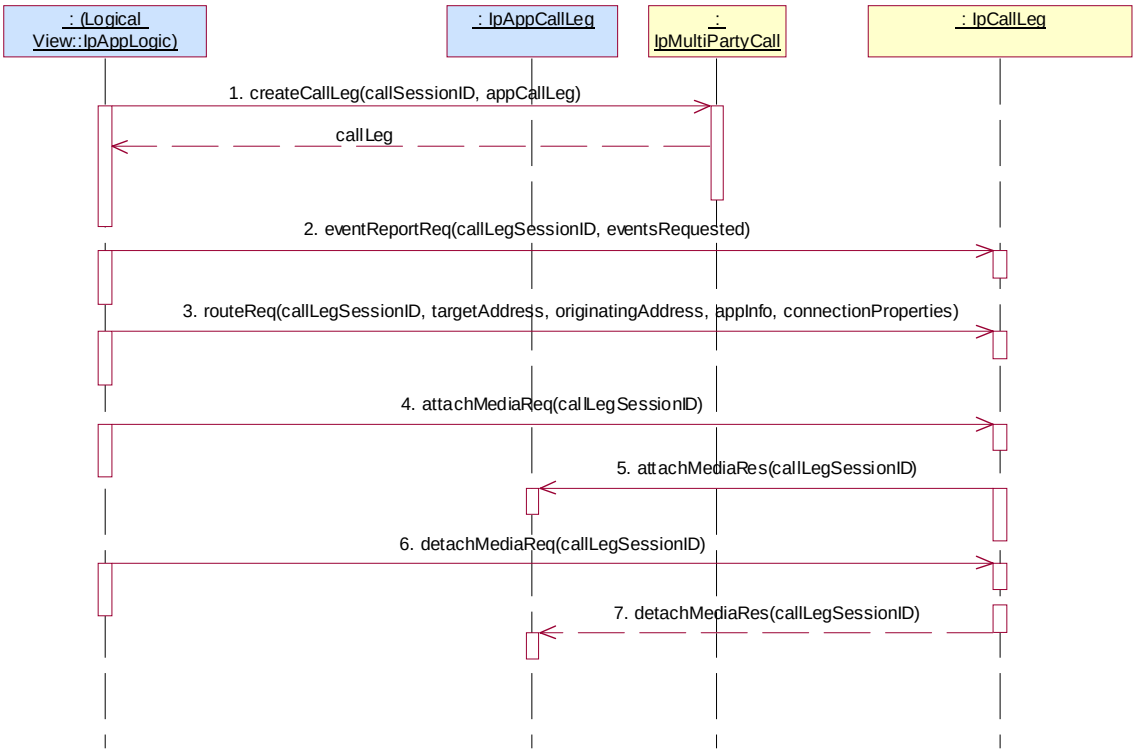
3. Method call **routeReq()**
   Parameters: valid callLegSessionID returned in 1., valid targetAddress, valid originatingAddress, valid appInfo, valid connectionProperties set to have explicit media management
   Check: no exception is returned

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 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

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

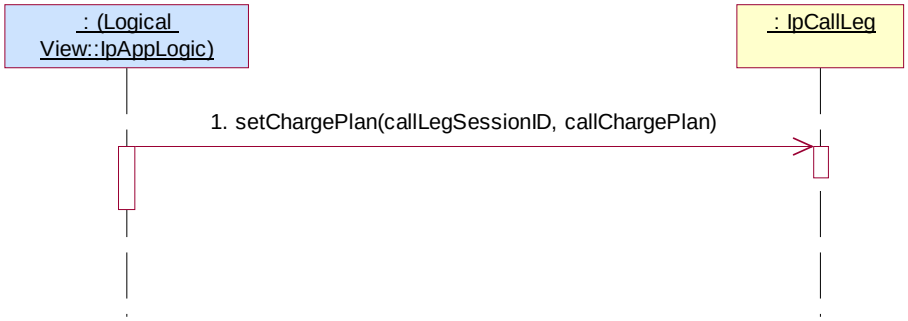
Reference: ES 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

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

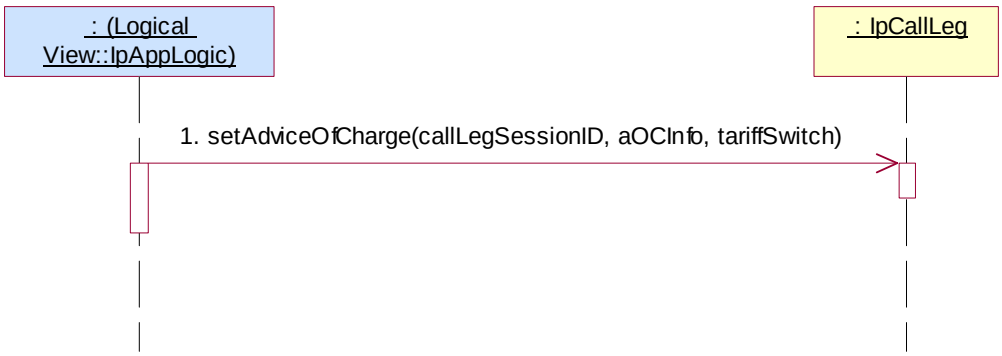
Reference: ES 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

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 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

Preamble: Same as MPCC_ IpMultiPartyCall _14

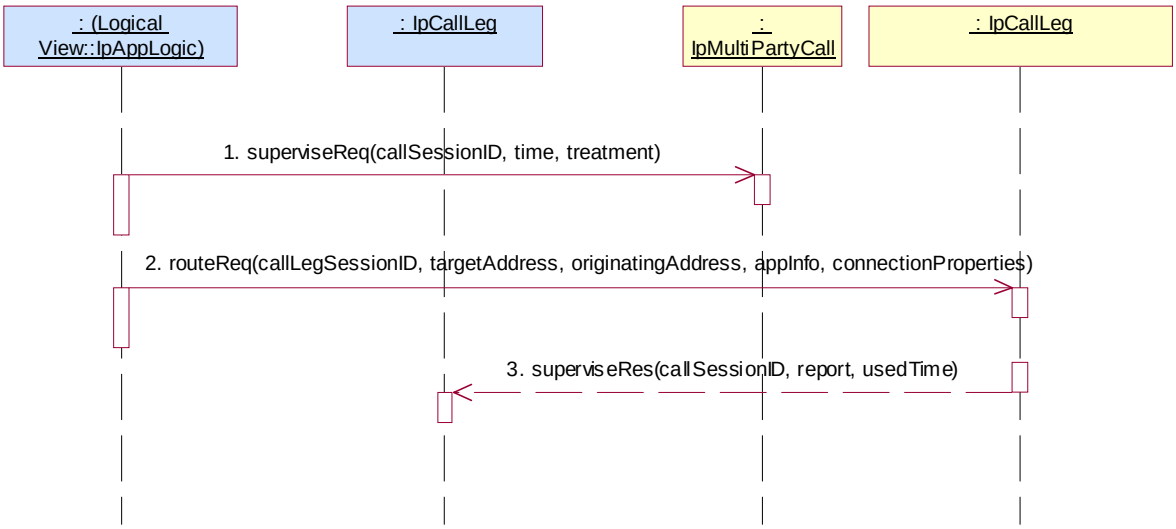
Condition: createCallLeg and superviseReq methods are supported.

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

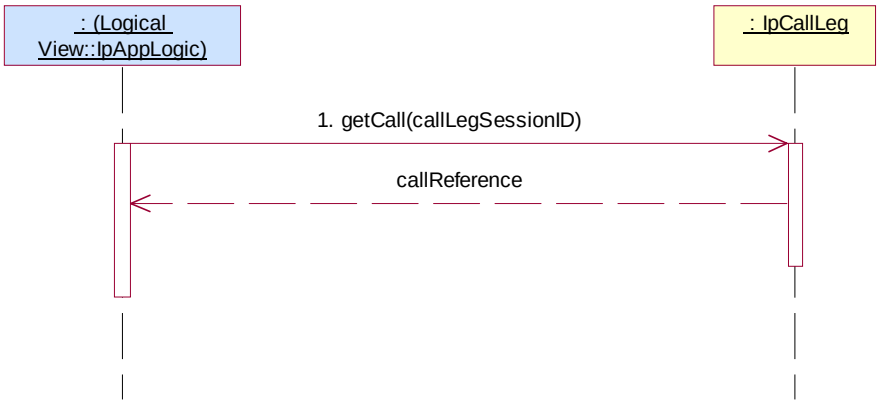
Reference: ES 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

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 202 915-4-3 [3], clause 6.5

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 is returned

Test MPCC_ IpCallLeg _21

Summary: IpCallLeg, attachMediaReq, P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.5

Preamble: Same as MPCC_ IpMultiPartyCall _25

Condition: attachMediaReq method is supported.

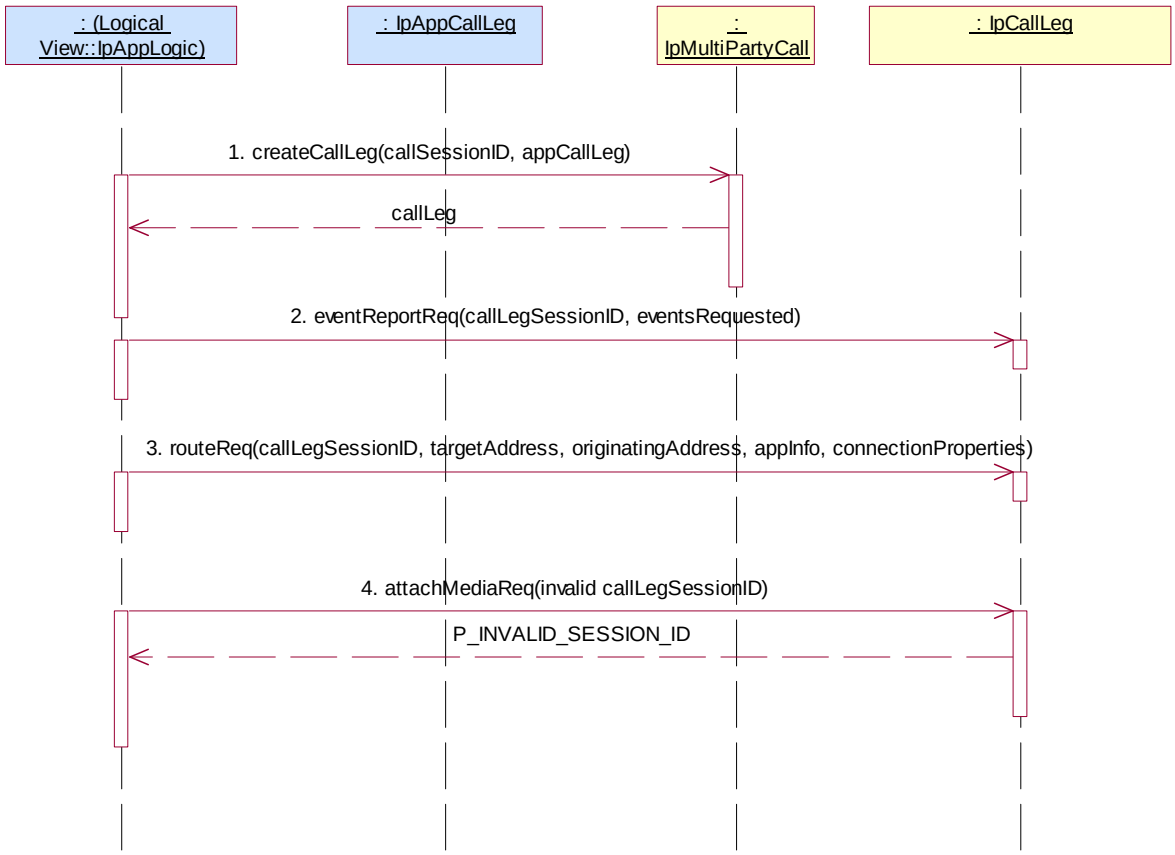
Test Sequence:

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 explicit media management
   Check: no exception is returned

4. Method call **attachMediaReq()**
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID is returned



Test MPCC_ IpCallLeg _22

Summary: IpCallLeg, detachMediaReq, P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.5

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

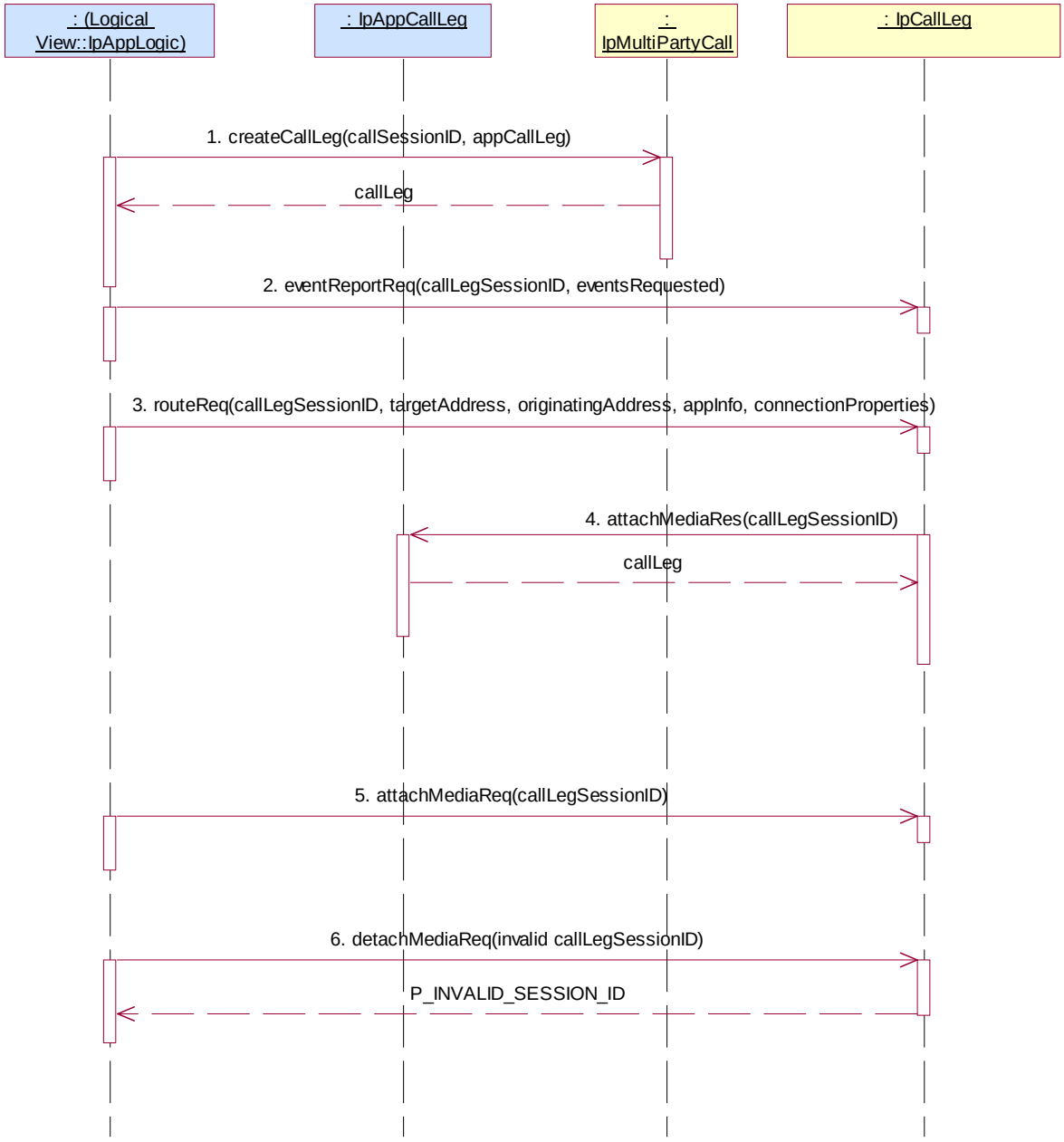
2. Method call **eventReportReq()**
   Parameters: valid callLegSessionID returned in 1., valid eventsRequested with Interrupt event
   Check: no exception is returned

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 is returned



Test MPCC_ IpCallLeg _23

Summary: IpCallLeg, getCurrentDestinationAddress, P_INVALID_SESSION_ID

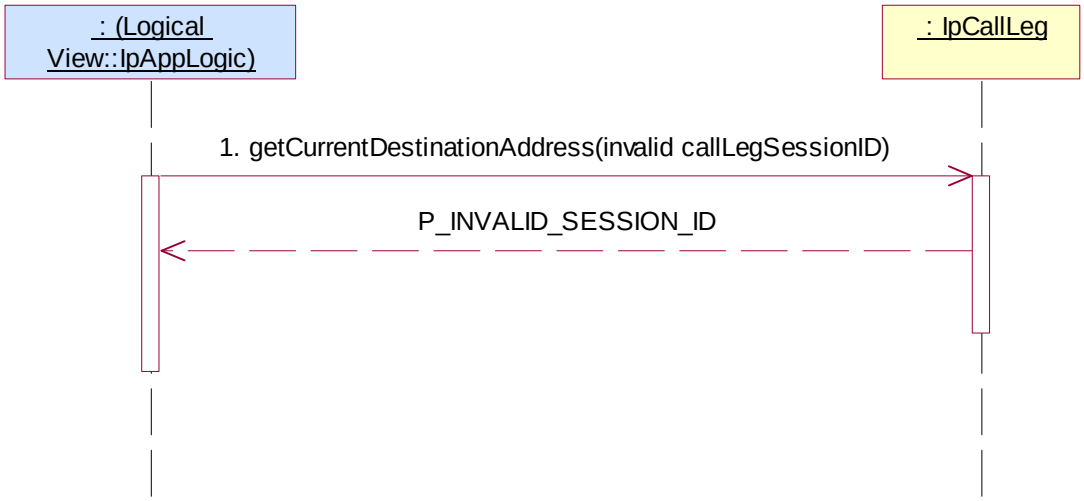
Reference: ES 202 915-4-3 [3], clause 6.5

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 is returned



Test MPCC_ IpCallLeg _24

Summary: IpCallLeg, setChargePlan, P_INVALID_SESSION_ID

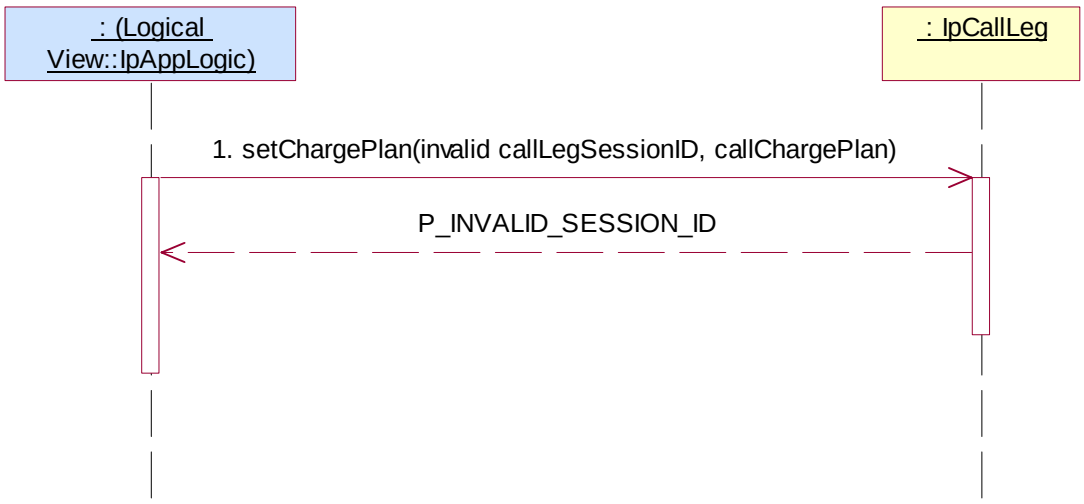
Reference: ES 202 915-4-3 [3], clause 6.5

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 is returned



Test MPCC_ IpCallLeg _25

Summary: IpCallLeg, setAdviceOfCharge, P_INVALID_SESSION_ID

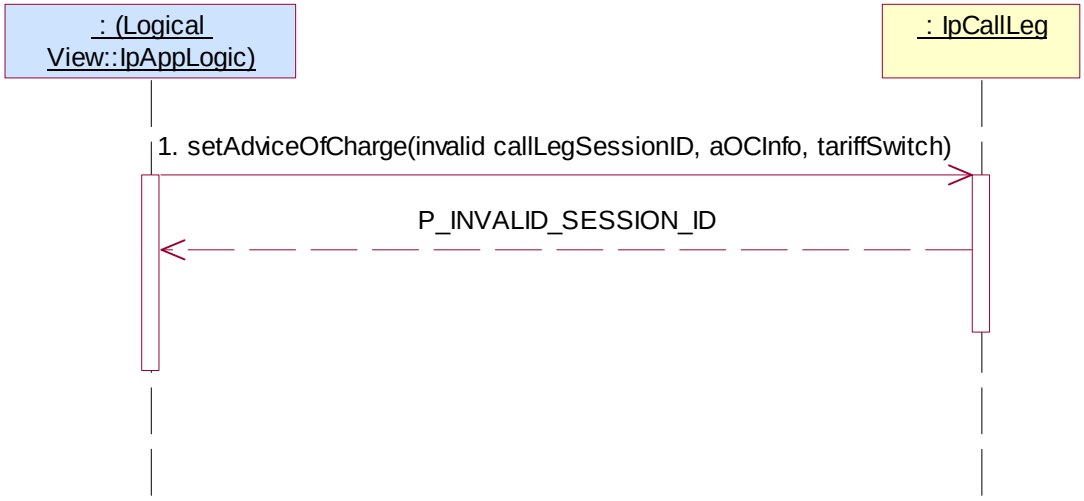
Reference: ES 202 915-4-3 [3], clause 6.5

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 is returned



Test MPCC_ IpCallLeg _26

Summary: IpCallLeg, setAdviceOfCharge, P_INVALID_CURRENCY

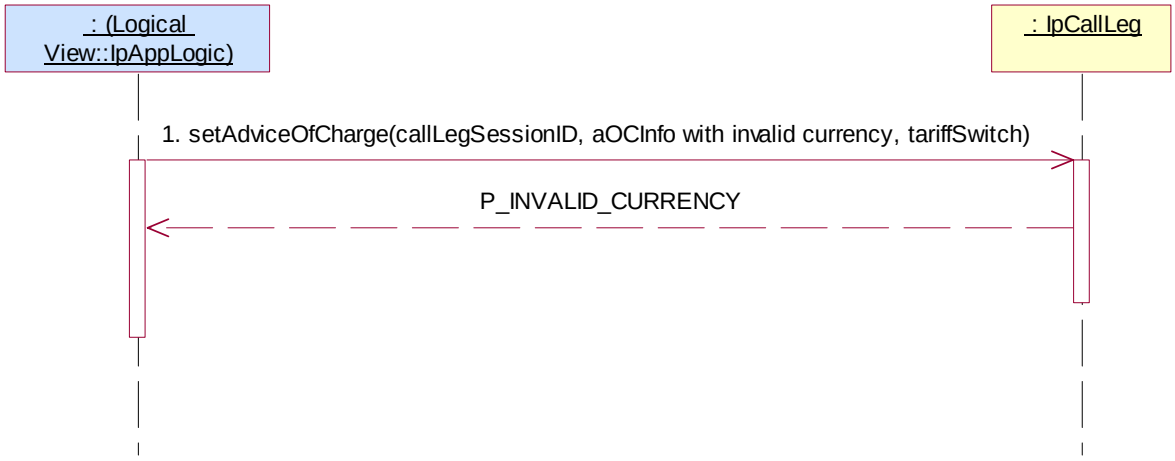
Reference: ES 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

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 is returned



Test MPCC_ IpCallLeg _27

Summary: IpCallLeg, setAdviceOfCharge, P_INVALID_AMOUNT

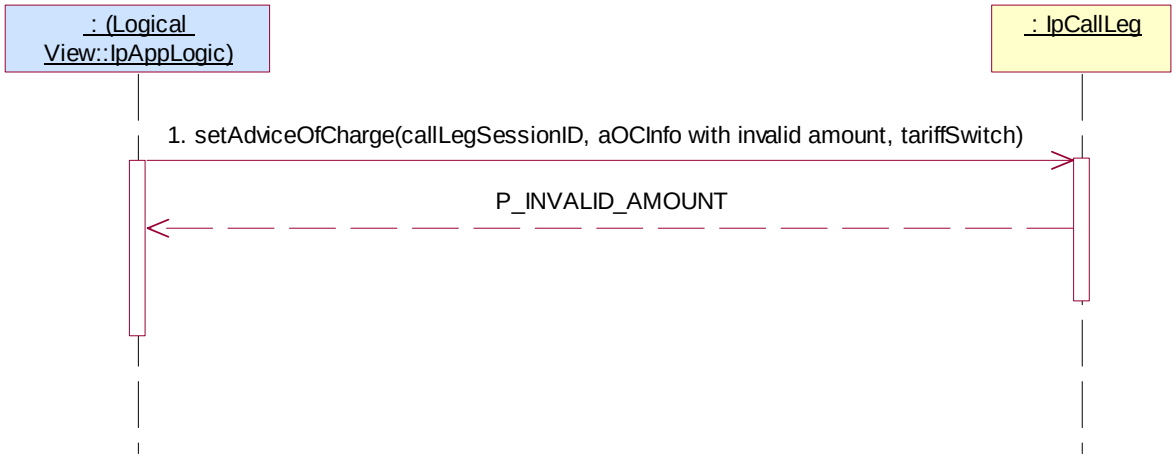
Reference: ES 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

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 is returned



Test MPCC_ IpCallLeg _28

Summary: IpCallLeg, superviseReq, P_INVALID_SESSION_ID

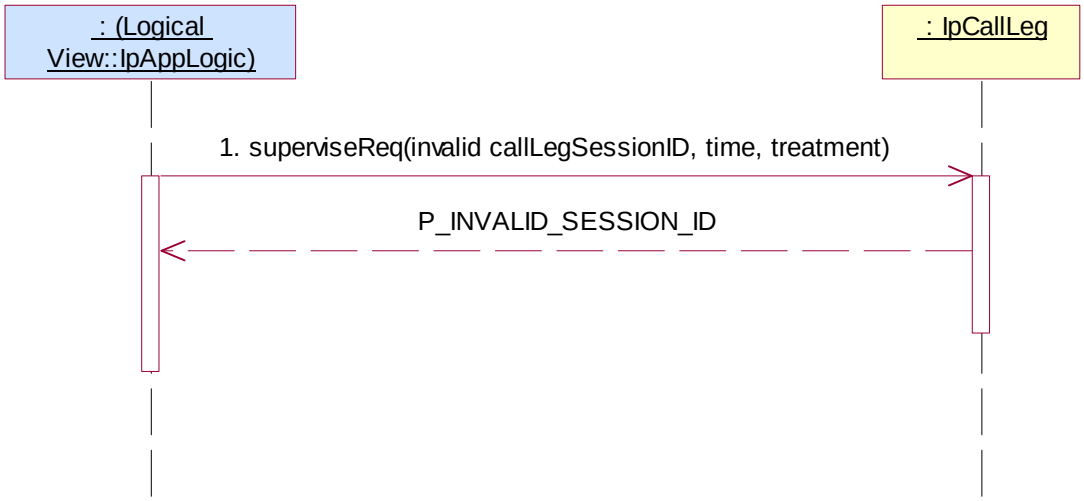
Reference: ES 202 915-4-3 [3], clause 6.5

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 is returned



Test MPCC_ IpCallLeg _29

Summary: IpCallLeg, getCall**,** P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.5

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 is returned

5.2.3 MultiMedia Call Control Service (MMCC)

The TPs in this clause are based on ES 202 915-4-4 [4].

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 202 915-4-3 [3], clause 6.1

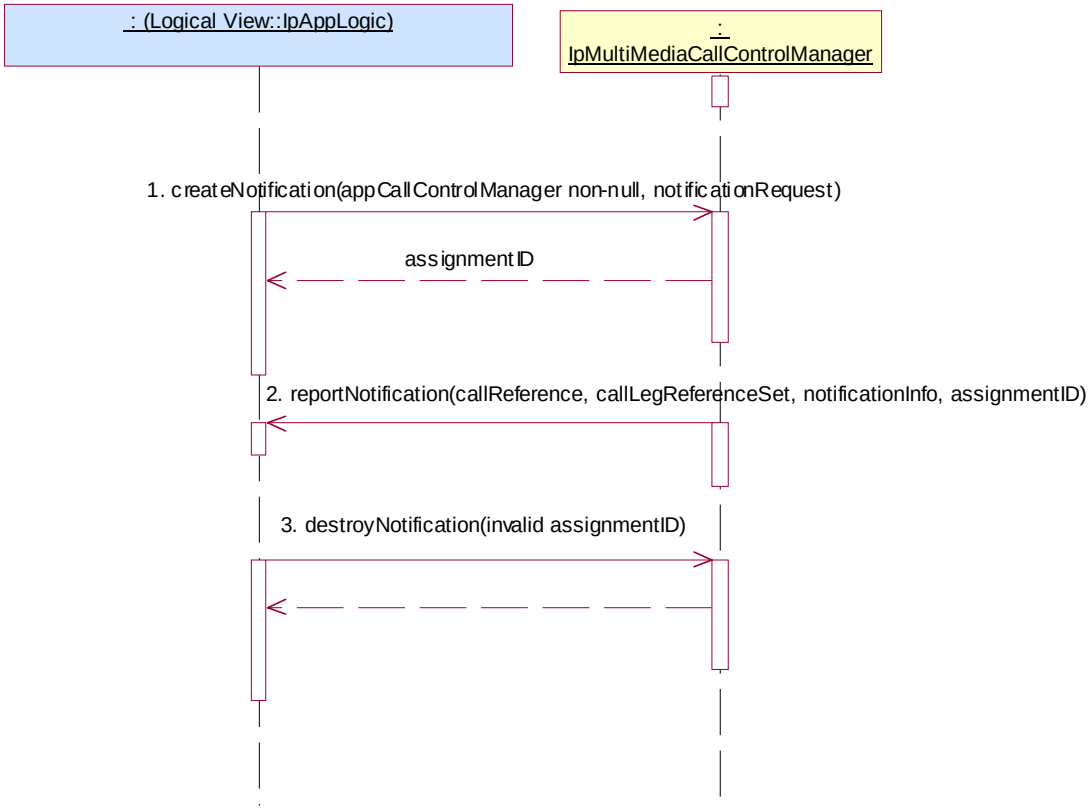
Condition: createNotification method is supported.

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 202 915-4-3 [3], clause 6.1

Condition: createNotification method is supported.

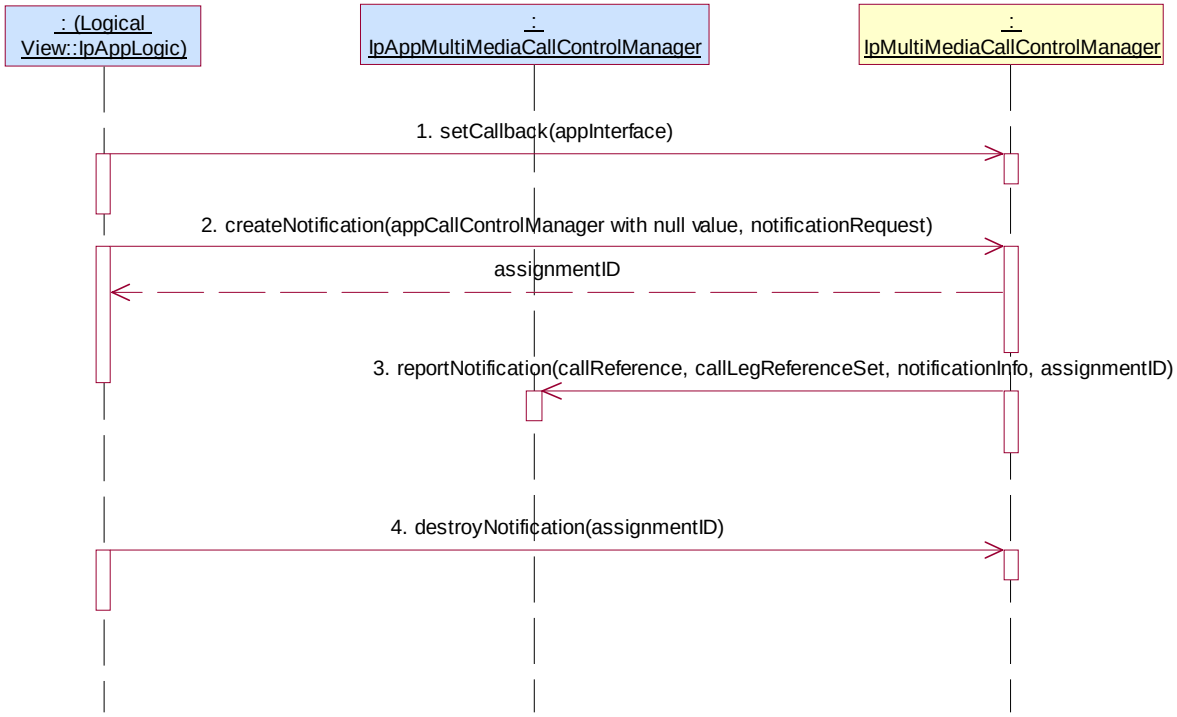
Test Sequence:

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

2. Method call **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 202 915-4-3 [3], clause 6.1

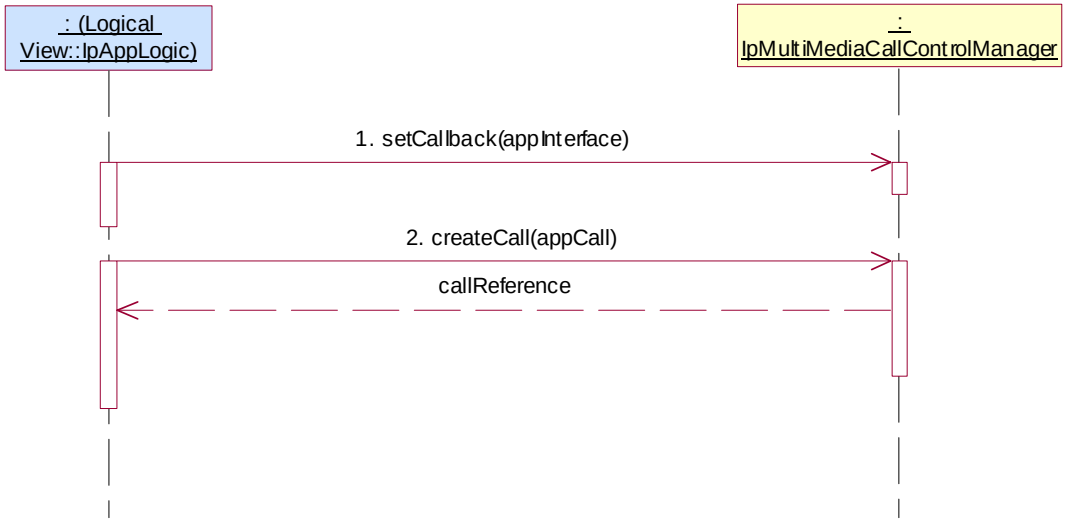
Preamble: Application has a reference interface used for callbacks.

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 mandatory methods, successful

Reference: ES 202 915-4-3 [3], clause 6.1

Condition: enableNotifications method is supported.

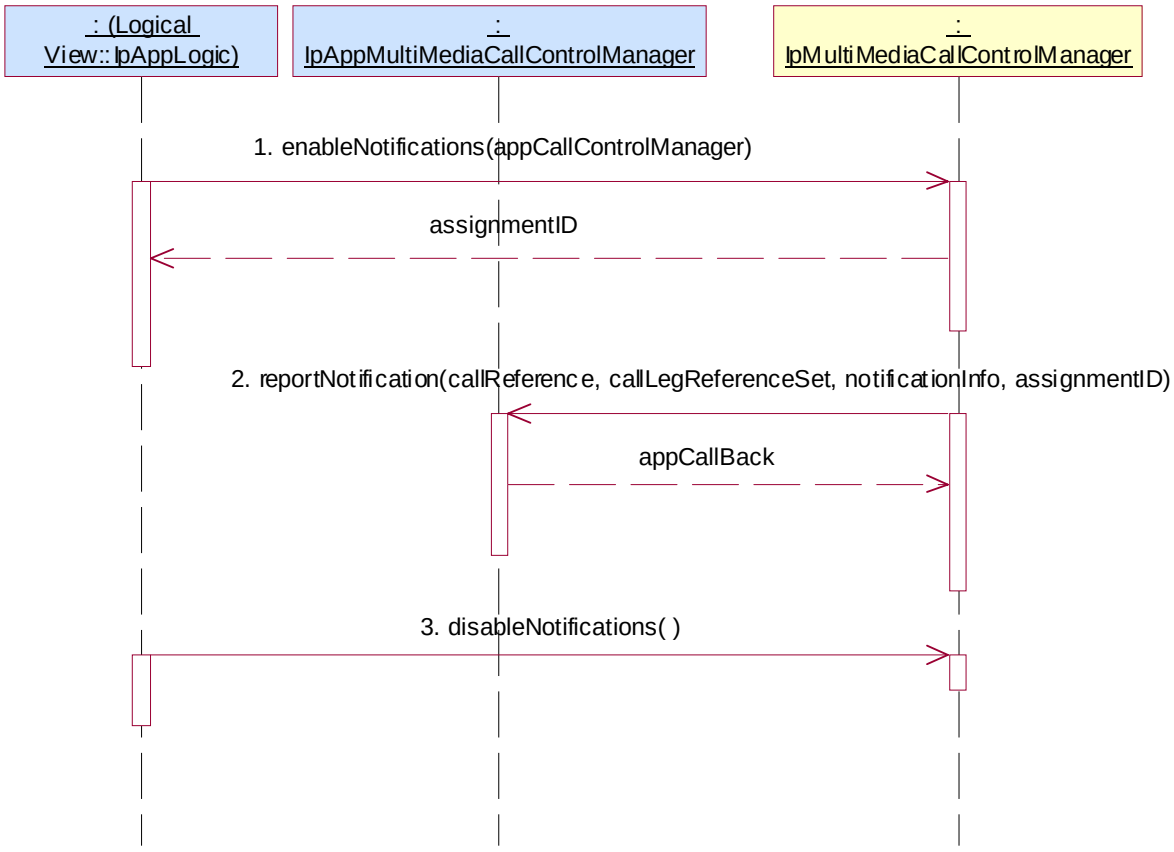
Test Sequence:

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

2. Method call **enableNotifications()**
   Parameters: appCallControlManager with null value
   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 **disableNotifications()**
   Parameters: none
   Check: no exception is returned



Test MMCC_ IpMultiMediaCallControlManager _05

Summary: IpMultiMediaCallControlManager, all methods mandatory, successful

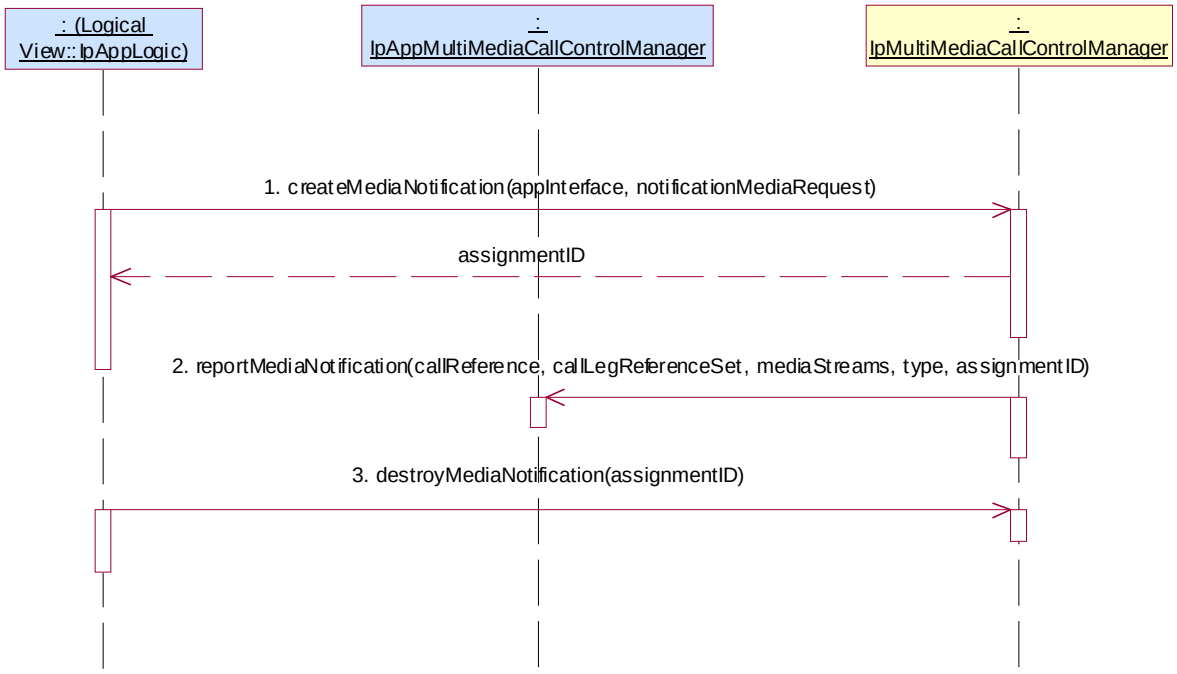
Reference: ES 202 915-4-3 [3], clause 6.1 and ES 202 915-4-4 [4], clause 6.1

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 _06

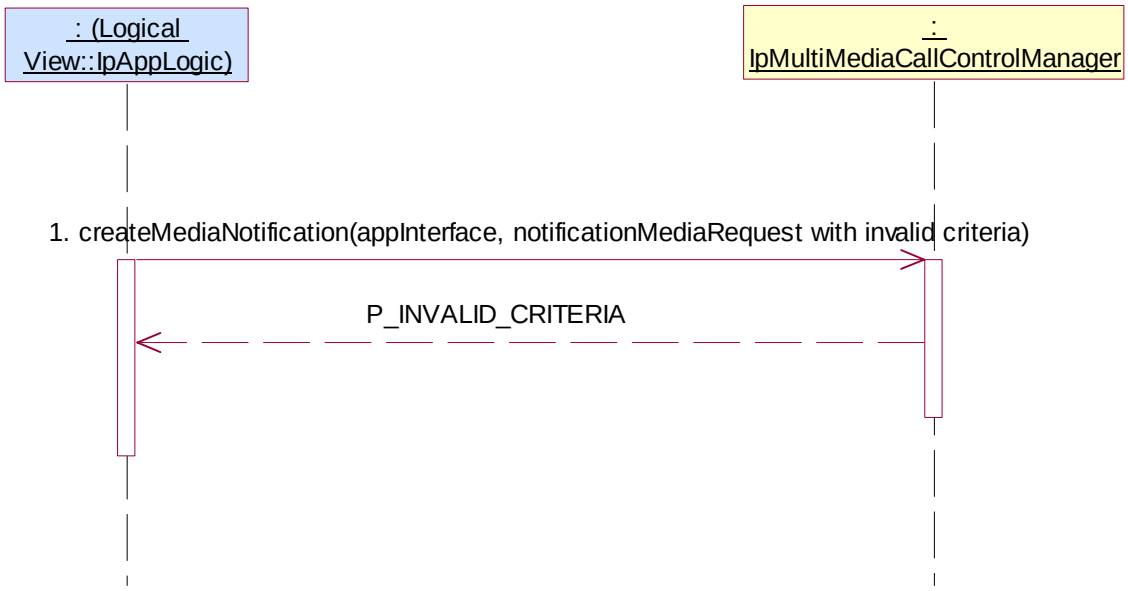
Summary: IpMultiMediaCallControlManager, createMediaNotification, P_INVALID_CRITERIA

Reference: ES 202 915-4-4 [4], clause 6.1

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 is returned



Test MMCC_ IpMultiMediaCallControlManager _07

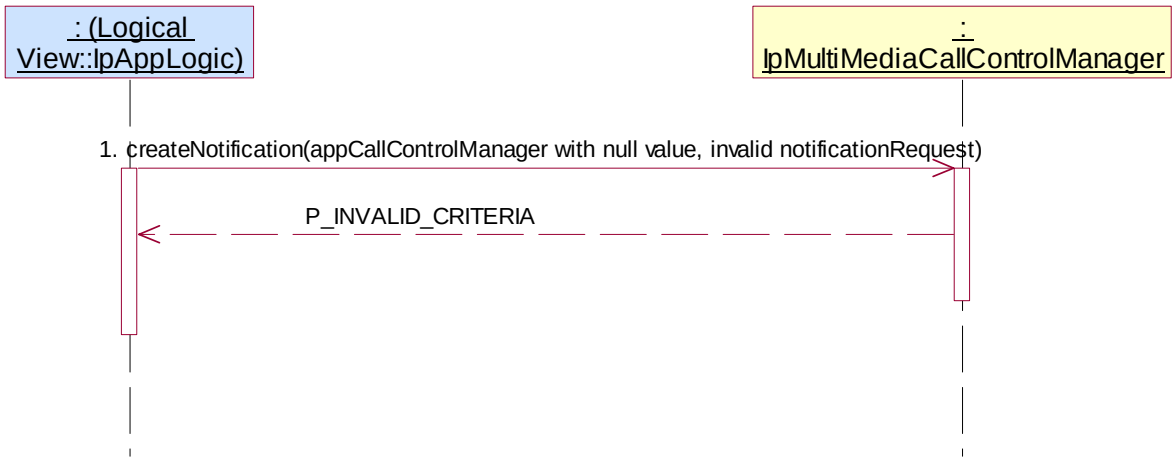
Summary: IpMultiMediaCallControlManager, createNotification, P_INVALID_CRITERIA

Reference: ES 202 915-4-3 [3], clause 6.1

Condition: createNotification method is supported.

Test Sequence:

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



Test MMCC_ IpMultiMediaCallControlManager _08

Summary: IpMultiMediaCallControlManager, destroyNotification, P_INVALID_ASSIGNMENT_ID

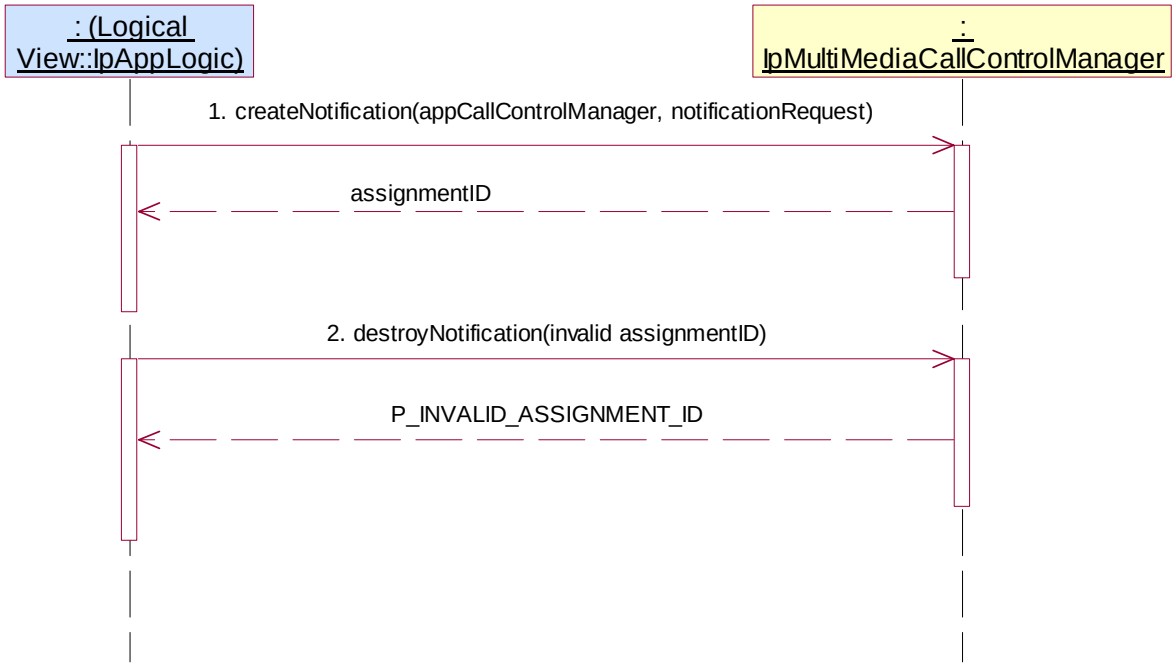
Reference: ES 202 915-4-3 [3], clause 6.1

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 is returned



Test MMCC_ IpMultiMediaCallControlManager _09

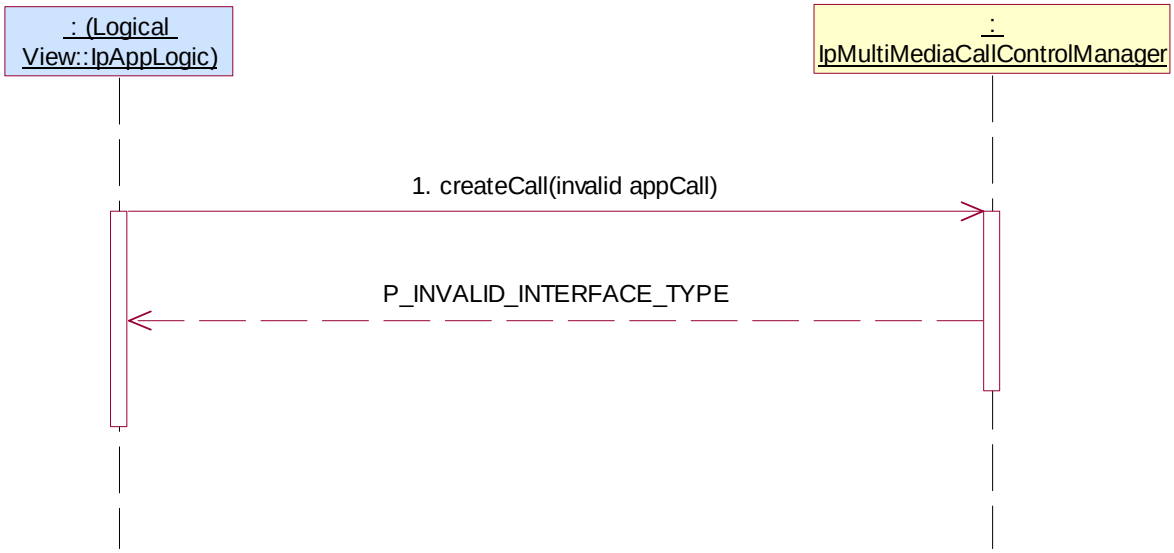
Summary: IpMultiMediaCallControlManager, createCall, P_INVALID_INTERFACE_TYPE

Reference: ES 202 915-4-3 [3], clause 6.1

Condition: createCall method is supported.

Test Sequence:

1. Method call **createCall()**
   Parameters: invalid value of appCall
   Check: P_INVALID_INTERFACE_TYPE is returned



5.2.3.1.3 Optional, valid behaviour

Test MMCC_ IpMultiMediaCallControlManager _10

Summary: IpMultiMediaCallControlManager, getNextNotification, successful

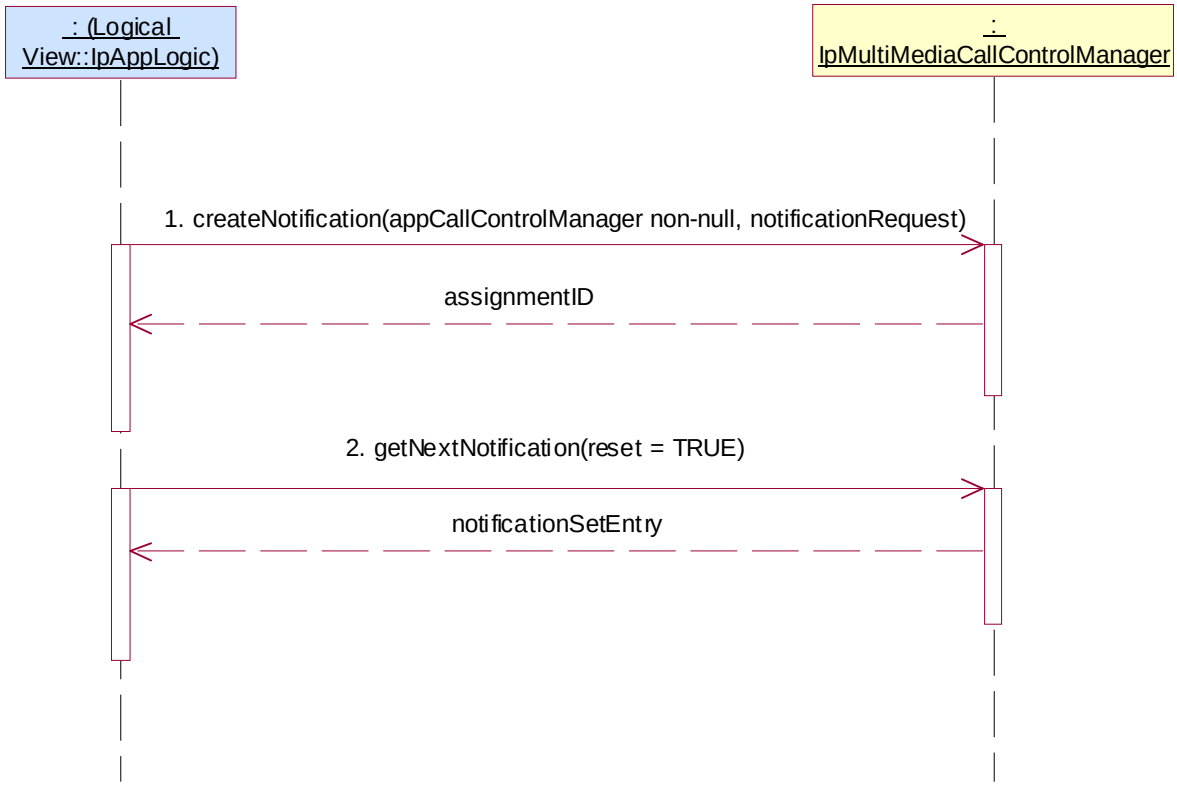
Reference: ES 202 915-4-3 [3], clause 6.1

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 **getNextNotification()**
   Parameters: reset = TRUE
   Check: valid value of TpNotificationRequestedSetEntry is returned where notificationRequest given in 1. is included as a value of this TpCallEventCriteriaResult



Test MMCC_ IpMultiMediaCallControlManager _11

Summary: IpMultiMediaCallControlManager, getMediaNotification, successful

Reference: ES 202 915-4-4 [4], clause 6.1

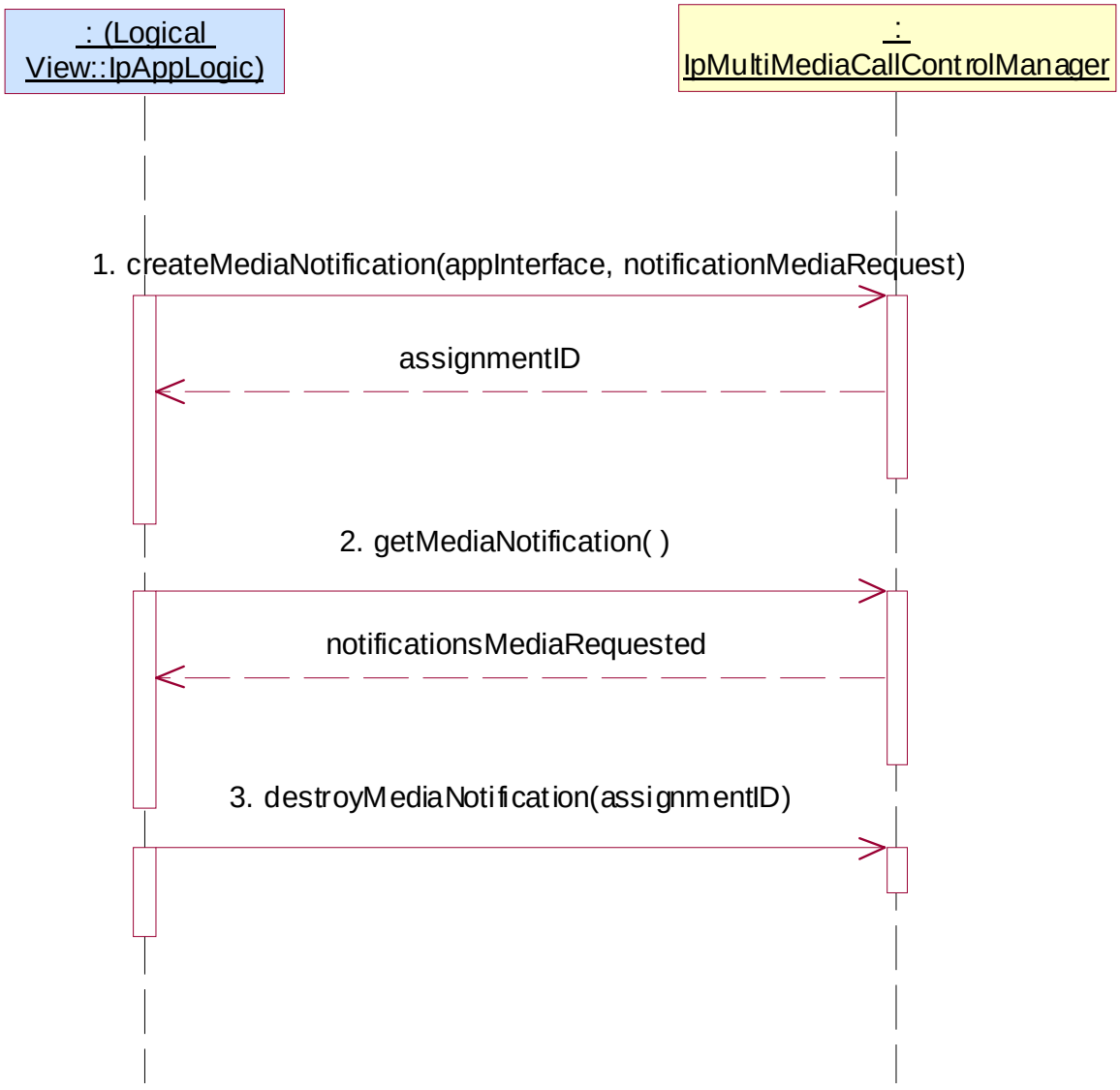
Condition: getMediaNotification method is supported.

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 _12

Summary: IpMultiMediaCallControlManager, changeMediaNotification, successful

Reference: ES 202 915-4-4 [4], clause 6.1

Condition: getMediaNotification and changeMediaNotification methods are supported.

Test Sequence:

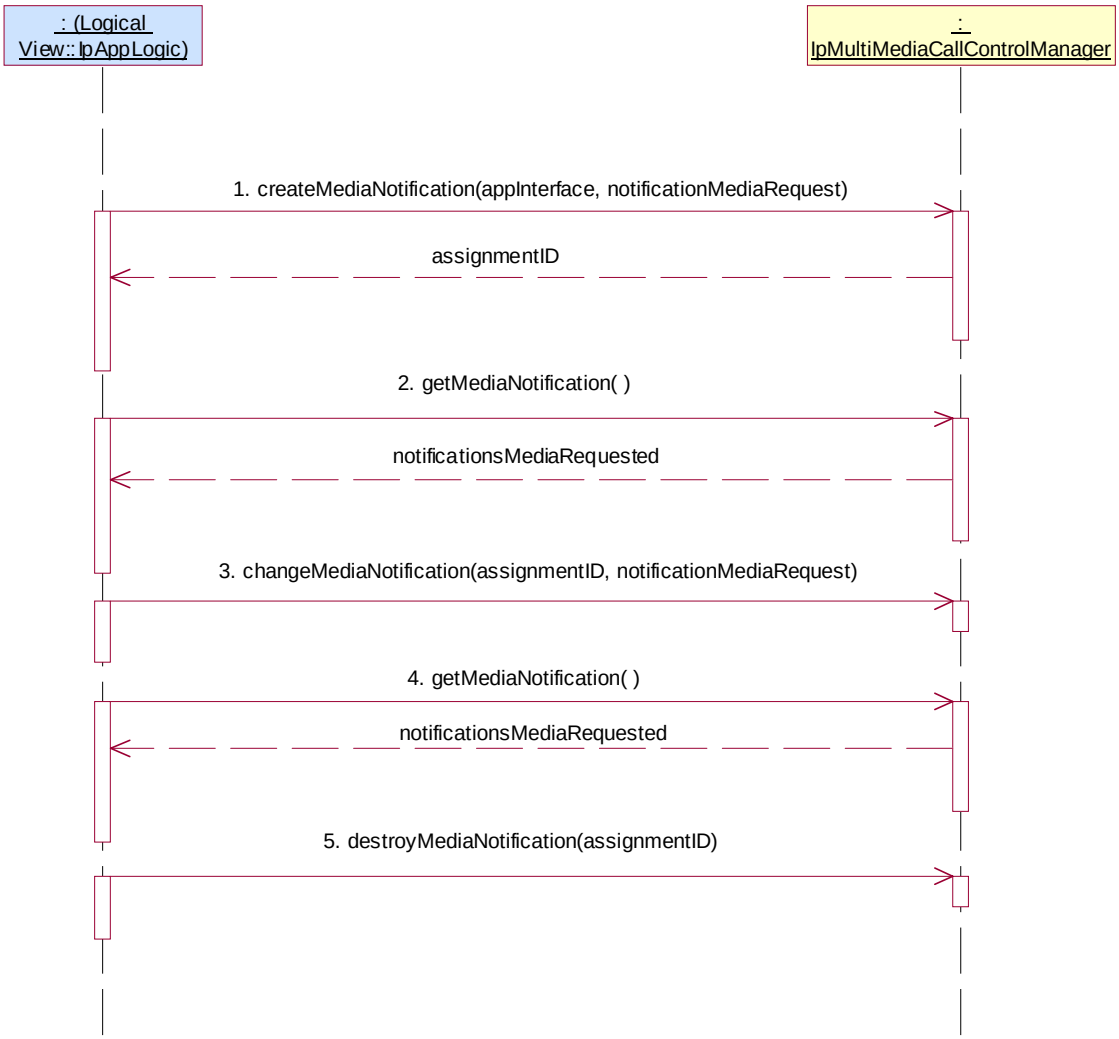
1. Method call **createMediaNotification()**
   Parameters: valid appInterface, valid notificationMediaRequest
   Check: valid value of TpAssignmentID is returned

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



Test MMCC_ IpMultiMediaCallControlManager _13

Summary: IpMultiMediaCallControlManager, changeNotification, successful

Reference: ES 202 915-4-3 [3], clause 6.1

Condition: createNotification, getNextNotification and changeNotification methods are supported.

Test Sequence:

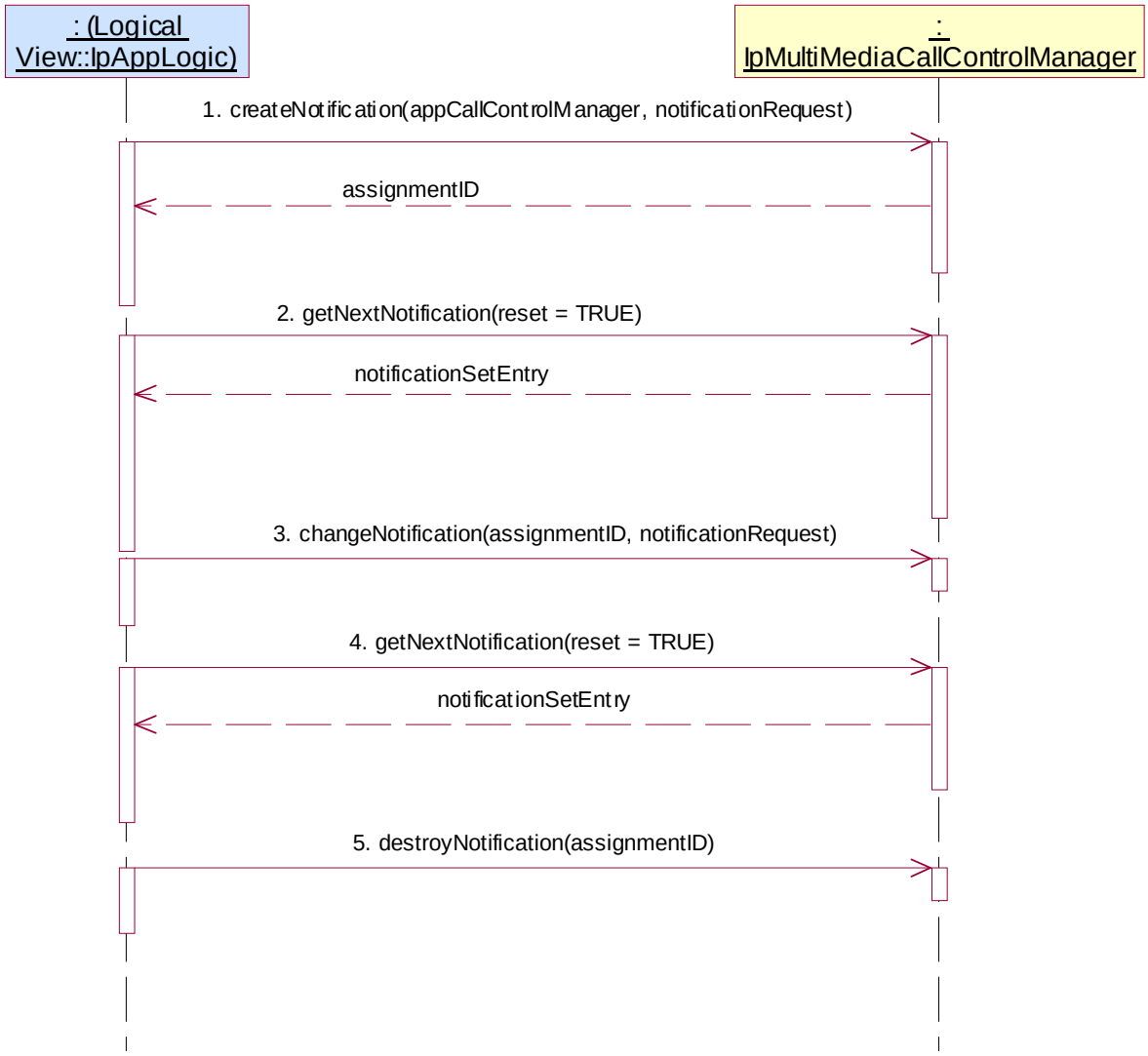
1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, non-null, value, valid notificationRequest
   Check: valid value of TpAssignmentID is returned

2. Method call **getNextNotification()**
   Parameters: reset =TRUE
   Check: valid value of TpNotificationRequestedSetEntry 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 **getNextNotification()**
   Parameters: reset =TRUE
   Check: valid value of TpNotificationRequestedSetEntry is returned where notificationRequest given in 3. is included as a value of this TpCallEventCriteriaResult

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



Test MMCC_ IpMultiMediaCallControlManager _14

Summary: IpMultiMediaCallControlManager, all methods, successful

Reference: ES 202 915-4-3 [3], clause 6.1

Condition: createNotification, callOverLoadEncountered and callOverLoadCeased methods are supported.

Test Sequence:

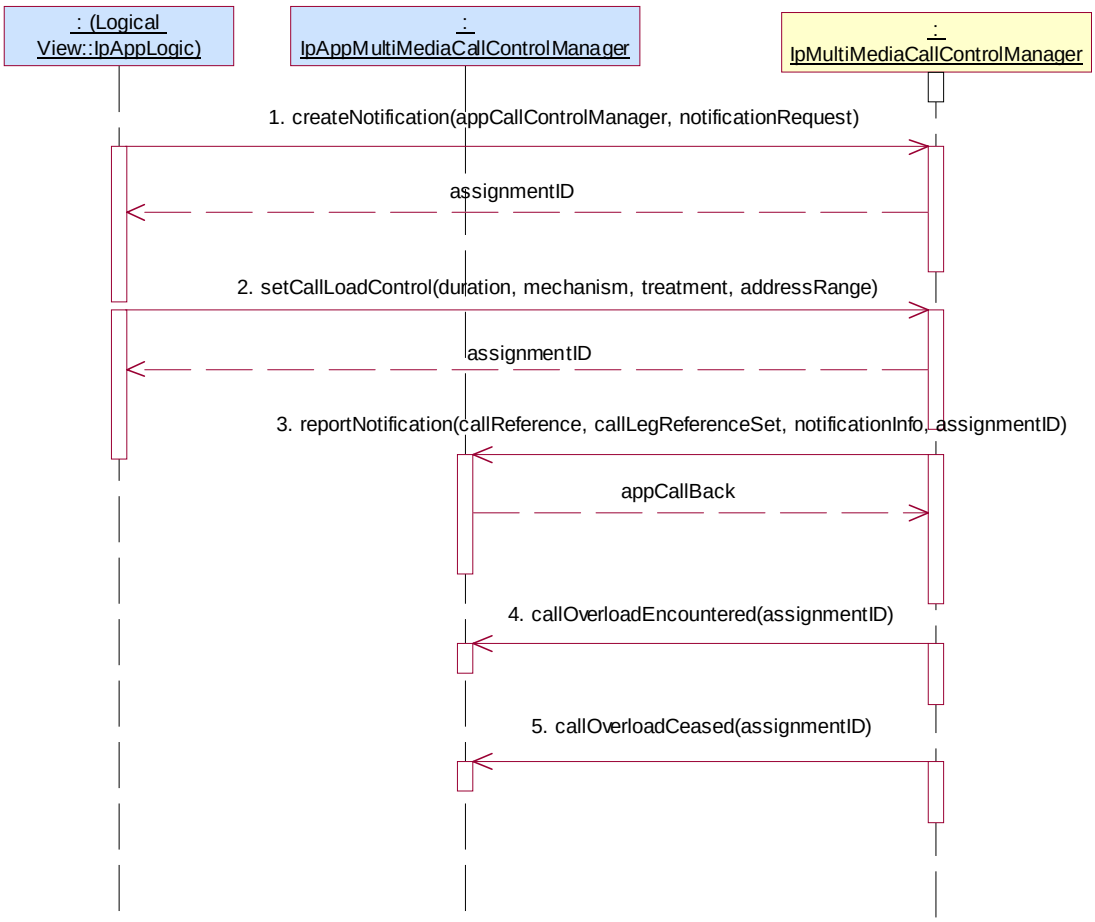
1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, non-null, value, valid notificationRequest
   Check: valid value of TpAssignmentID is returned

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.



Test MMCC_ IpMultiMediaCallControlManager _15

Summary: IpMultiMediaCallControlManager, all methods, successful

Reference: ES 202 915-4-3 [3], clause 6.1

Condition: createNotification, setCallLoadControl methods are supported.

Test Sequence:

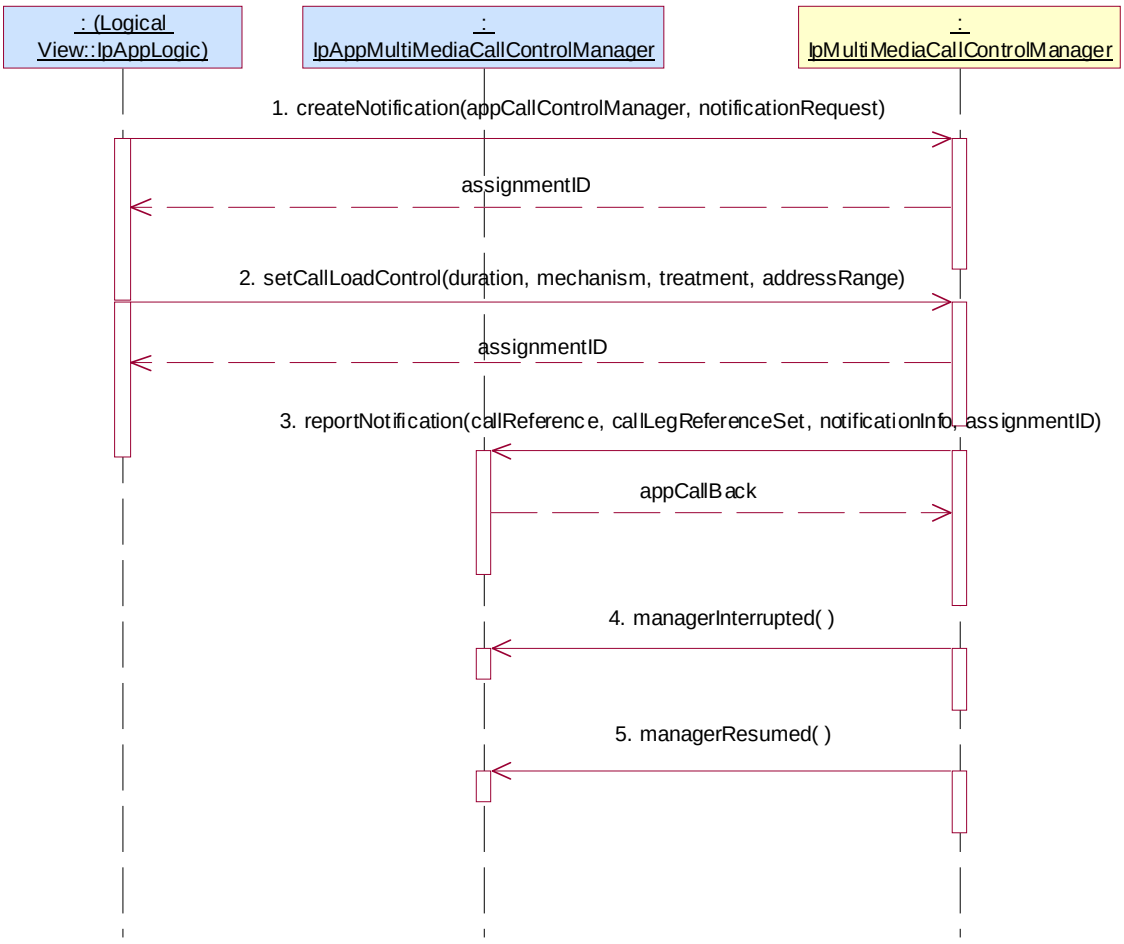
1. Method call **createNotification()**
   Parameters: appCallControlManager with valid, not null, value, valid eventCriteria
   Check: valid value of TpAssignmentID is returned

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 _16

Summary: IpMultiMediaCallControlManager, all methods, successful

Reference: ES 202 915-4-3 [3], clause 6.1

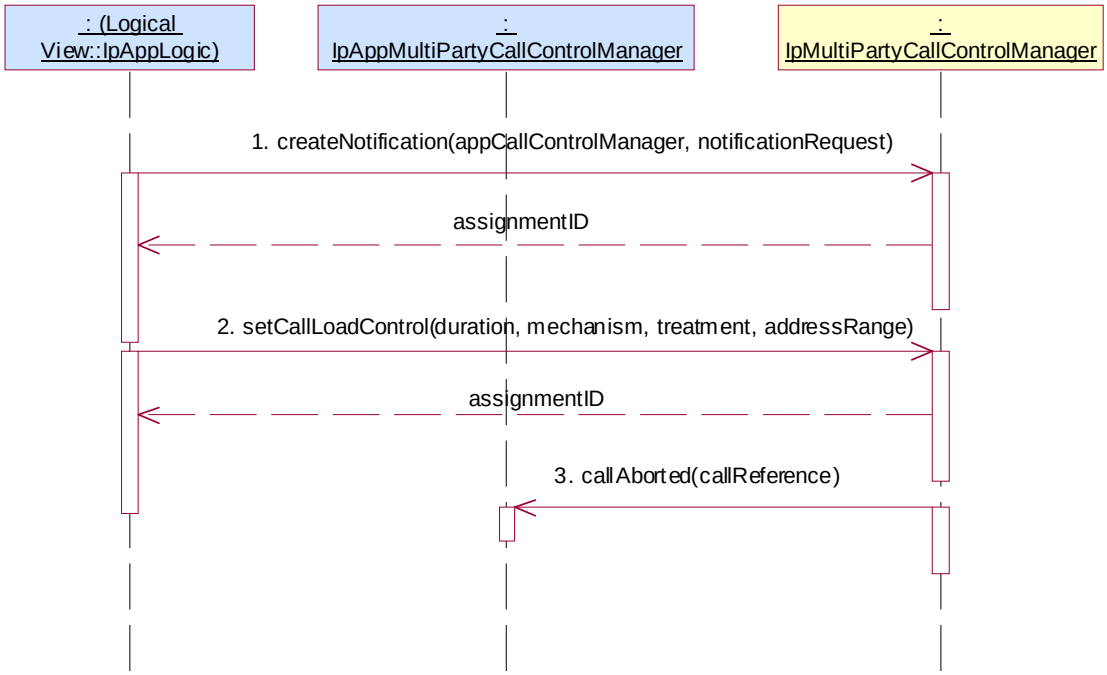
Condition: createNotification and callAborted methods are supported.

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) **IpAppMultiMediaCallControlManager** interface.

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



5.2.3.1.4 Optional, invalid behaviour

Test MMCC_ IpMultiMediaCallControlManager _17

Summary: IpMultiMediaCallControlManager, changeMediaNotification, P_INVALID_CRITERIA

Reference: ES 202 915-4-4 [4], clause 6.1

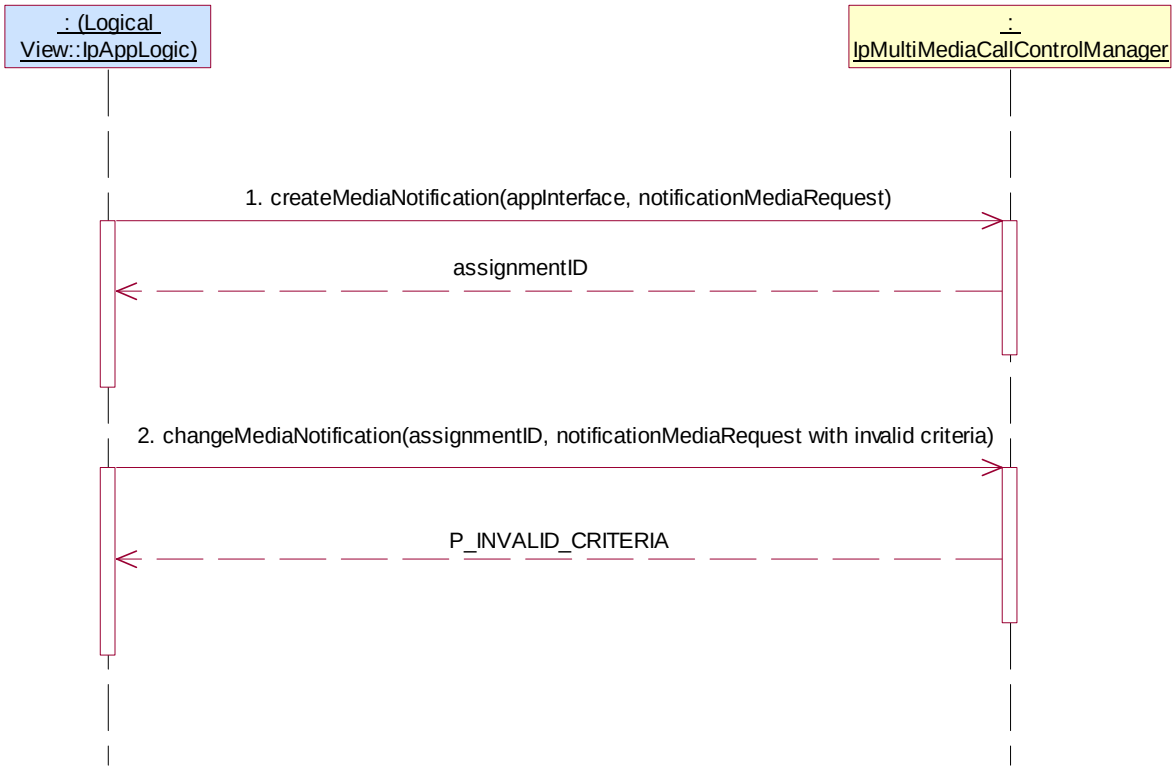
Preamble: Application has a reference interface used for callbacks.

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 is returned



Test MMCC_ IpMultiMediaCallControlManager _18

Summary: IpMultiMediaCallControlManager, changeMediaNotification, P_INVALID_ASSIGNMENT_ID

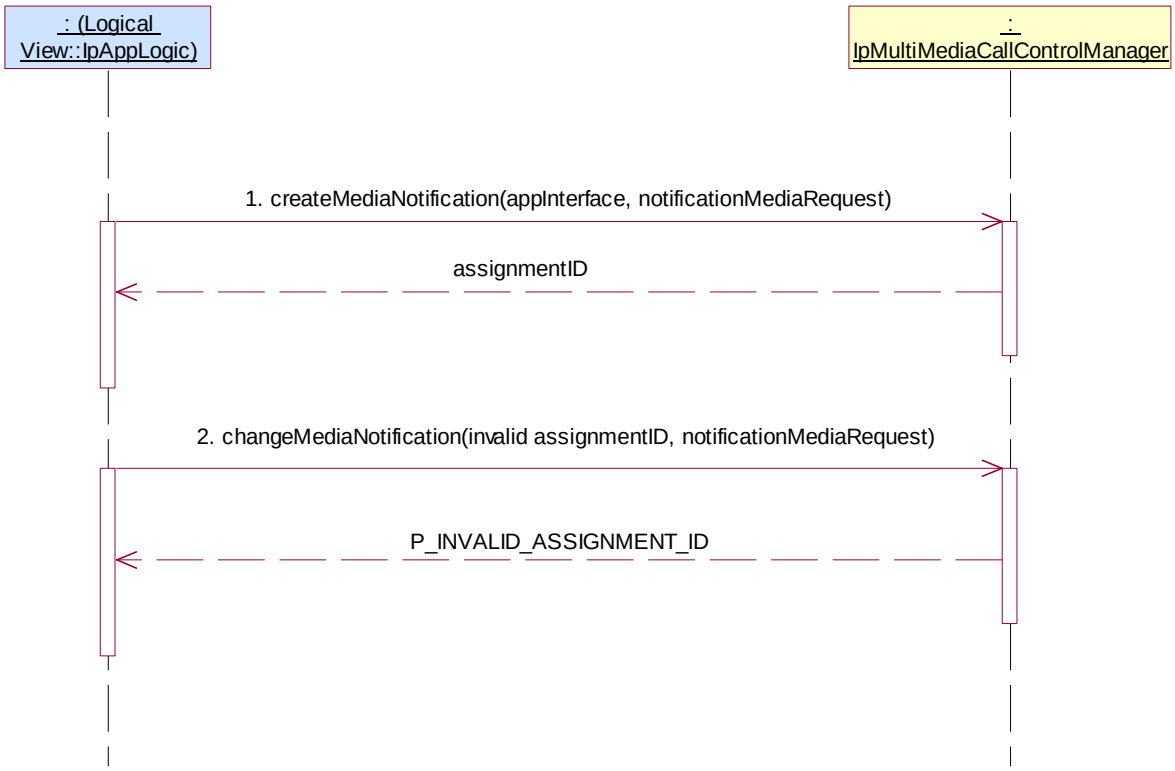
Reference: ES 202 915-4-4 [4], clause 6.1

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 is returned



Test MMCC_ IpMultiMediaCallControlManager _19

Summary: IpMultiMediaCallControlManager, changeNotification, P_INVALID_ASSIGNMENT_ID

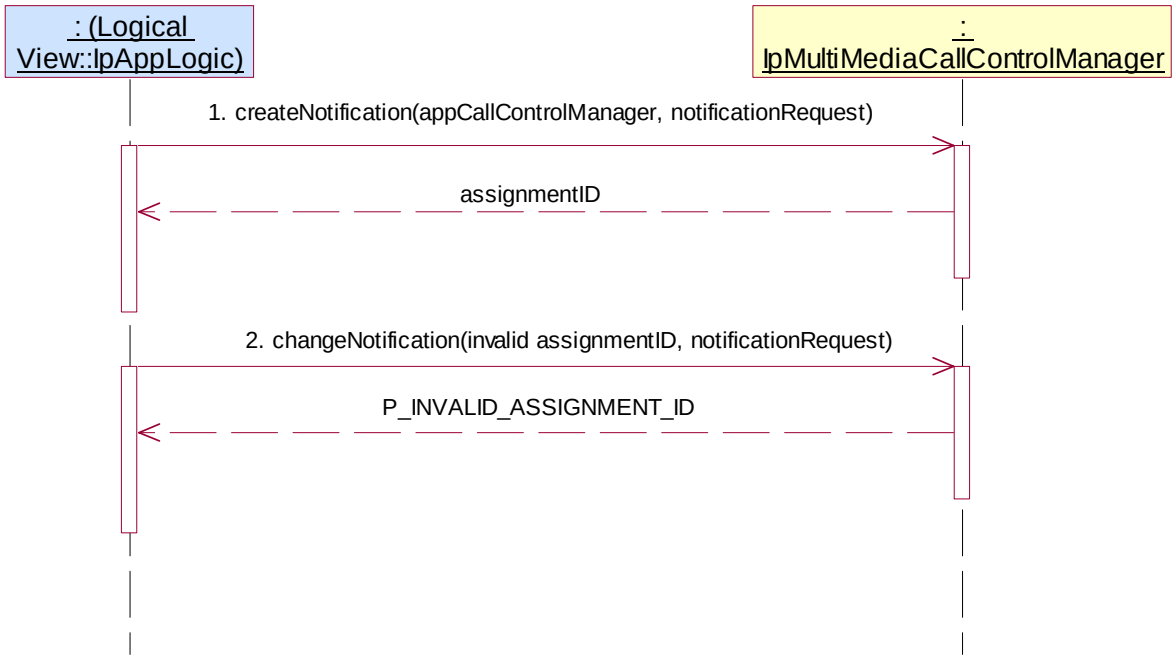
Reference: ES 202 915-4-3 [3], clause 6.1

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 is returned



Test MMCC_ IpMultiMediaCallControlManager _20

Summary: IpMultiMediaCallControlManager, changeNotification, P_INVALID_CRITERIA

Reference: ES 202 915-4-3 [3], clause 6.1

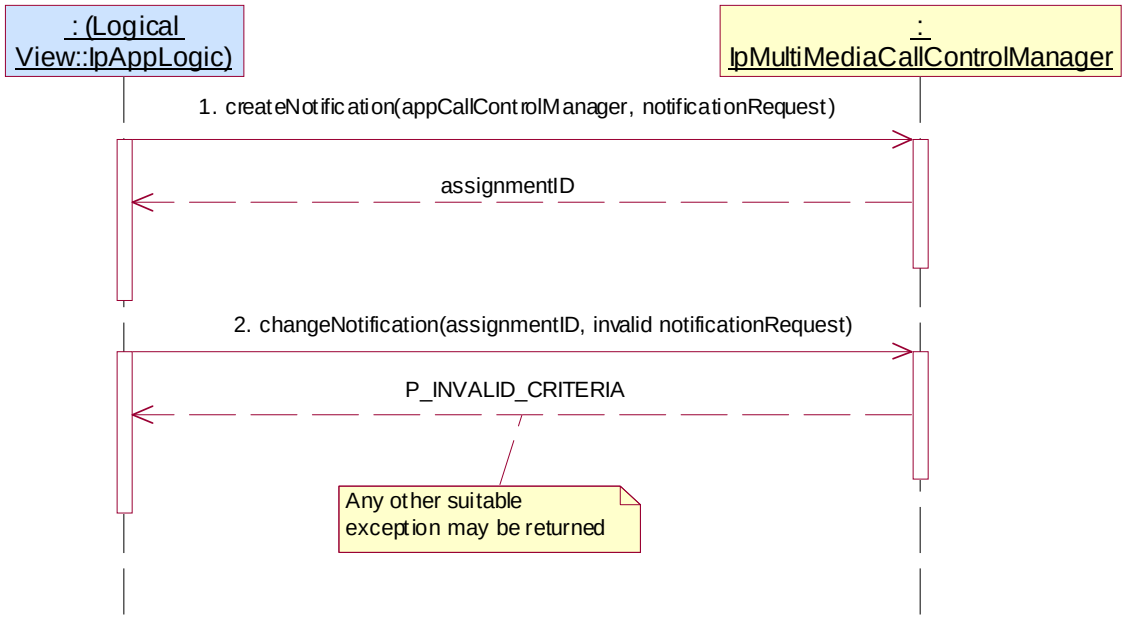
Preamble: Application has a reference interface used for callbacks.

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 is returned



Test MMCC_ IpMultiMediaCallControlManager _21

Summary: IpMultiMediaCallControlManager, setCallLoadControl, P_INVALID_ADDRESS

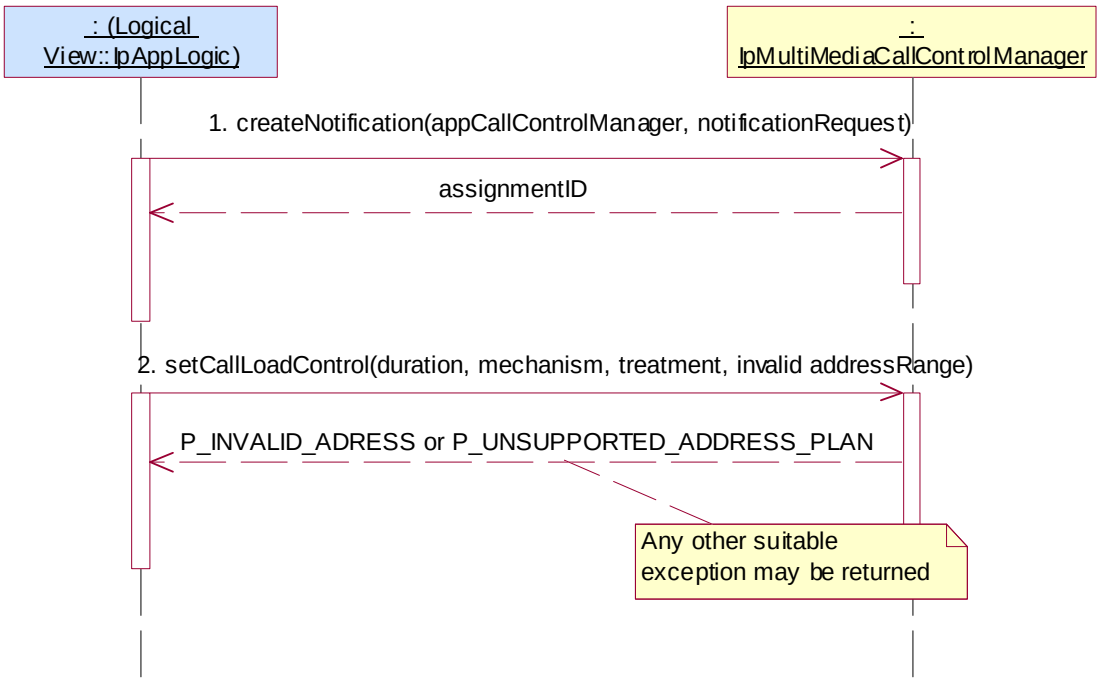
Reference: ES 202 915-4-3 [3], clause 6.1

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

Test MMCC_ IpMultiMediaCall _01

Summary: IpMultiMediaCall, all methods mandatory, successful

Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.3

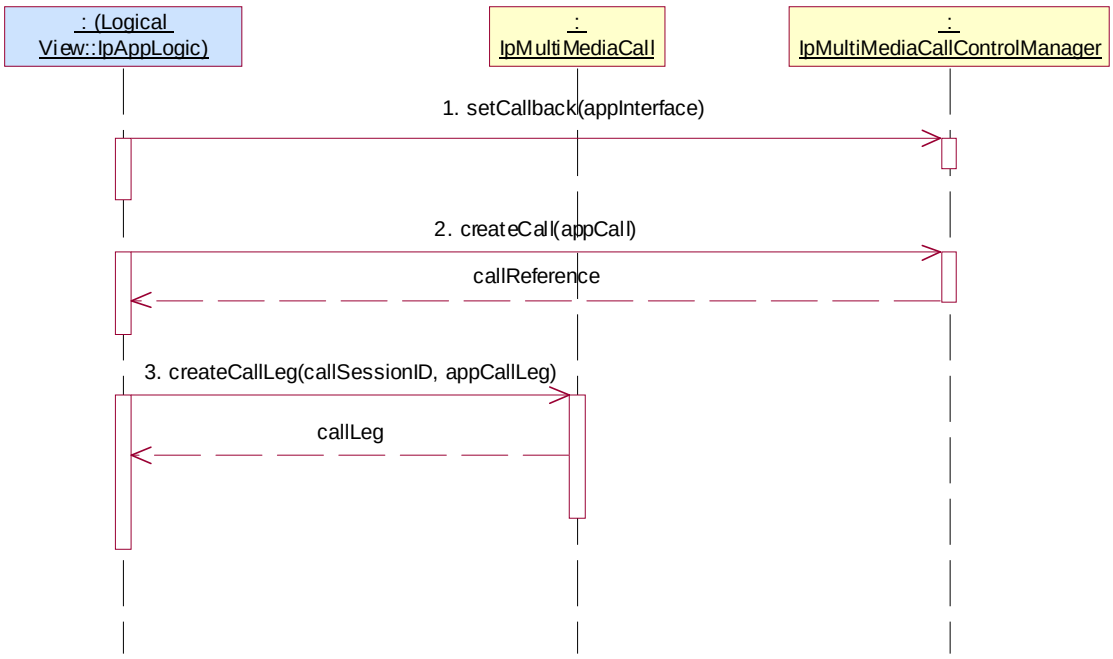
Condition: createCall, createCallLeg methods are supported.

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 202 915-4-3 [3], clauses 6.1 and 6.3

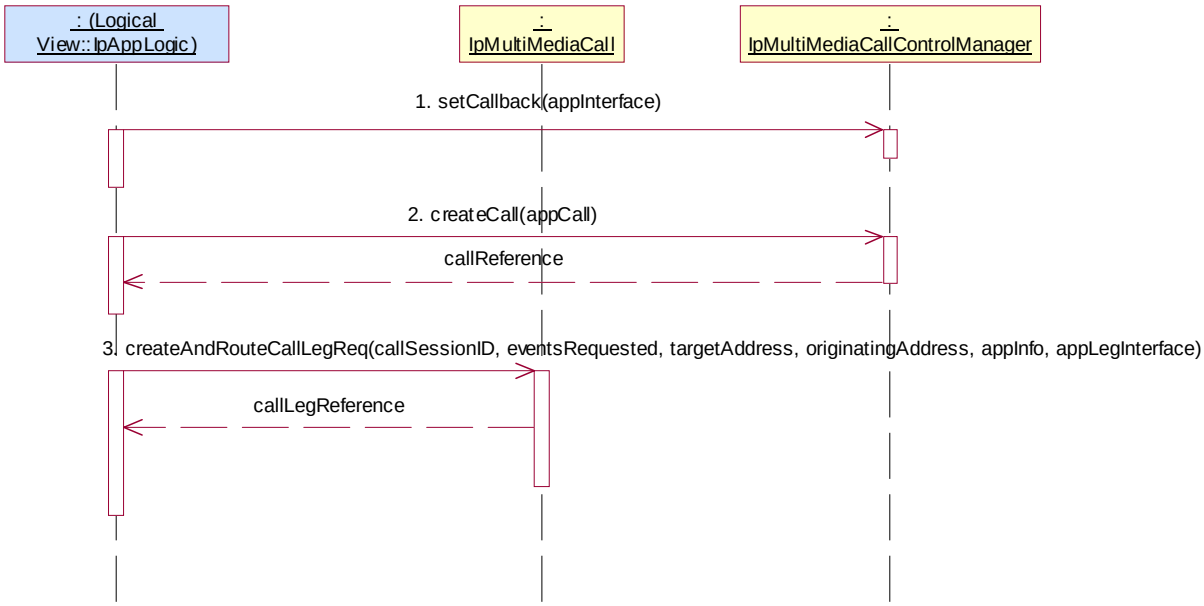
Condition: createCall, createAndRouteCallLeg method is supported.

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 202 915-4-3 [3], clauses 6.1 and 6.3 and ES 202 915-4-4 [4], clause 6.1

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 **eventReportReq()**
   Parameters: valid callLegSessionID returned in 1., valid eventsRequested with an Interrupt event
   Check: no exception 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

3. Method call **createAndRouteCallLegReq()** on IpMultiMediaCall
   Parameters: valid callSessionID returned in 2., valid eventsRequested with Interrupt event, 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 **eventReportReq()**
   Parameters: valid callLegSessionID returned in 1., valid eventsRequested with an Interrupt event
   Check: no exception 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

3. Method call **createAndRouteCallLegReq()** on IpMultiMediaCall
   Parameters: valid callSessionID reported in 2., valid eventsRequested with Interrupt event, 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 **eventReportReq()**
   Parameters: valid callLegSessionID returned in 1., valid eventsRequested with an Interrupt event
   Check: no exception 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

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

or

1. Method call **enableNotifications()**
   Parameters: appCallControlManager with valid, non-null, value
   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

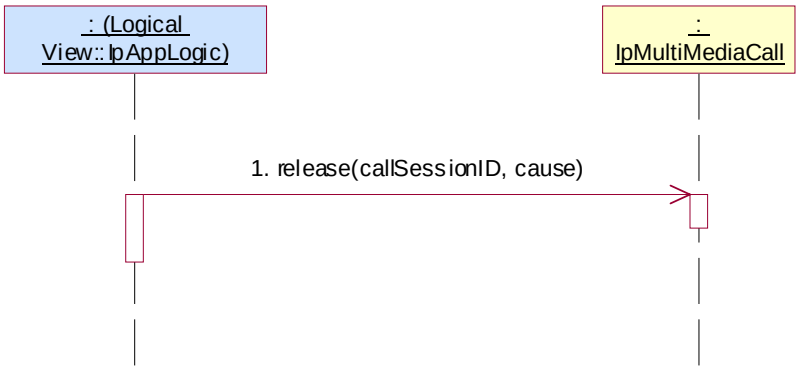
4. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 3., valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

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

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

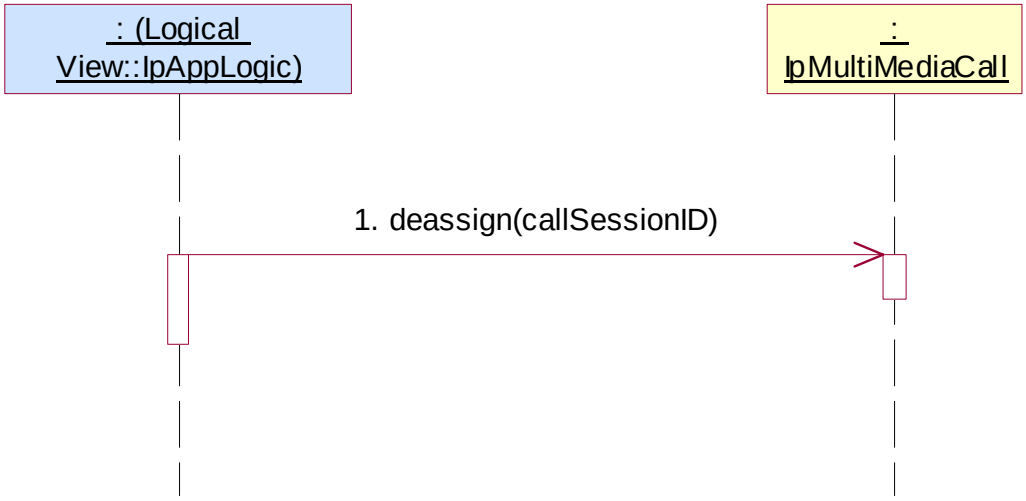
Summary: IpMultiMediaCall, all methods mandatory, successful

Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.3

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 202 915-4-3 [3], clause 6.3

Preamble: Same as MMCC_ IpMultiMediaCall _03

Test Sequence:

1. Method call **deassignCall()** on IpMultiMediaCall
   Parameters: invalid callSessionID
   Check: P_INVALID_SESSION_ID is returned

Test MMCC_ IpMultiMediaCall _06

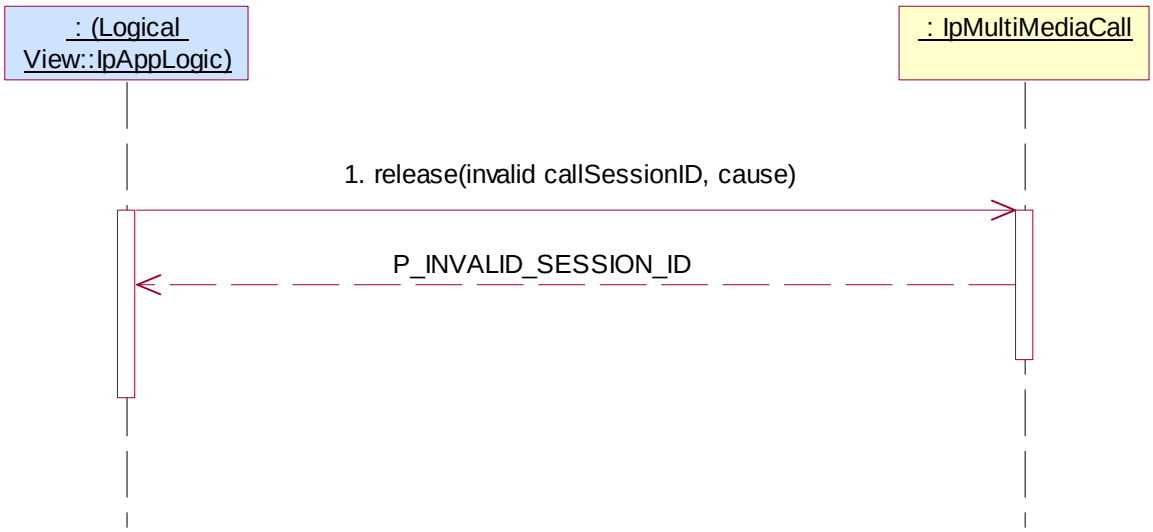
Summary: IpMultiMediaCall, release, P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.3

Preamble: Same as MMCC_ IpMultiMediaCall _03

Test Sequence:

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



Test MMCC_ IpMultiMediaCall _07

Summary: IpMultiMediaCall, createCallLeg, P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.3

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

or

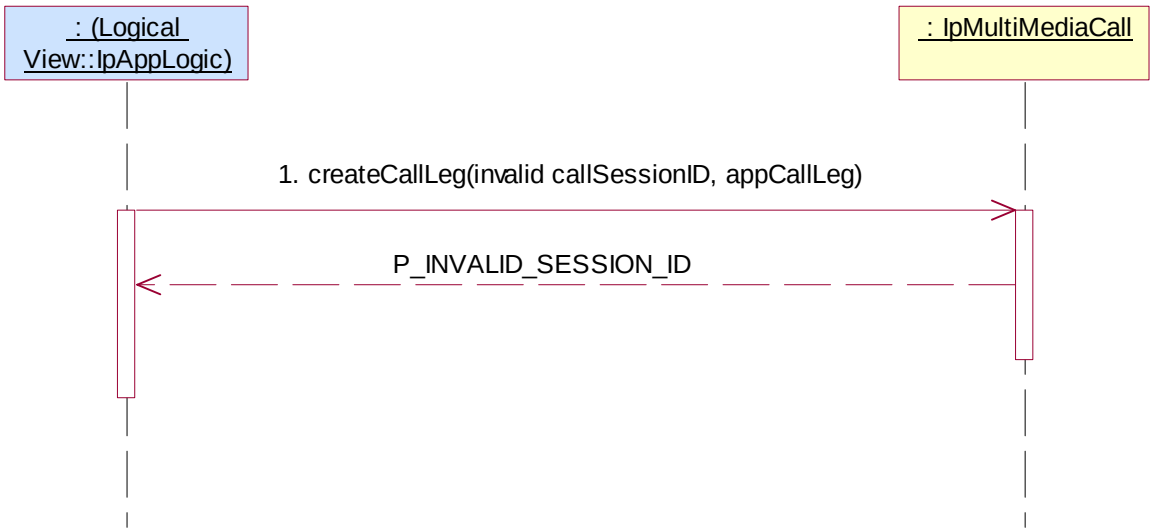
1. Method call **enableNotifications()**
   Parameters: appCallControlManager with valid, non-null, value
   Check: valid value of TpAssignmentID is returned

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 is returned



Test MMCC_ IpMultiMediaCall _08

Summary: IpMultiMediaCall, createCallLeg, P_INVALID_INTERFACE_TYPE

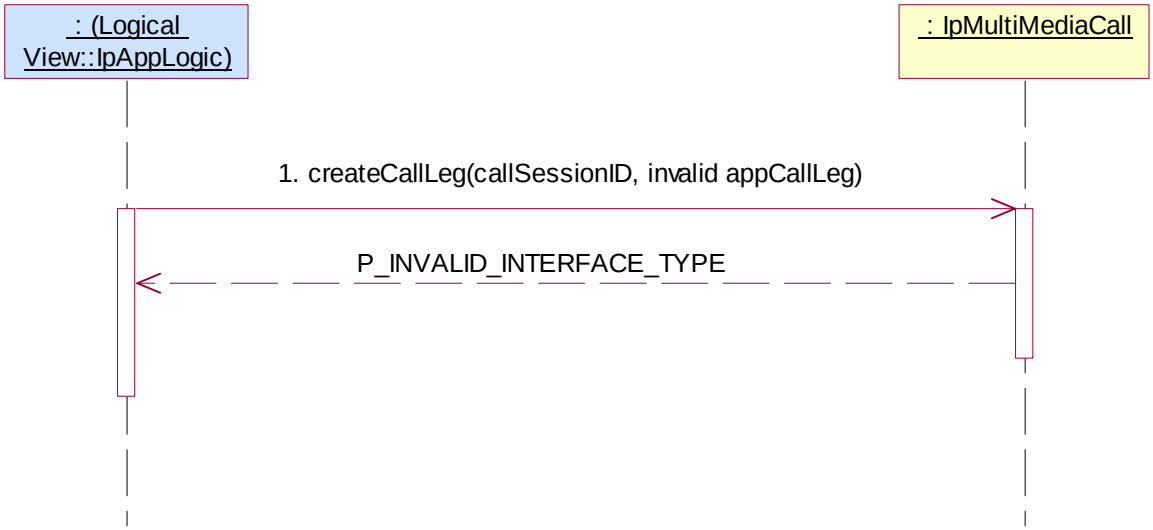
Reference: ES 202 915-4-3 [3], clause 6.3

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 is returned



Test MMCC_ IpMultiMediaCall _09

Summary: IpMultiMediaCall, createAndRouteCallLegReq, P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.3

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 is returned

Test MMCC_ IpMultiMediaCall _10

Summary: IpMultiMediaCall, createAndRouteCallLegReq, P_INVALID_INTERFACE_TYPE

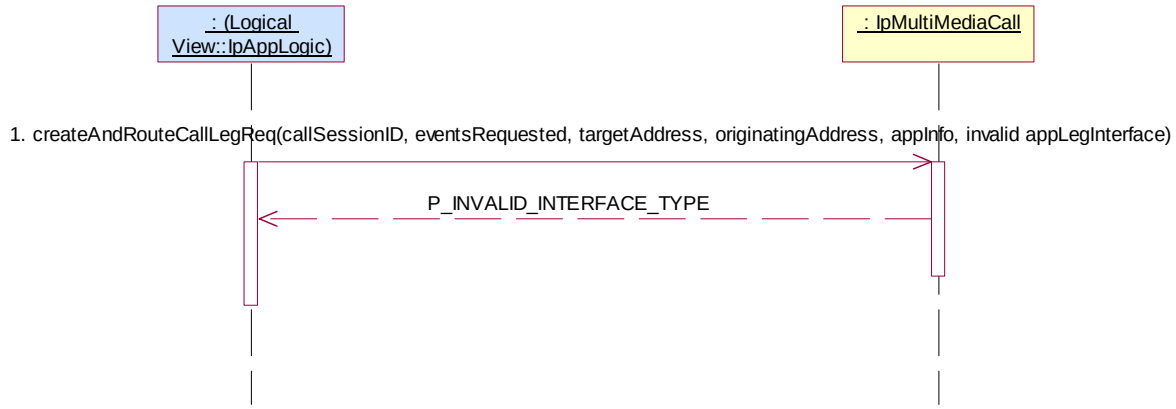
Reference: ES 202 915-4-3 [3], clause 6.3

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 is returned



Test MMCC_ IpMultiMediaCall _11

Summary: IpMultiMediaCall, createAndRouteCallLegReq, P_INVALID_ADDRESS

Reference: ES 202 915-4-3 [3], clause 6.3

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 is returned

Test MMCC_ IpMultiMediaCall _12

Summary: IpMultiMediaCall, createAndRouteCallLegReq, P_INVALID_ADDRESS

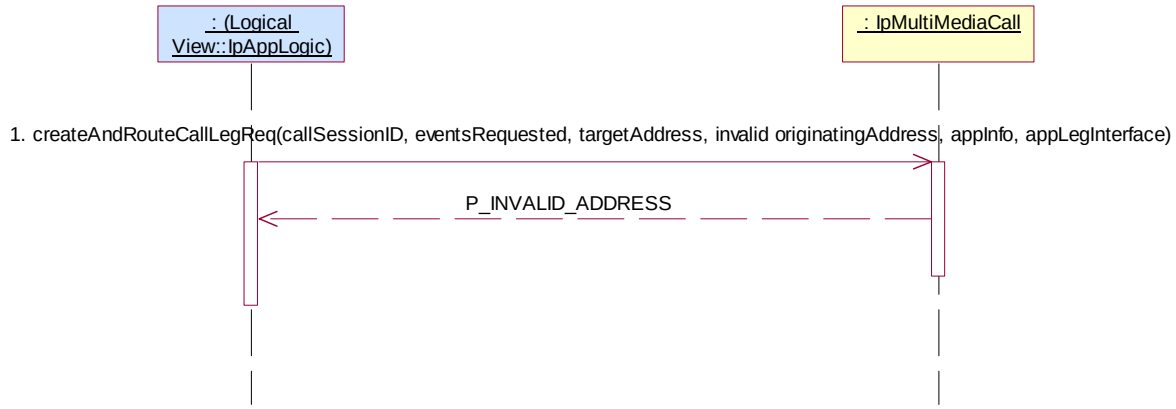
Reference: ES 202 915-4-3 [3], clause 6.3

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 is returned



Test MMCC_ IpMultiMediaCall _13

Summary: IpMultiMediaCall, createAndRouteCallLegReq, P_INVALID_CRITERIA

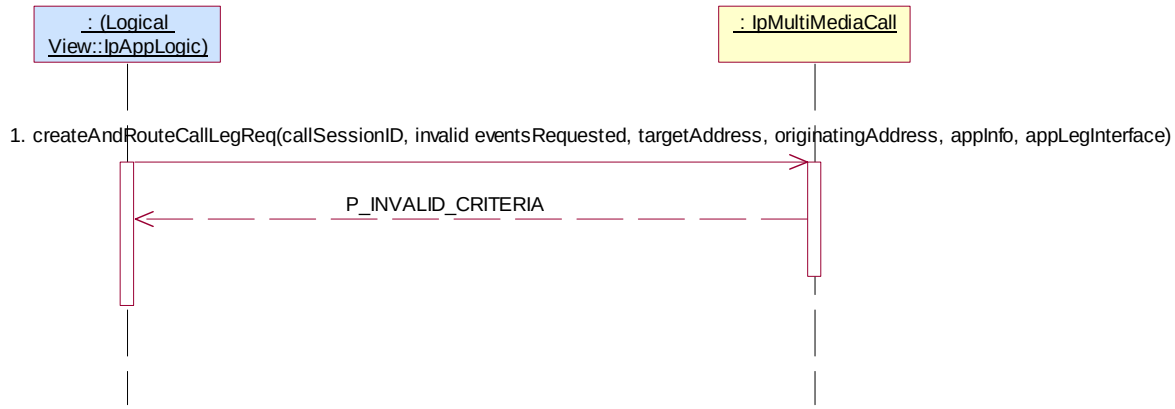
Reference: ES 202 915-4-3 [3], clause 6.3

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 is returned



5.2.3.2.3 Optional, valid behaviour

Test MMCC_ IpMultiMediaCall _14

Summary: IpMultiMediaCall, all methods, successful

Reference: ES 202 915-4-3 [3], clause 6.1 and ES 202 915-4-4 [4] clause 6.3

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

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

or

1. Method call **enableNotifications()**
   Parameters: appCallControlManager with valid, non-null, value
   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

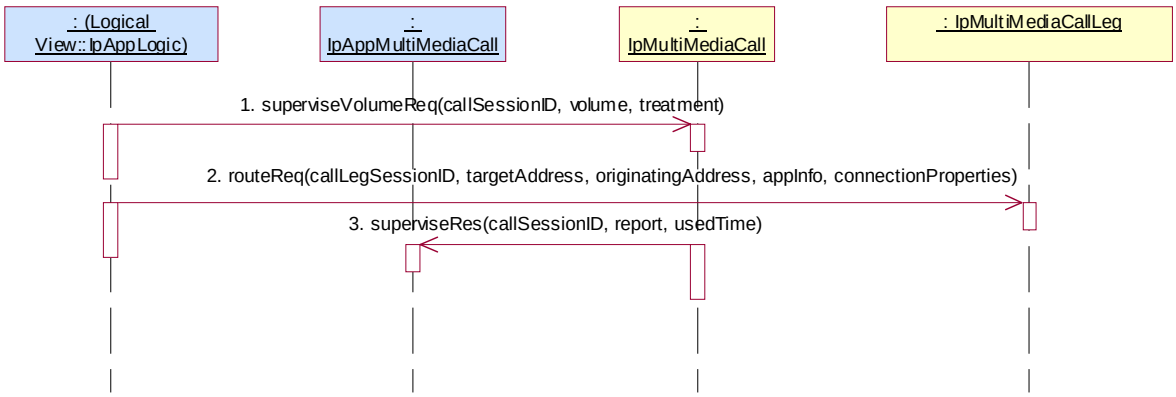
Condition: superviseVolumeReq method is supported.

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



Test MMCC_ IpMultiMediaCall _15

Summary: IpMultiMediaCall, getInfoReq, successful

Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.3

Preamble: Same as MMCC_ IpMultiMediaCall _14

Condition: createCallLeg and getInfoReq methods are supported.

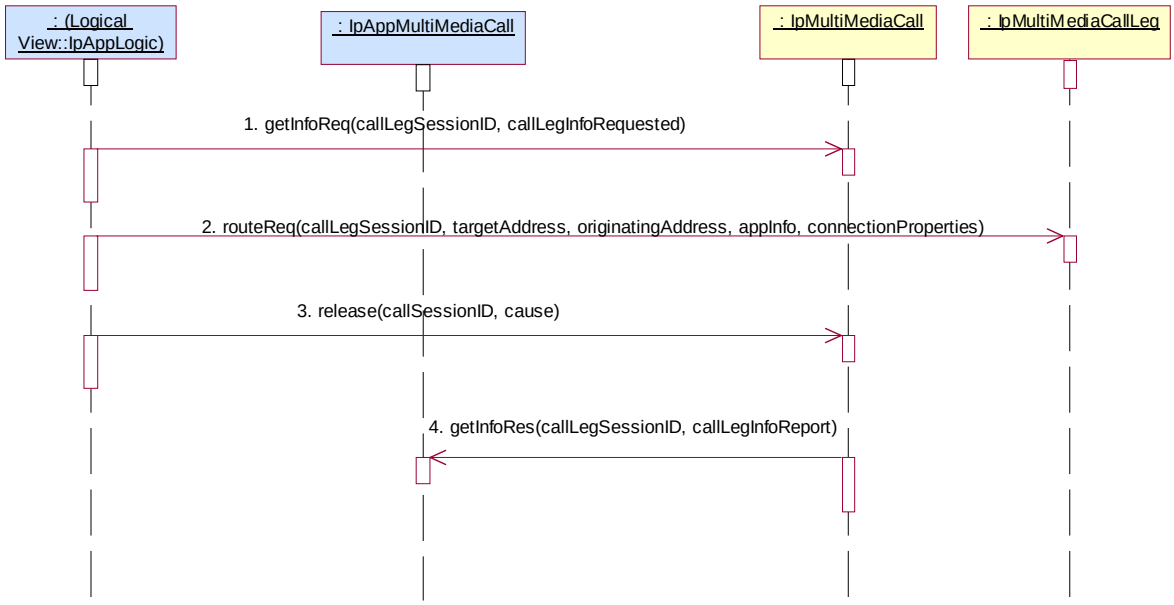
Test Sequence:

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

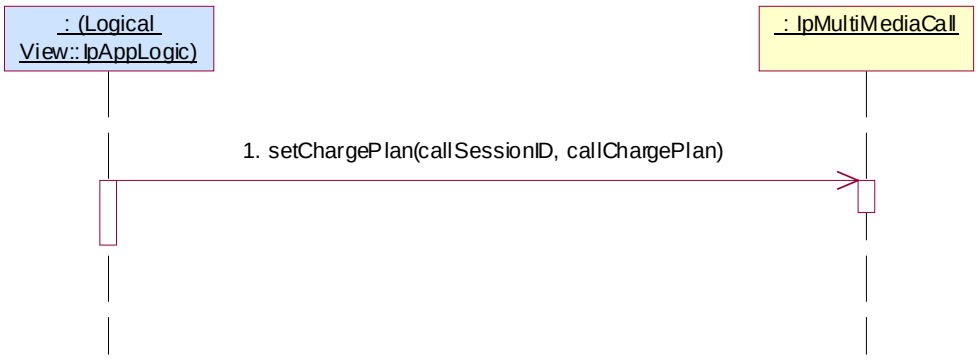
Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.3

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

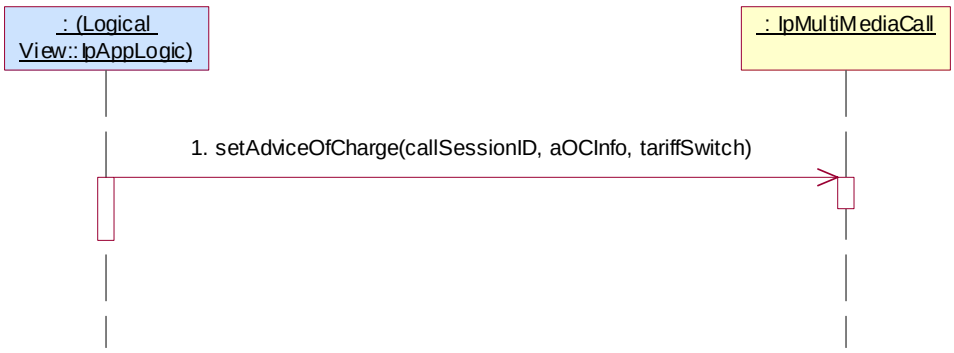
Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.3

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 202 915-4-3 [3], clauses 6.1 and 6.3

Preamble: Same as MMCC_ IpMultiMediaCall _14

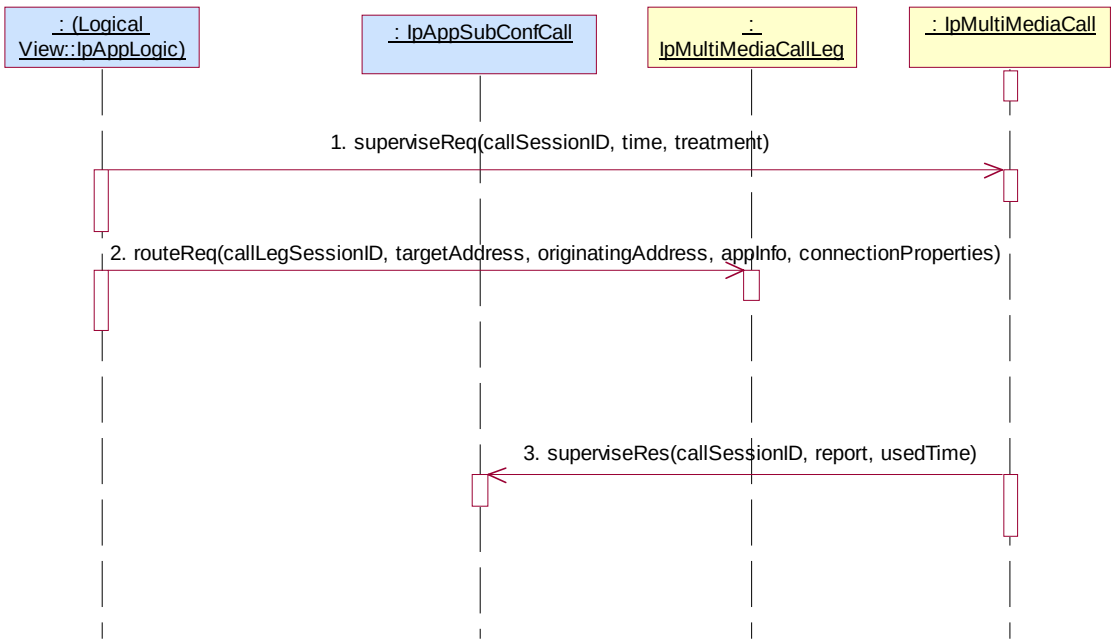
Condition: createCallLeg and superviseReq methods are supported.

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

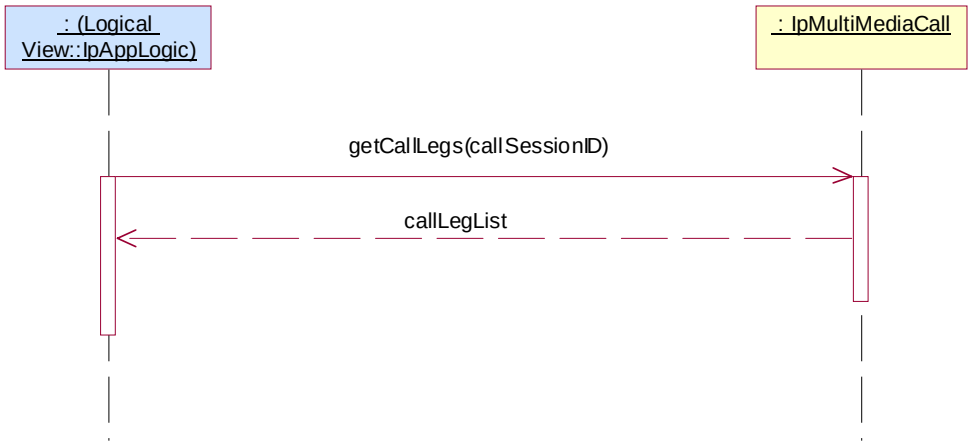
Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.3

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

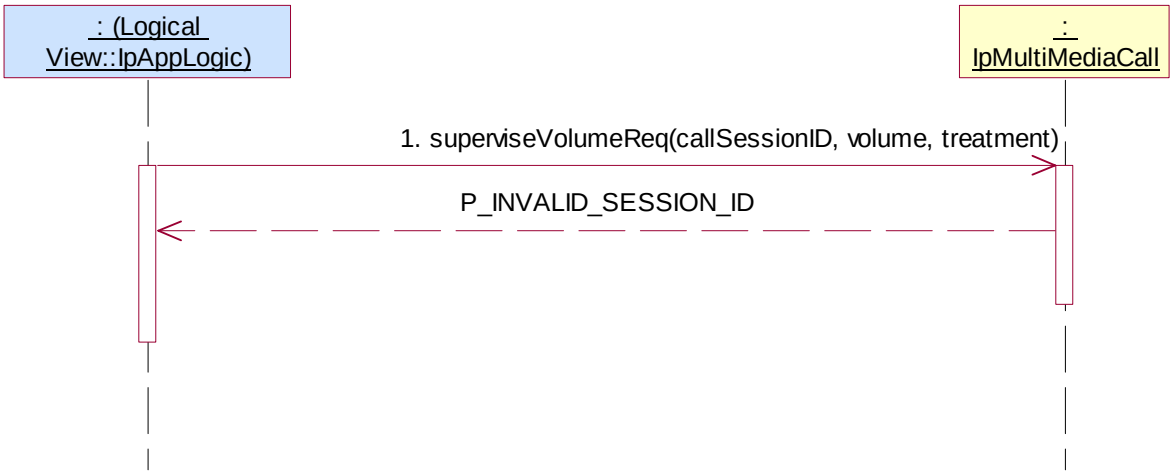
Reference: ES 202 915-4-4 [4], clause 6.3

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 is returned



Test MMCC_ IpMultiMediaCall _21

Summary: IpMultiMediaCall, getInfoReq, P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.3

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 is returned

Test MMCC_ IpMultiMediaCall _22

Summary: IpMultiMediaCall, setChargePlan, P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.3

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 is returned

Test MMCC_ IpMultiMediaCall _23

Summary: IpMultiMediaCall, setAdviceOfCharge, P_INVALID_SESSION_ID

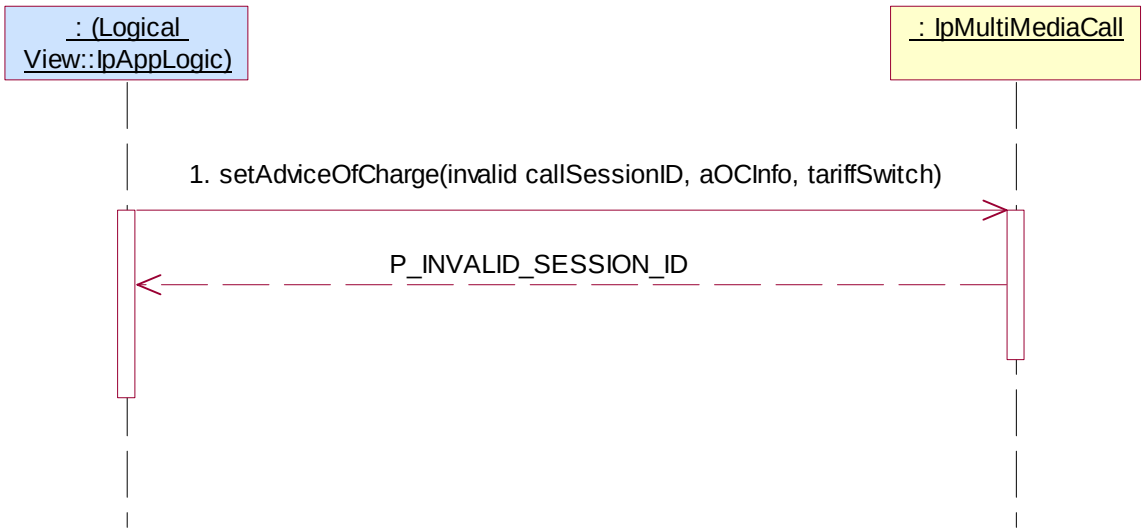
Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.3

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 is returned



Test MMCC_ IpMultiMediaCall _24

Summary: IpMultiMediaCall, setAdviceOfCharge, P_INVALID_CURRENCY

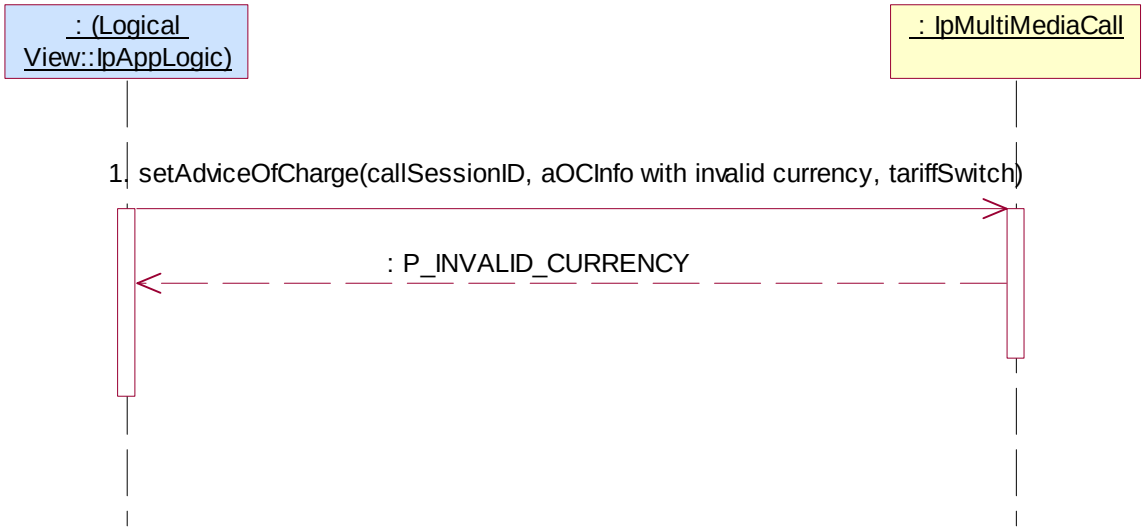
Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.3

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 is returned



Test MMCC_ IpMultiMediaCall _25

Summary: IpMultiMediaCall, setAdviceOfCharge, P_INVALID_AMOUNT

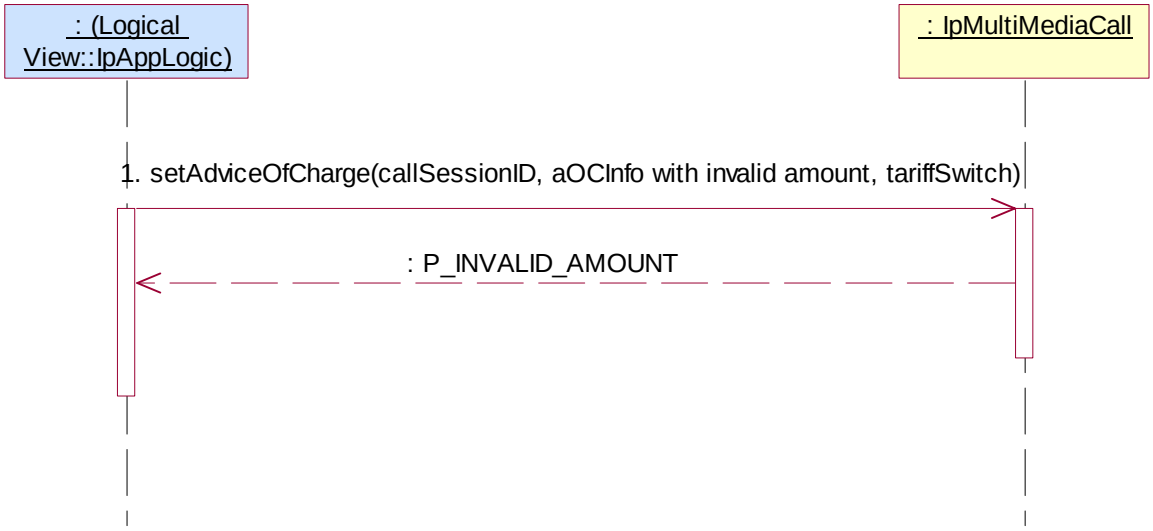
Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.3

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 is returned



Test MMCC_ IpMultiMediaCall _26

Summary: IpMultiMediaCall, superviseReq, P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.3

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 is returned

Test MMCC_ IpMultiMediaCall _27

Summary: IpMultiMediaCall, getCallLegs, P_INVALID_SESSION_ID

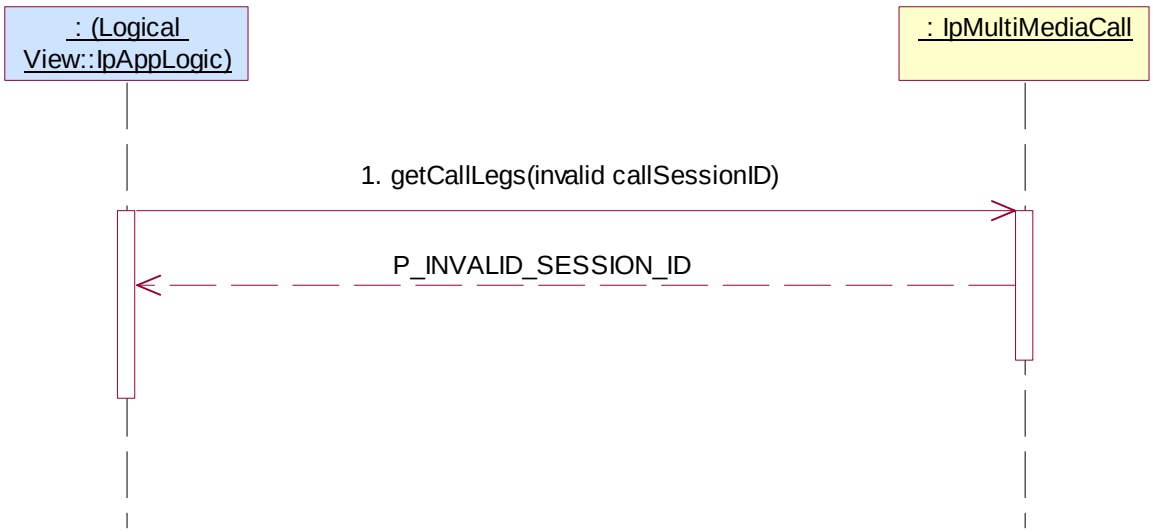
Reference: ES 202 915-4-3 [3], clause 6.3

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 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 202 915-4-3 [3], clauses 6.1 and 6.3 and ES 202 915-4-4 [4], clause 6.5

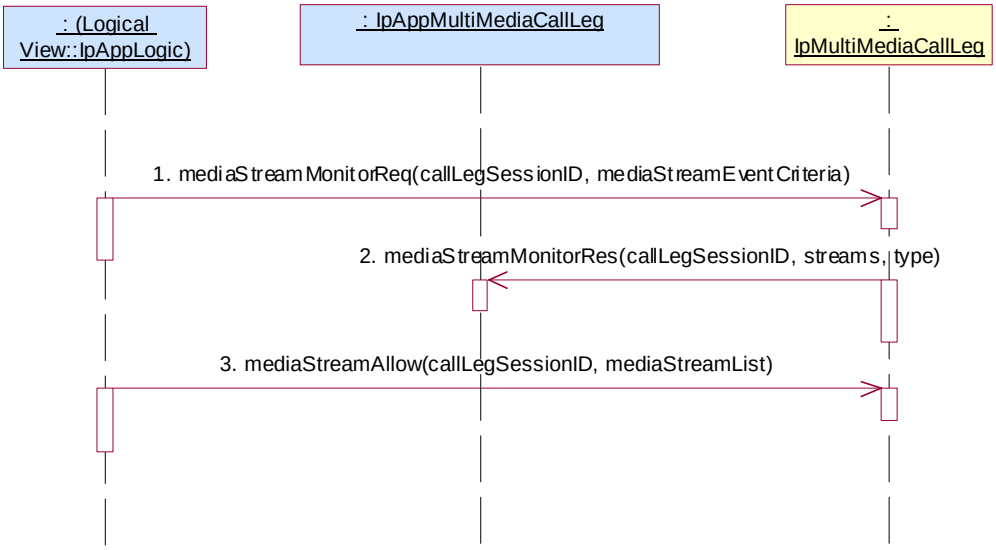
Preamble: Same as MMCC_ IpMultiMediaCall _03

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 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

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

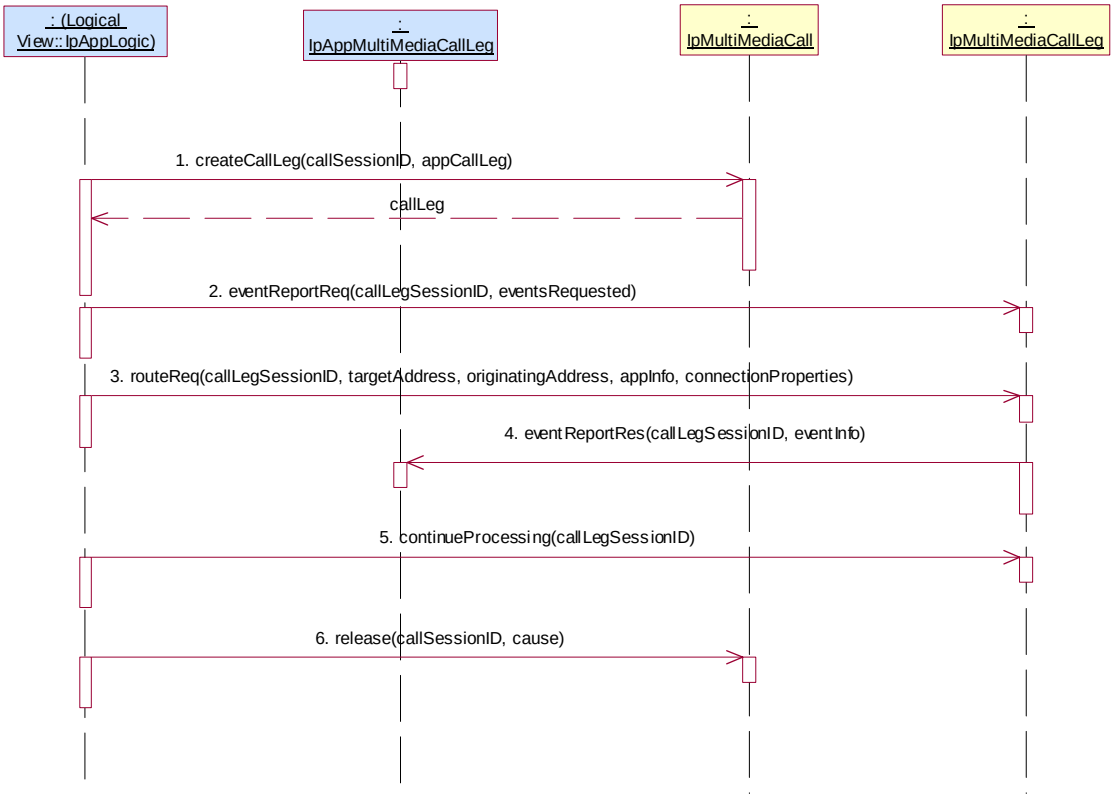
2. Method call **eventReportReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 1., valid eventsRequested with Interrupt event
   Check: no exception is returned

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, eventInfo

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



Test MMCC_ IpMultiMediaCallLeg _03

Summary: IpMultiMediaCallLeg, all mandatory methods, successful

Reference: ES 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

Preamble: Same as MMCC_ IpMultiMediaCall _07

Condition: createAndRouteCallLeg method is supported

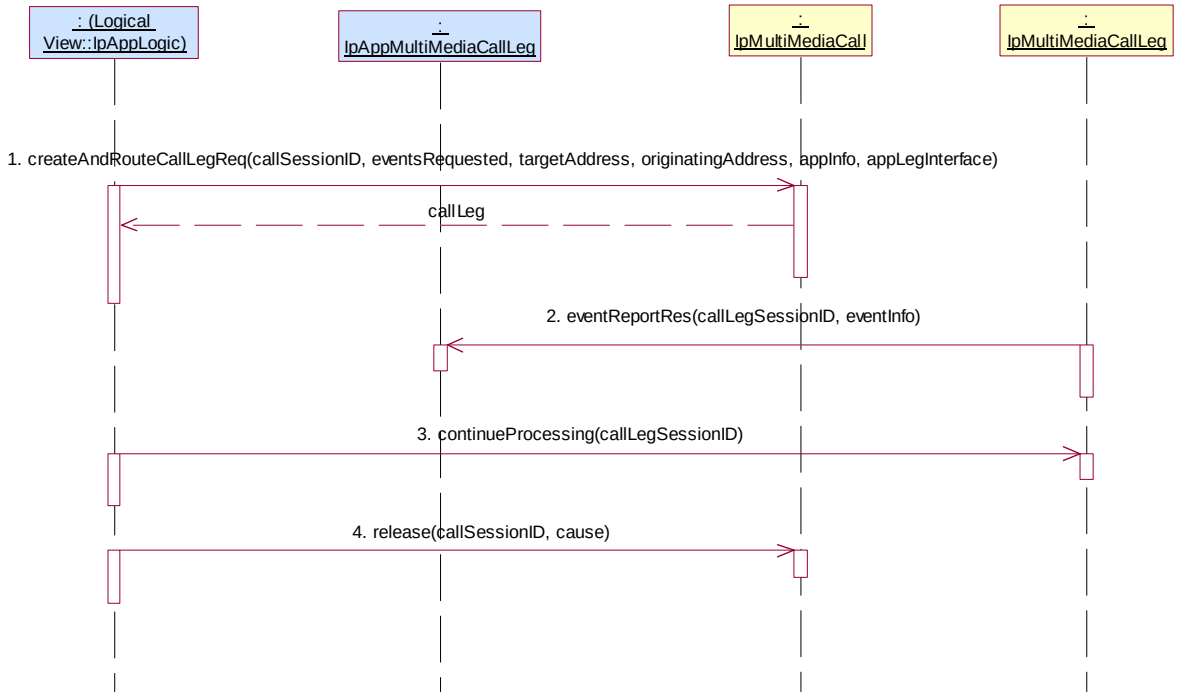
Test Sequence:

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., eventInfo

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 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

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 202 915-4-3 [3], clause 6.5

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 with Interrupt event
   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, eventInfo

5. Method call **continueProcessing()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID 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

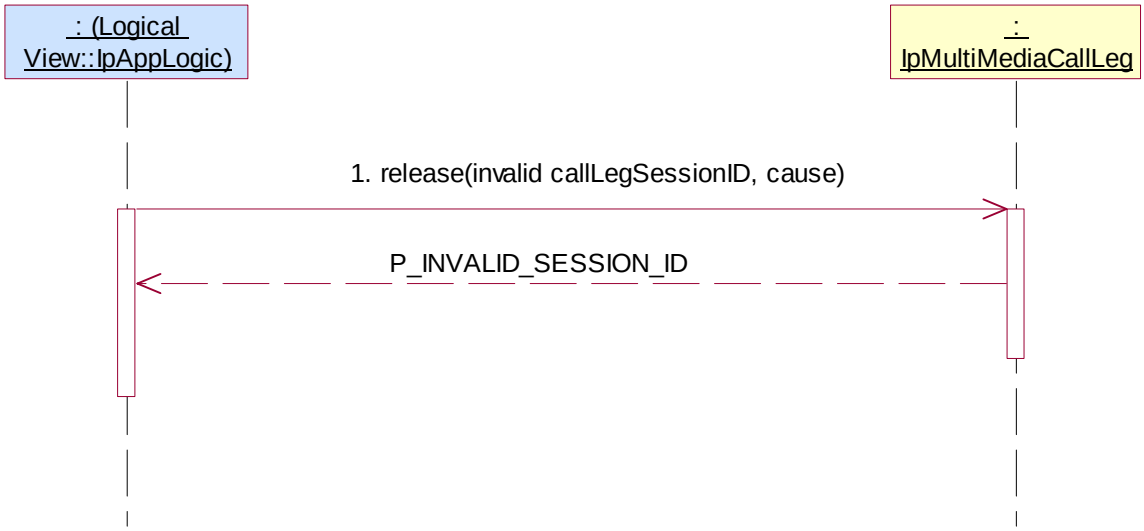
Summary: IpMultiMediaCallLeg, release**,** P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.5

Preamble: Same as MMCC_ IpMultiMediaCall _03

Test Sequence:

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



Test MMCC_ IpMultiMediaCallLeg _07

Summary: IpMultiMediaCallLeg, deassign**,** P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.5

Preamble: Same as MMCC_ IpMultiMediaCall _03

Test Sequence:

1. Method call **deassign()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID is returned

Test MMCC_ IpMultiMediaCallLeg _08

Summary: IpMultiMediaCallLeg, mediaStreamAllow, P_INVALID_SESSION_ID

Reference: ES 202 915-4-4 [4], clause 6.5

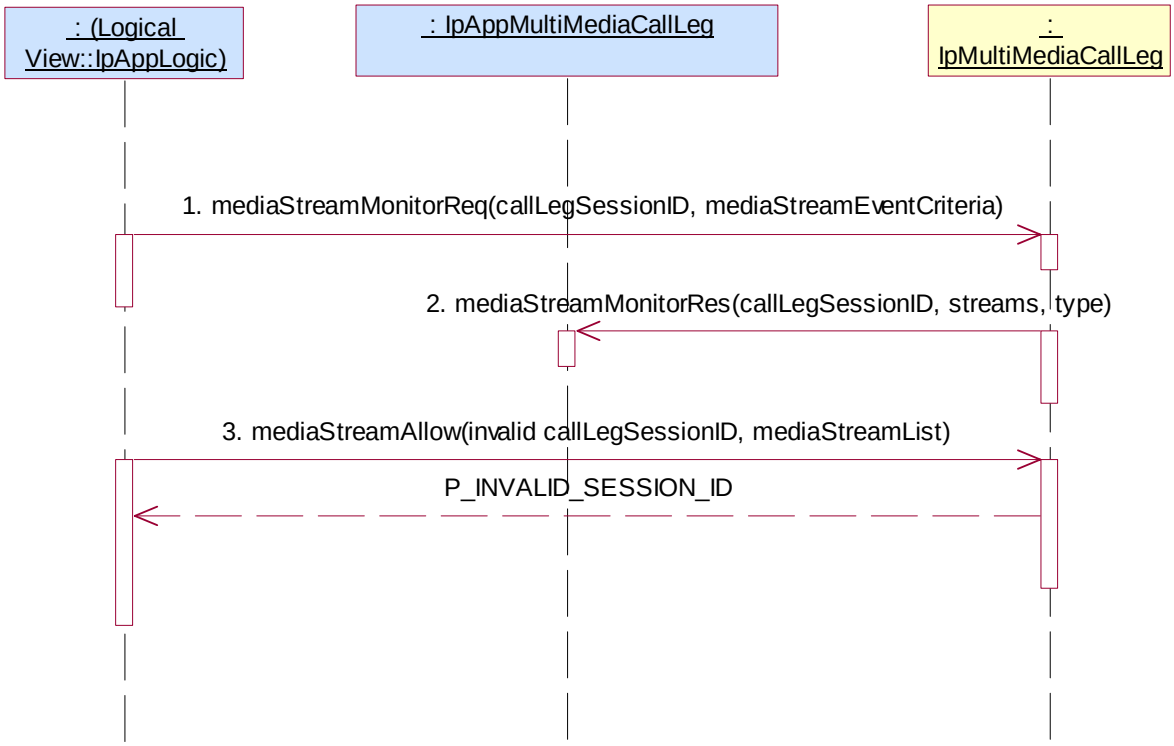
Preamble: Same as MMCC_ IpMultiMediaCall _03

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 is returned



Test MMCC_ IpMultiMediaCallLeg _09

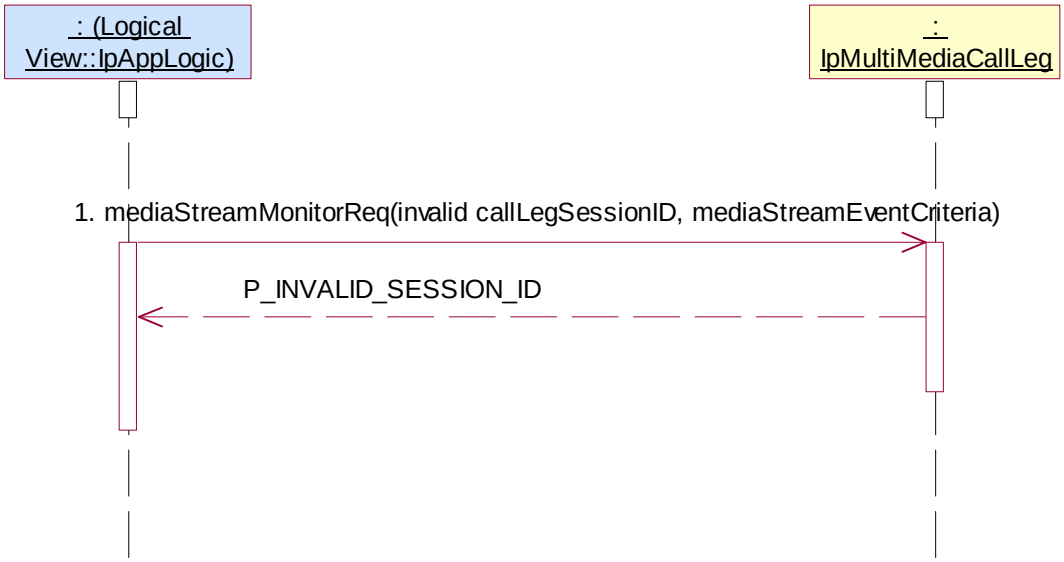
Summary: IpMultiMediaCallLeg, mediaStreamMonitorReq, P_INVALID_SESSION_ID

Reference: ES 202 915-4-4 [4], clause 6.5

Preamble: Same as MMCC_ IpMultiMediaCall _03

Test Sequence:

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



Test MMCC_ IpMultiMediaCallLeg _10

Summary: IpMultiMediaCallLeg, mediaStreamMonitorReq, P_INVALID_CRITERIA

Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.3 and ES 202 915-4-4 [4], clause 6.5

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 is returned

5.2.3.3.3 Optional, valid behaviour

Test MMCC_ IpMultiMediaCallLeg _11

Summary: IpMultiMediaCallLeg, getInfoReq, successful

Reference: ES 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

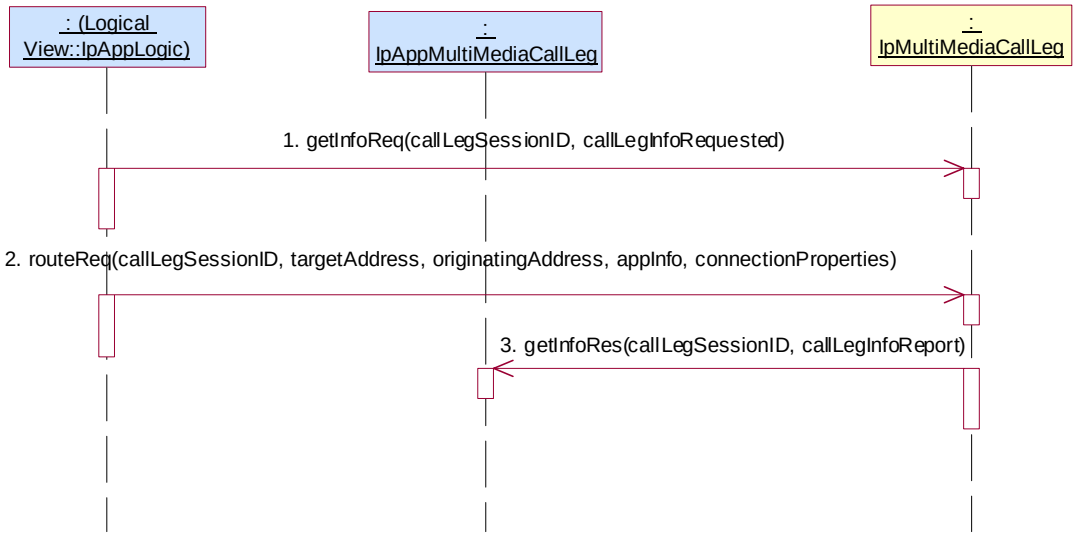
Preamble: Same as MMCC_ IpMultiMediaCall _14

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 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5 and ES 202 915-4-4 [4], clauses 6.5

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

or

1. Method call **enableNotifications()**
   Parameters: appCallControlManager with valid, non-null, value
   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

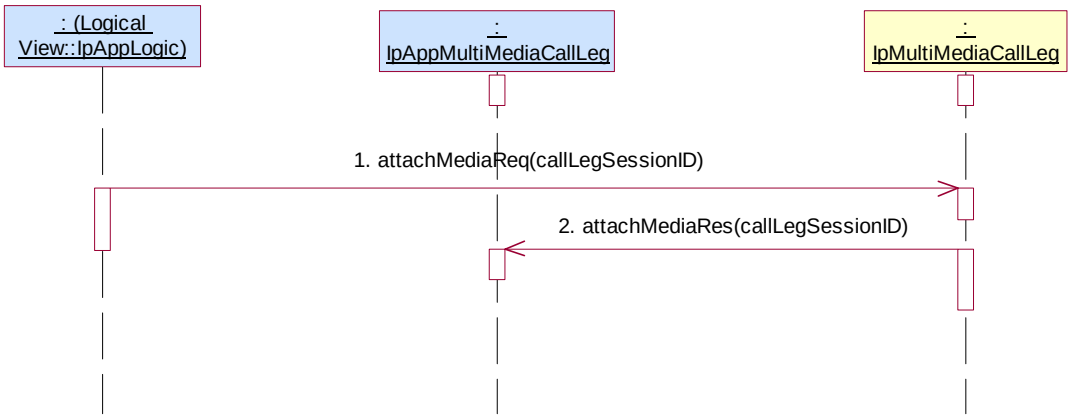
3. Method call **createCallLeg()** on IpMultiMediaCall
   Parameters: valid callSessionID reported in 2., valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

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

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 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

Preamble: Same as MMCC_ IpMultiMediaCallleg _12

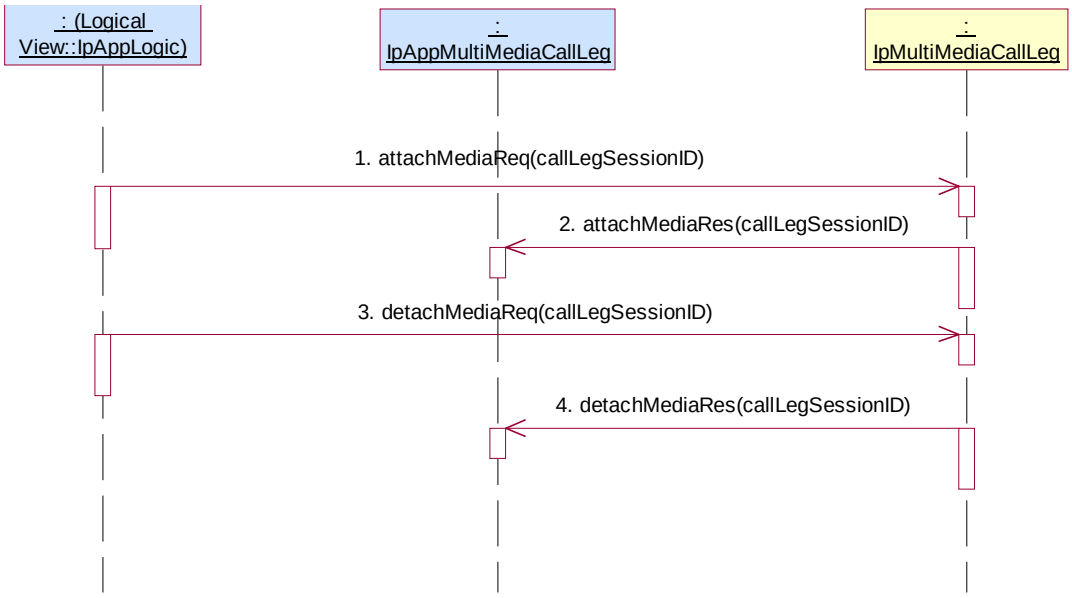
Test Sequence:

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 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

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

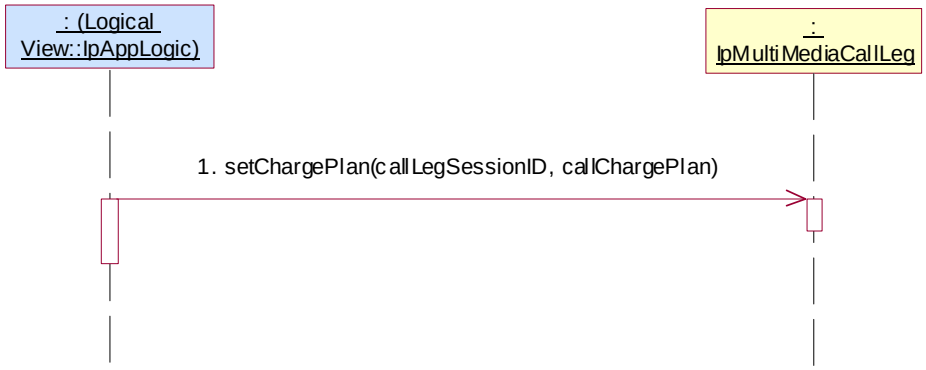
Summary: IpMultiMediaCallLeg, setChargePlan, successful

Reference: ES 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

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

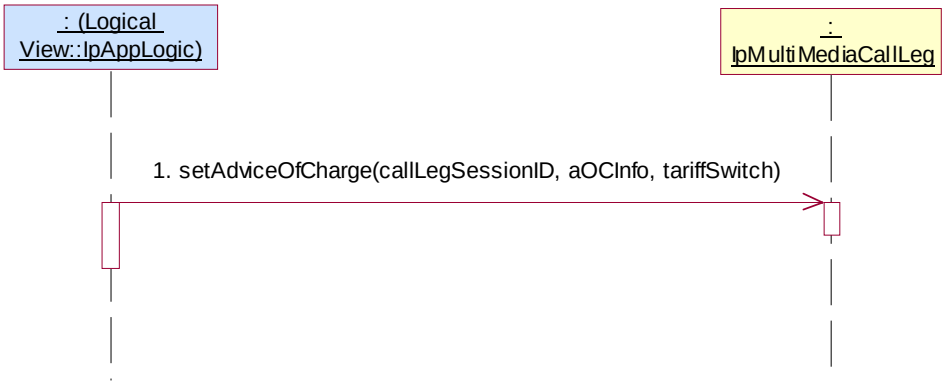
Summary: IpMultiMediaCallLeg, setAdviceOfCharge, successful

Reference: ES 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

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 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

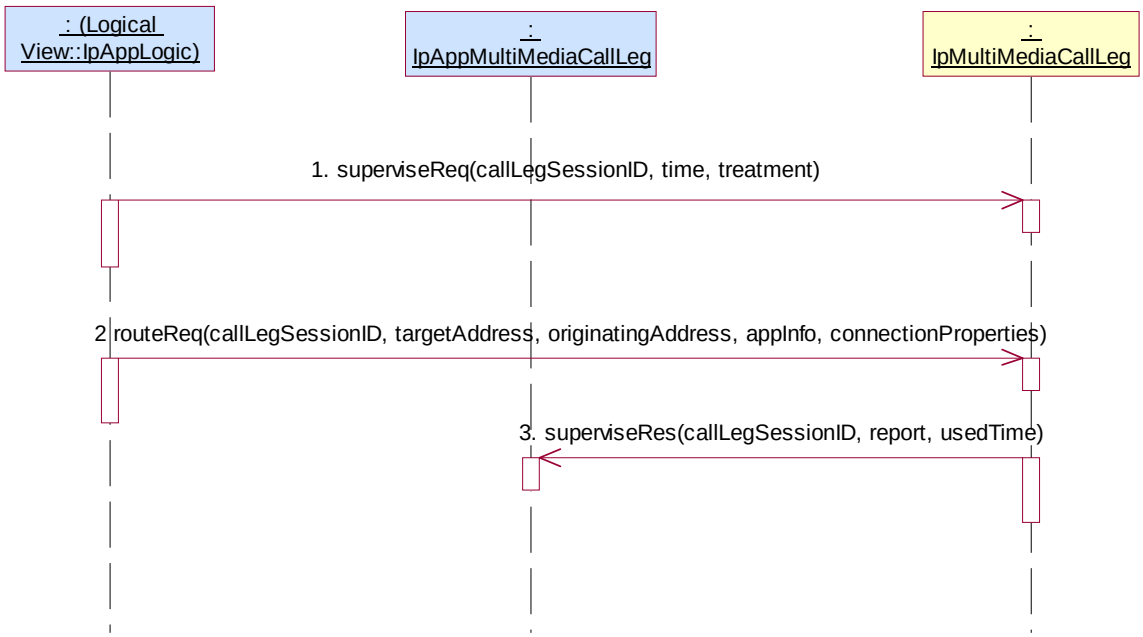
Preamble: Same as MMCC_ IpMultiMediaCall _14

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 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

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, CallLegEnded, successful

Reference: ES 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

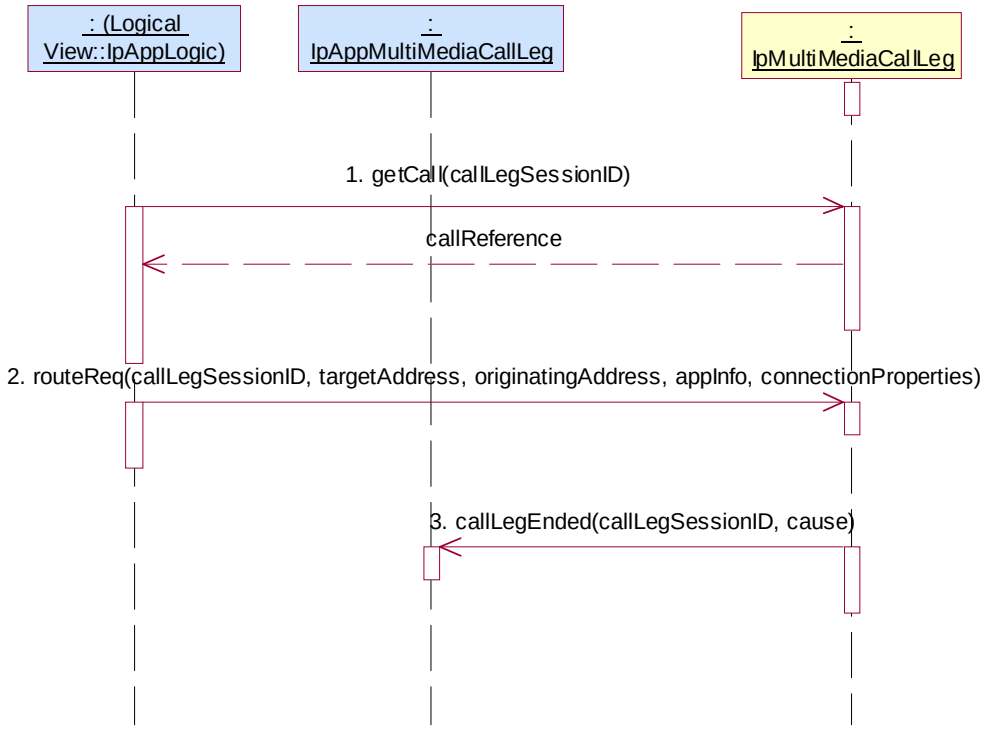
Preamble: Same as MMCC_ IpMultiMediaCall _14

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 202 915-4-3 [3], clause 6.5

Preamble: Same as MMCC_ IpMultiMediaCall _14

Test Sequence:

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

Test MMCC_ IpMultiMediaCallLeg _21

Summary: IpMultiMediaCallLeg, attachMediaReq: P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.5

Preamble: Same as MMCC_ IpMultiMediaCallLeg _12

Test Sequence:

1. Method call **attachMediaReq()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID is returned

Test MMCC_ IpMultiMediaCallLeg _22

Summary: IpMultiMediaCallLeg, detachMediaReq, : P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.5

Preamble: Same as MMCC_ IpMultiMediaCall _03

Test Sequence:

1. Method call **detachMediaReq()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID is returned

Test MMCC_ IpMultiMediaCallLeg _23

Summary: IpMultiMediaCallLeg, getCurrentDestinationAddress, : P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.5

Preamble: Same as MMCC_ IpMultiMediaCall _03

Test Sequence:

1. Method call **getCurrentDestinationAddress()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID is returned

Test MMCC_ IpMultiMediaCallLeg _24

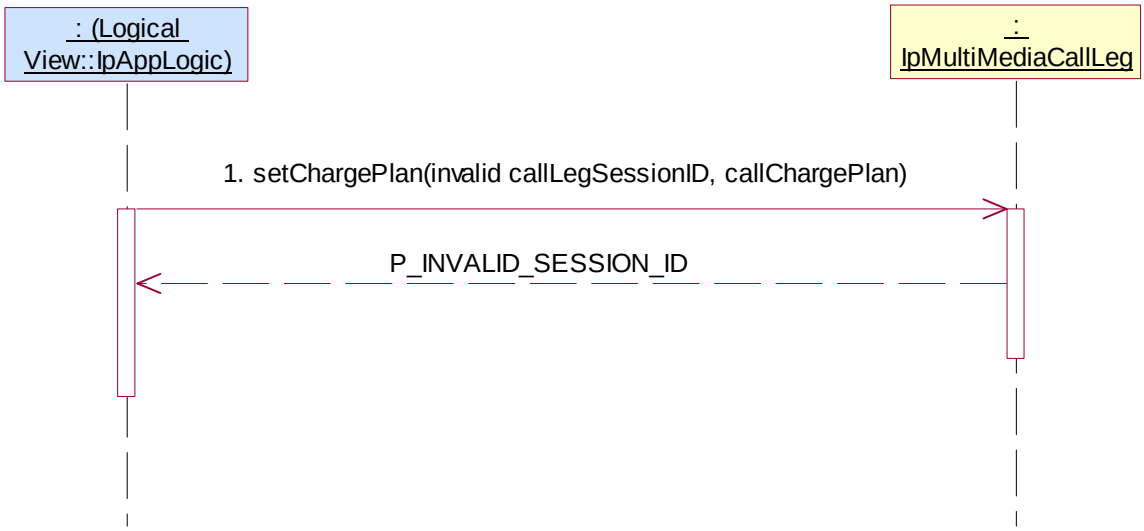
Summary: IpMultiMediaCallLeg, setChargePlan: P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.5

Preamble: Same as MMCC_ IpMultiMediaCall _14

Test Sequence:

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



Test MMCC_ IpMultiMediaCallLeg _25

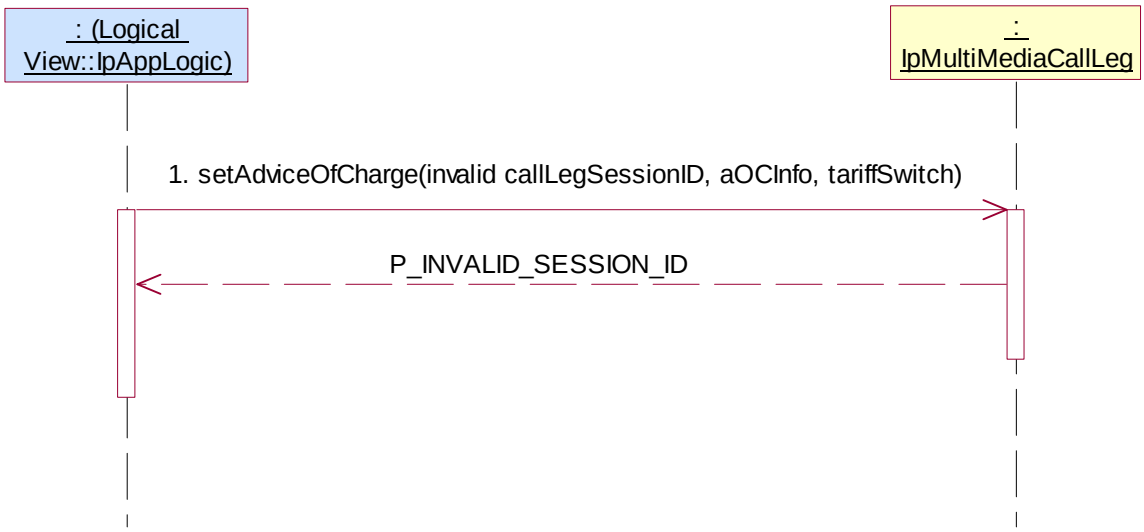
Summary: IpMultiMediaCallLeg, setAdviceOfCharge, : P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.5

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 is returned



Test MMCC_ IpMultiMediaCallLeg _26

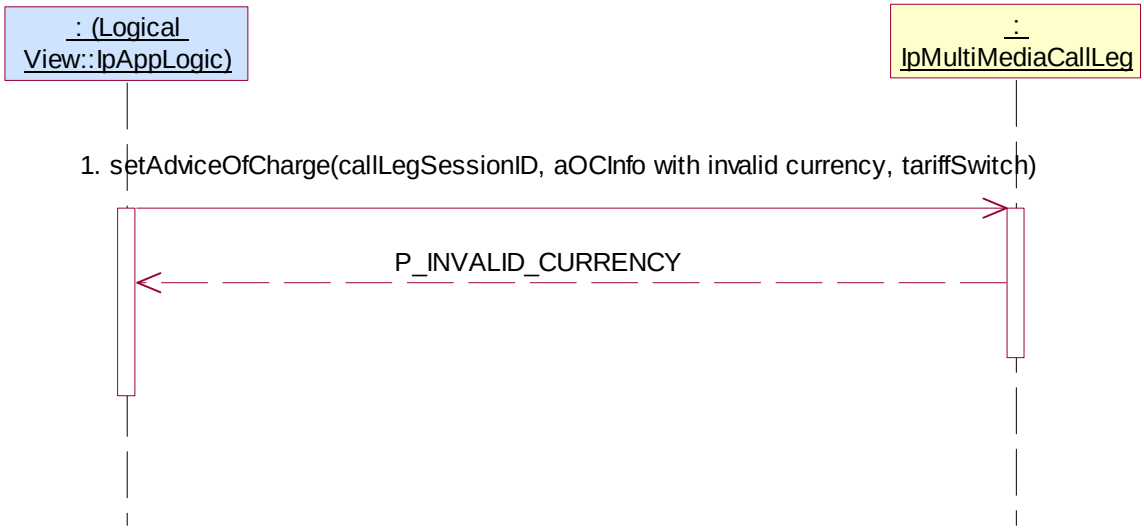
Summary: IpMultiMediaCallLeg, setAdviceOfCharge, P_INVALID_CURRENCY

Reference: ES 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

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 is returned



Test MMCC_ IpMultiMediaCallLeg _27

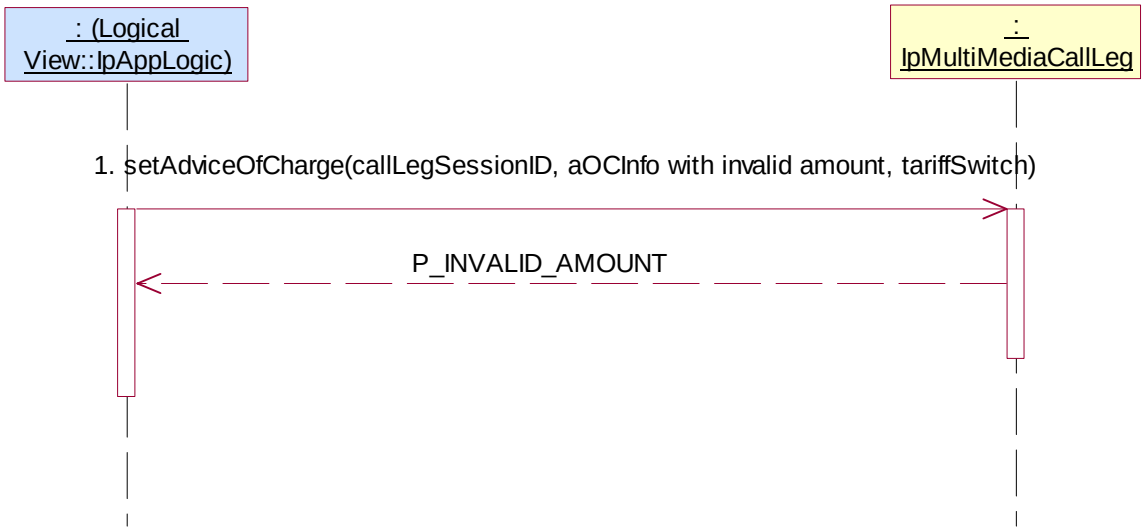
Summary: IpMultiMediaCallLeg, setAdviceOfCharge, P_INVALID_AMOUNT

Reference: ES 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

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 is returned



Test MMCC_ IpMultiMediaCallLeg _28

Summary: IpMultiMediaCallLeg, superviseReq, : P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.5

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 is returned

Test MMCC_ IpMultiMediaCallLeg _29

Summary: IpMultiMediaCallLeg, routeReq, : P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.5

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 is returned

Test MMCC_ IpMultiMediaCallLeg _30

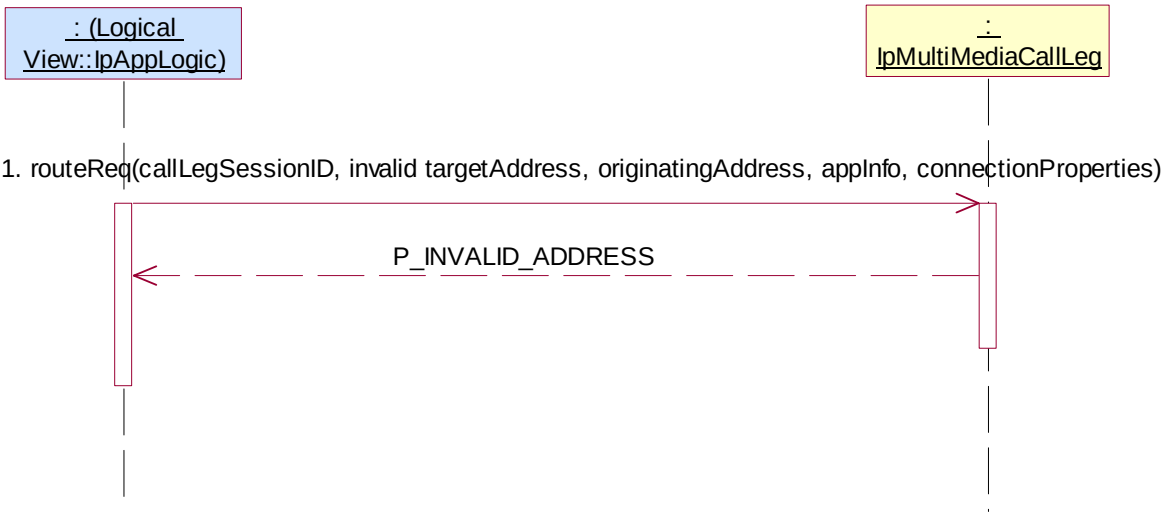
Summary: IpMultiMediaCallLeg, routeReq, P_INVALID_ADDRESS

Reference: ES 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

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 is returned



Test MMCC_ IpMultiMediaCallLeg _31

Summary: IpMultiMediaCallLeg, routeReq, P_INVALID_ADDRESS

Reference: ES 202 915-4-3 [3], clause 6.5

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 is returned

Test MMCC_ IpMultiMediaCallLeg _32

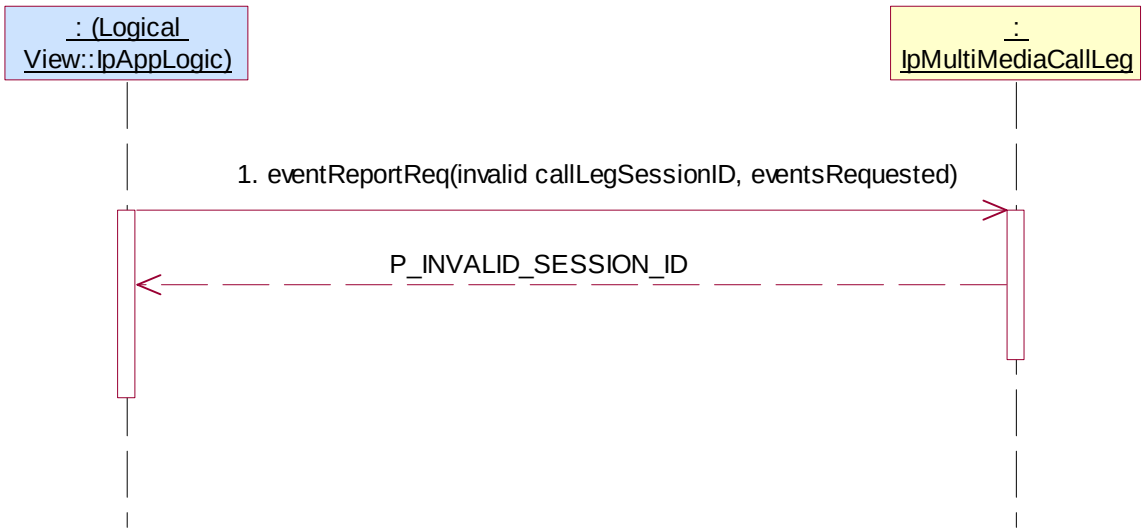
Summary: IpMultiMediaCallLeg, eventReportReq**,** P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.5

Preamble: Same as MMCC_ IpMultiMediaCall _14

Test Sequence:

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



Test MMCC_ IpMultiMediaCallLeg _33

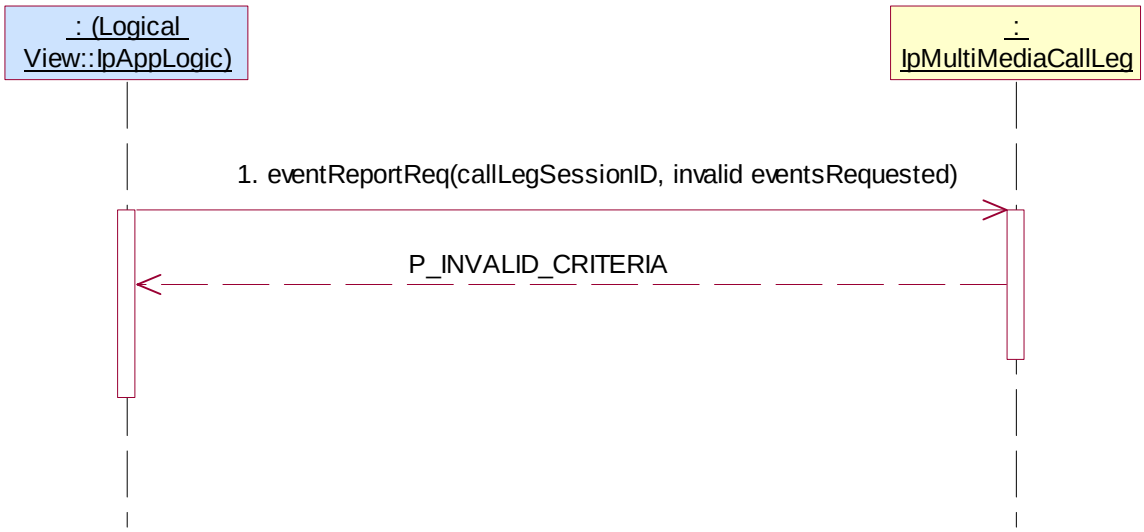
Summary: IpMultiMediaCallLeg, eventReportReq, P_INVALID_CRITERIA

Reference: ES 202 915-4-3 [3], clauses 6.1, 6.3 and 6.5

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 is returned



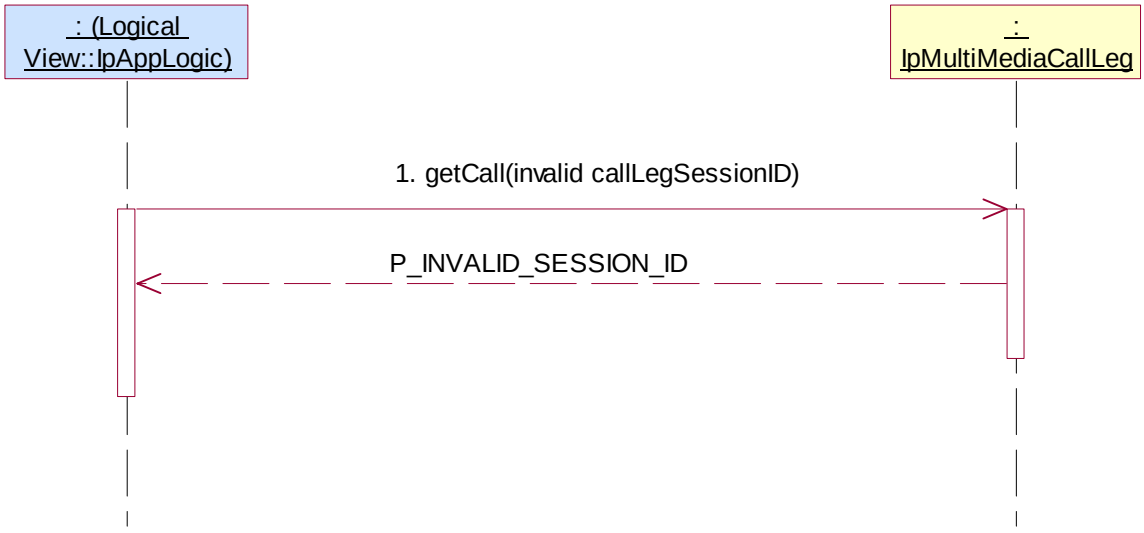
Test MMCC_ IpMultiMediaCallLeg _34

Summary: IpMultiMediaCallLeg, getCall**,** P_INVALID_SESSION_ID

Preamble: Same as MMCC_ IpMultiMediaCall _03

Reference: ES 202 915-4-3 [3], clause 6.5

1. Method call **getCall()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID is returned



Test MMCC_ IpMultiMediaCallLeg _35

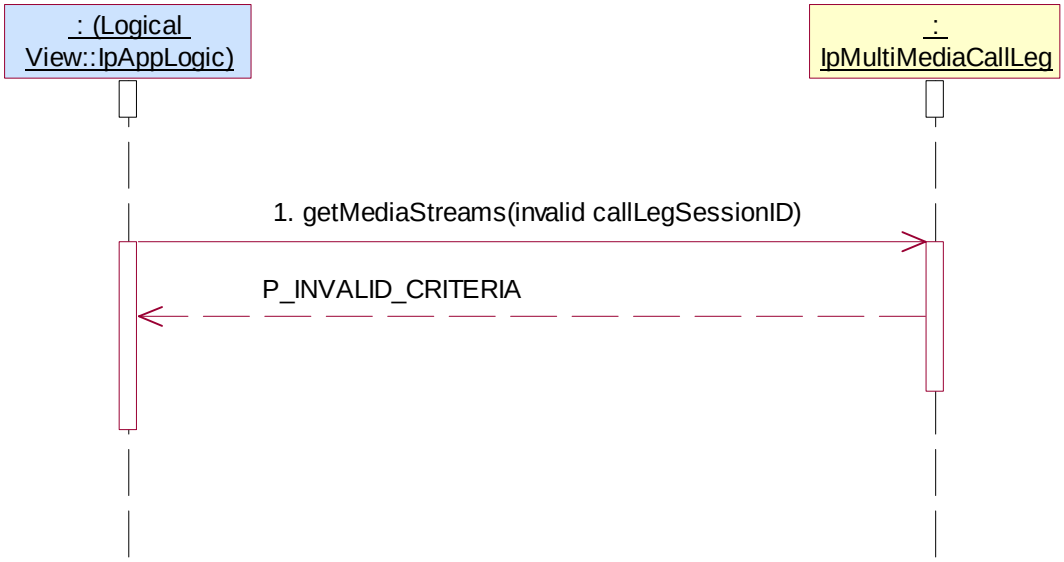
Summary: IpMultiMediaCallLeg, getMediaStreams, P_INVALID_SESSION_ID

Reference: ES 202 915-4-4 [4], clause 6.5

Preamble: Same as MMCC_ IpMultiMediaCall _03

Test Sequence:

1. Method call **getMediaStreams()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID 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 202 915-4-3 [3], clauses 6.1 and 6.3 and ES 202 915-4-4 [4], clauses 6.5 and 6.7

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

or

1. Method call **enableNotifications()**
   Parameters: appCallControlManager with valid, non-null, value
   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

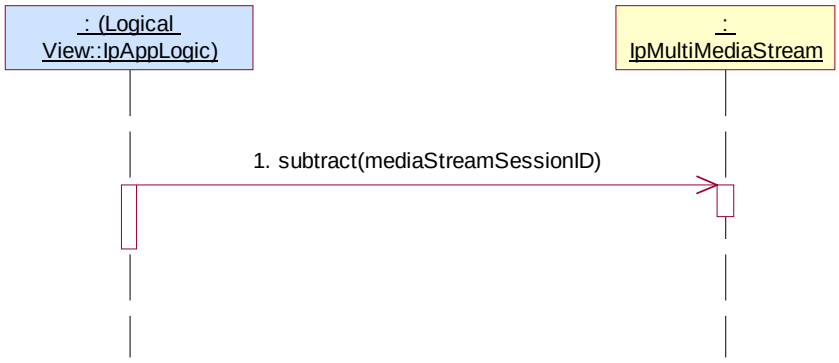
5. Method call **mediaStreamMonitorReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 3., valid mediaStreamEventCriteria
   Check: no exception is returned

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

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

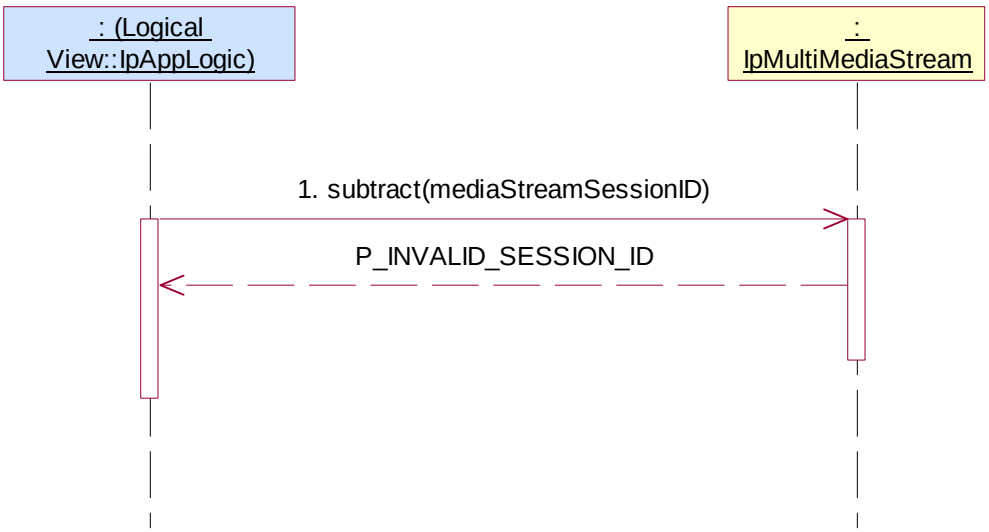
Summary: IpMultiMediaStream, subtract, P_INVALID_SESSION_ID

Reference: ES 202 915-4-4 [4], clause 6.7

Preamble: Same as MMCC_ IpMultiMediaStream _01

Test Sequence:

1. Method call **subtract()** on IpMultiMediaStream
   Parameters: invalid mediaStreamSessionID
   Check: P_INVALID_SESSION_ID is returned



5.2.4 Conference Call Control Service (CCC)

The TPs in this clause are based on ES 202 915-4-5 [5].

5.2.4.1 IpConfCallControlManager

5.2.4.1.1 Mandatory, valid behaviour

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

Test CCC _ IpConfCallControlManager _01

Summary: IpConfCallControlManager, all mandatory methods, successful

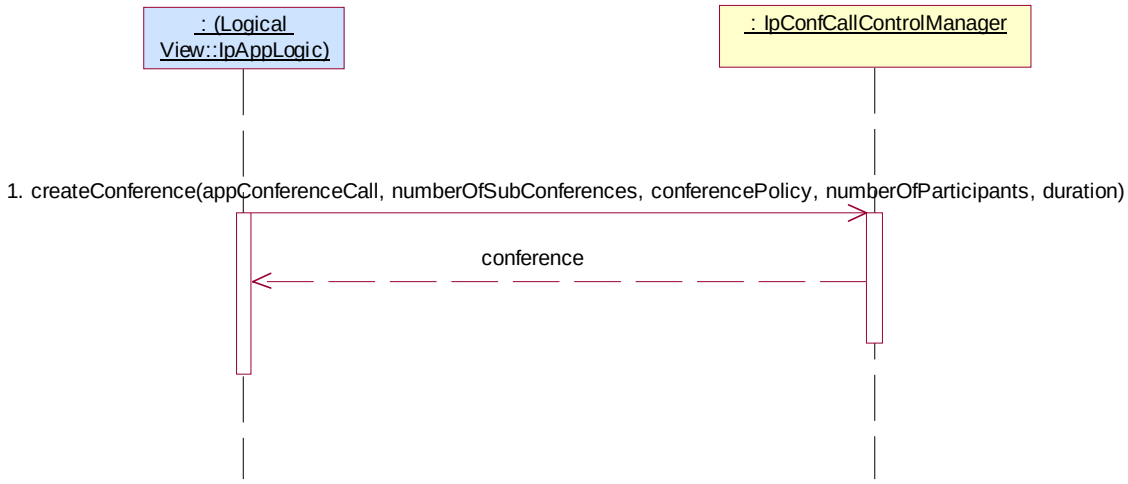
Reference: ES 202 915-4-5 [5], clause 6.1

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 202 915-4-5 [5], clause 6.1

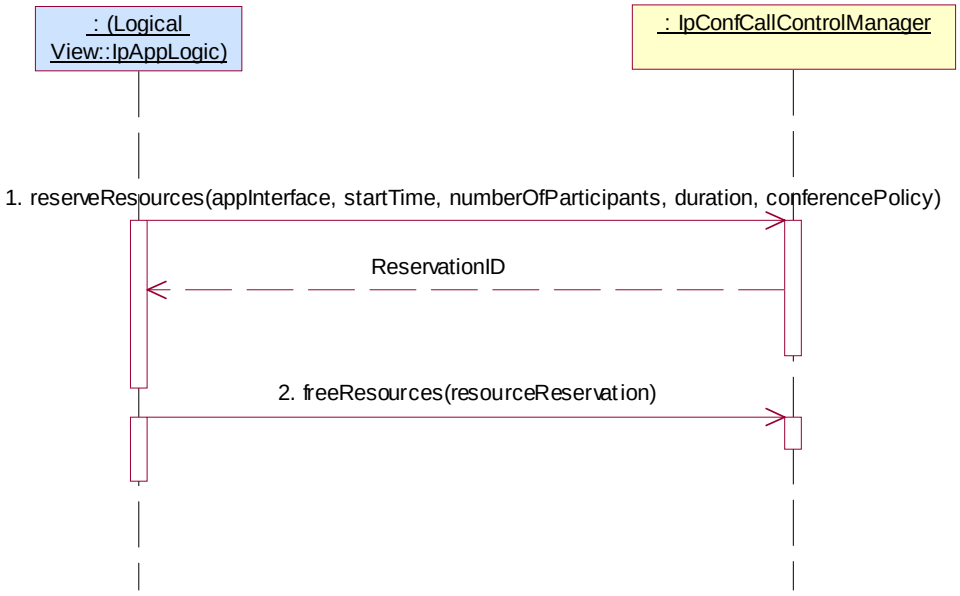
Preamble: Application has a reference interface used for callbacks.

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 202 915-4-5 [5], clause 6.1

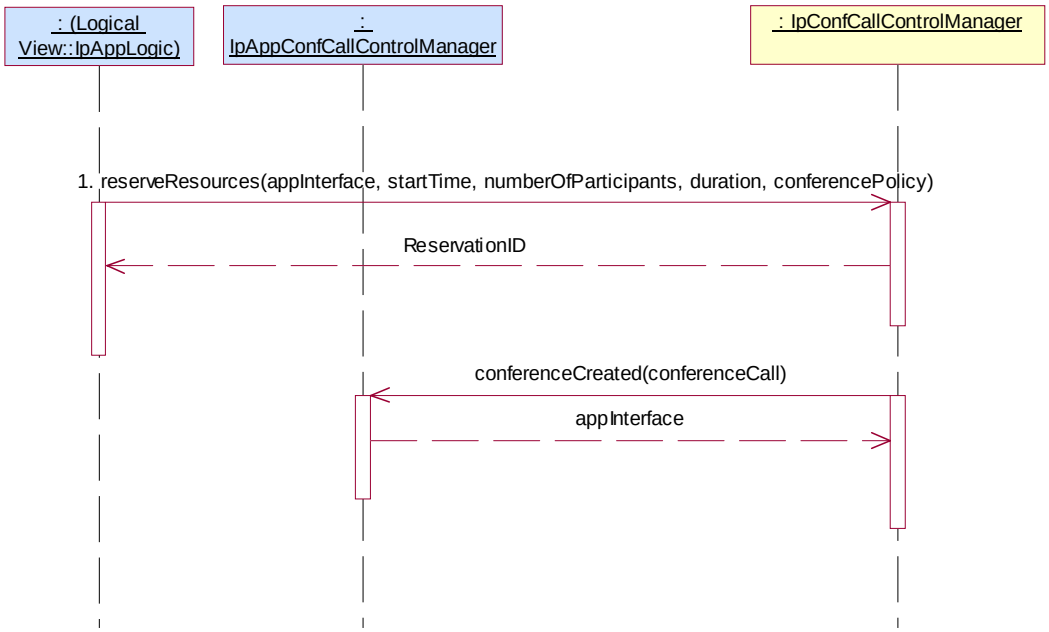
Preamble: Application has a reference interface used for callbacks.

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 202 915-4-5 [5], clause 6.1

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

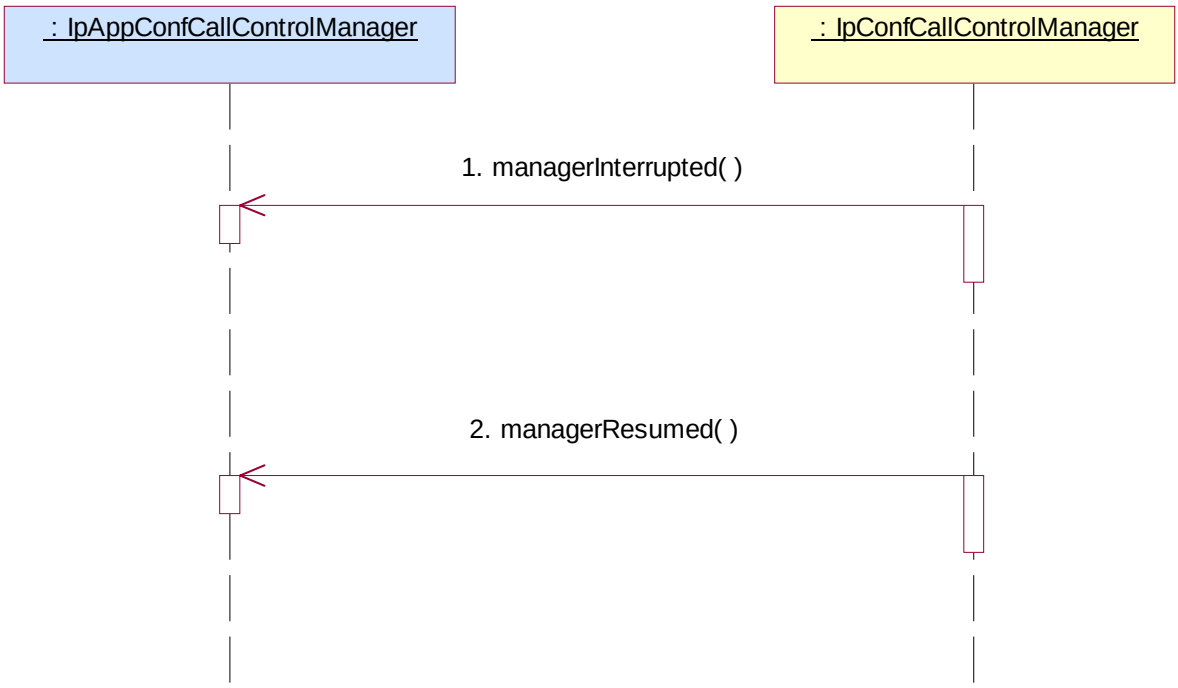
Reference: ES 202 915-4-3 [3], clause 6.1

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 202 915-4-5 [5], clauses 6.1 and 6.3

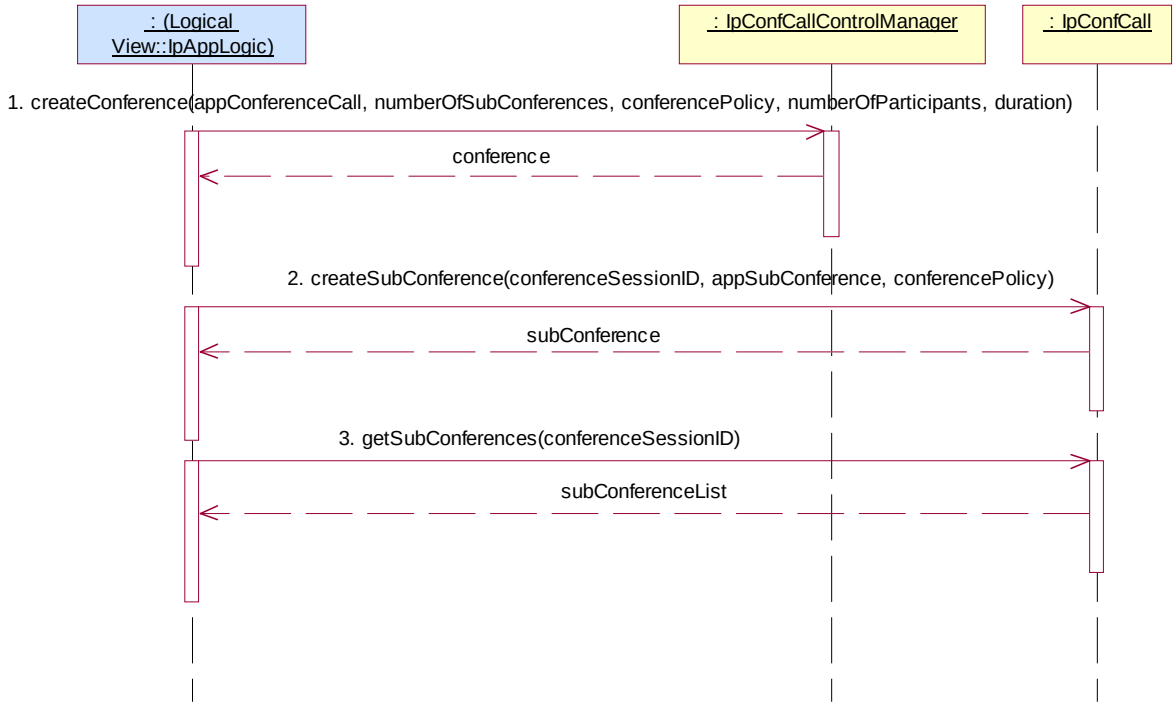
Preamble: Application has a reference interface used for callbacks.

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 202 915-4-5 [5], clauses 6.1 and 6.3 and ES 202 915-4-3 [3], clause 6.34

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

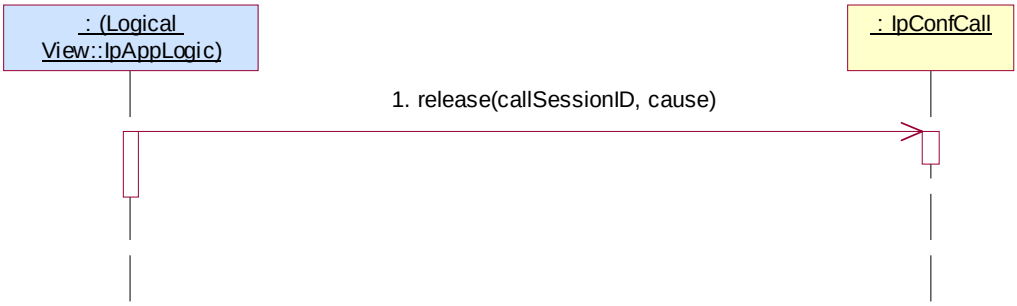
1. Method call **reserveResources()**
   Parameters: valid appInterface, valid startTime, valid numberOfParticipants, valid duration, valid conferencePolicy
   Check: valid value of TpResourceReservation is returned

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

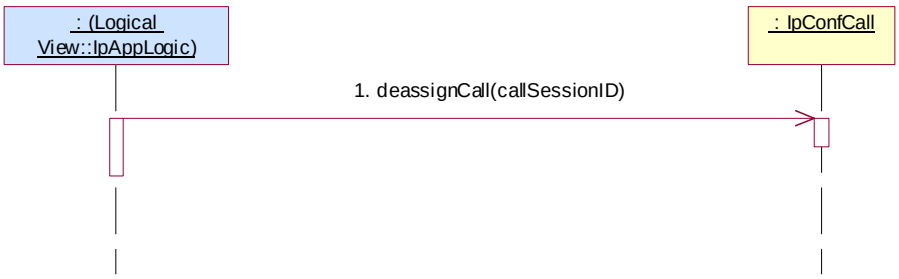
Summary: IpConfCall, all methods mandatory, successful

Reference: ES 202 915-4-3 [3], clause 6.3

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

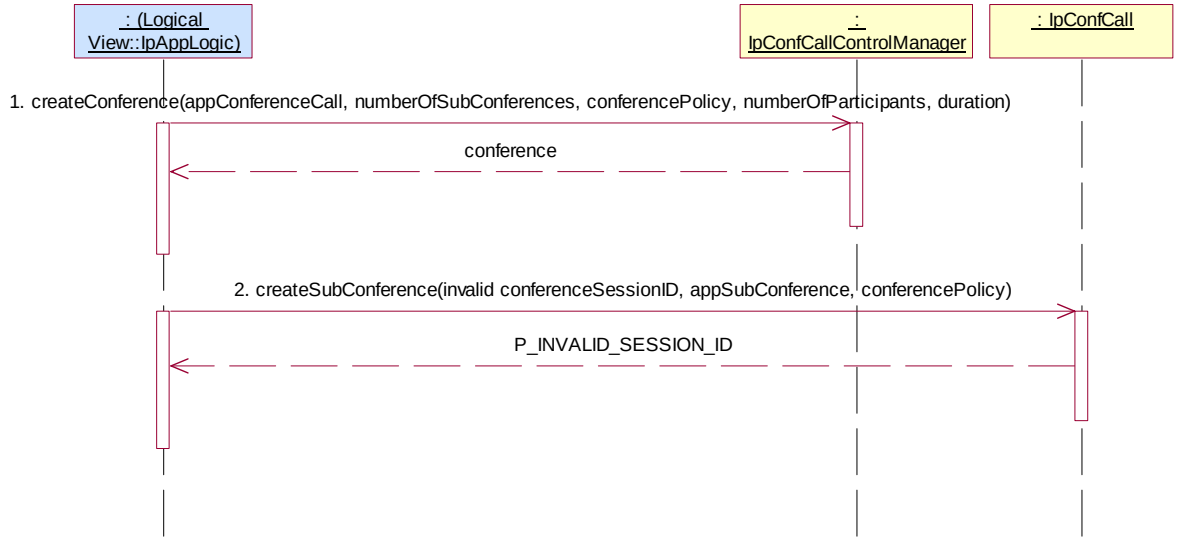
Reference: ES 202 915-4-5 [5], clauses 6.1 and 6.3

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 is returned



Test CCC _ IpConfCall _05

Summary: IpConfCall, deassignCall, P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.3

Preamble: Same as CCC _ IpConfCall _02

Test Sequence:

1. Method call **deassignCall()** on IpConfCall
   Parameters: invalid callSessionID
   Check: P_INVALID_SESSION_ID is returned

Test CCC _ IpConfCall _06

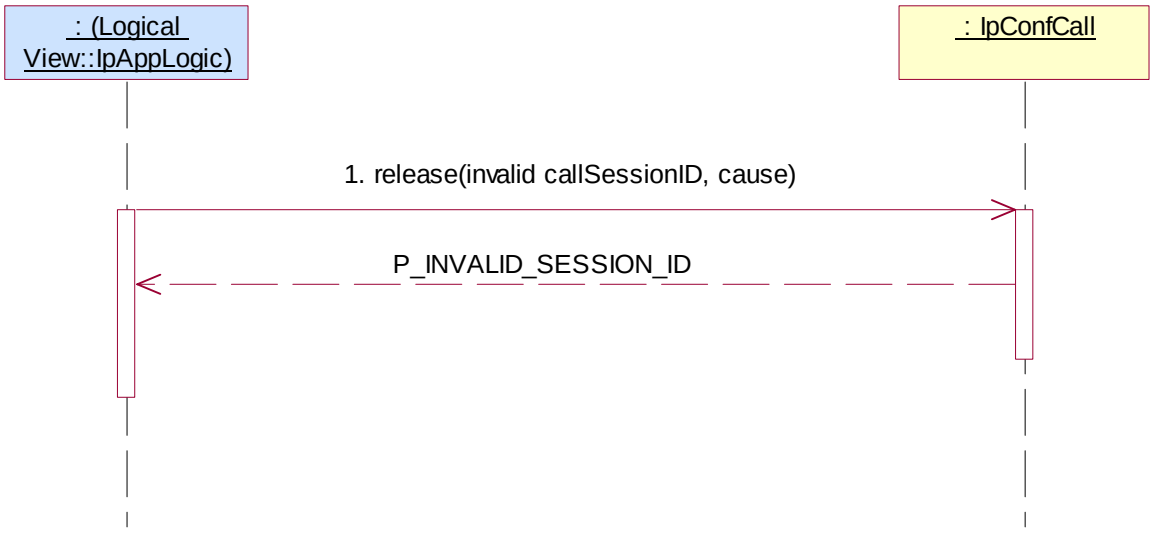
Summary: IpConfCall, release, P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.3

Preamble: Same as CCC _ IpConfCall _02

Test Sequence:

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



5.2.4.2.3 Optional, valid behaviour

Test CCC _ IpConfCall _07

Summary: IpConfCall, all methods, successful

Reference: ES 202 915-4-5 [5], clauses 6.1 and 6.3

Preamble: Application has a reference interface used for callbacks.

Condition: leaveMonitorReq method is supported.

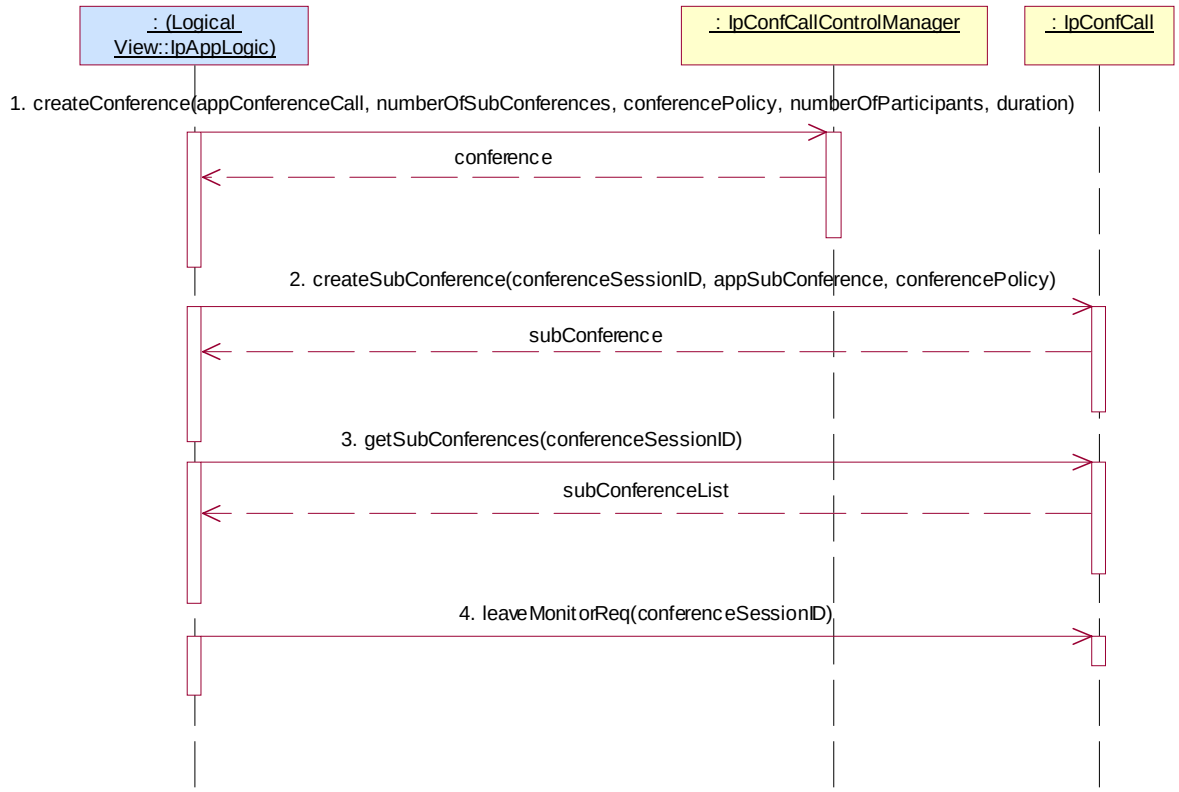
Test Sequence:

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 202 915-4-5 [5], clauses 6.1 and 6.3

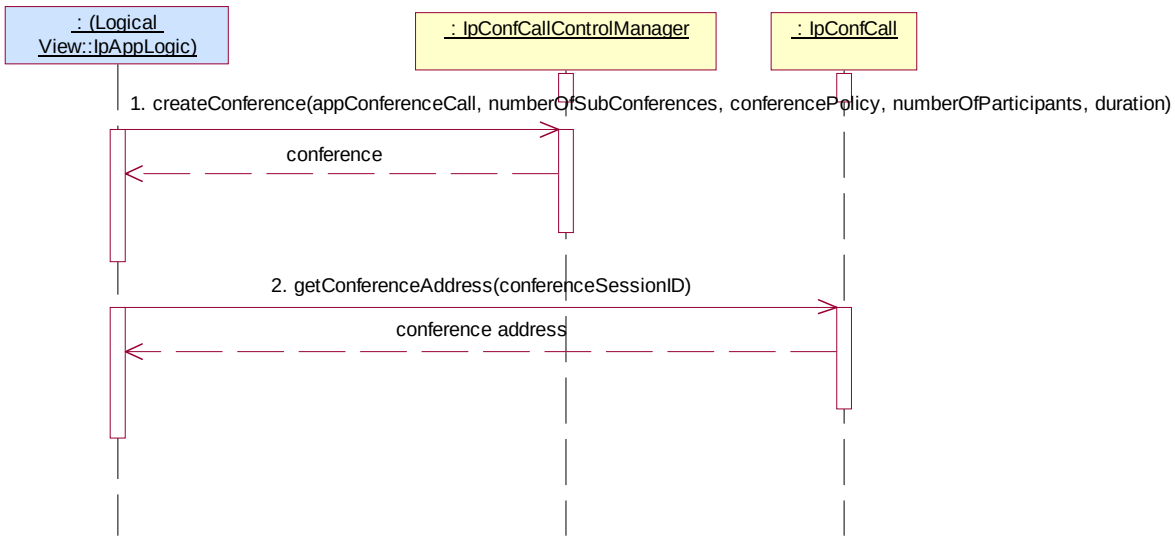
Preamble: Application has a reference interface used for callbacks.

Condition: getConferenceAddress 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 **getConferenceAddress()** on IpConfCall
   Parameters: valid conferenceSessionID returned in 1.
   Check: valid value of TpAddress is returned



Test CCC _ IpConfCall _09

Summary: IpConfCall, all methods, successful

Reference: ES 202 915-4-5 [5], clauses 6.1, 6.3 and 6.5 and ES 202 915-4-3 [3], clause 6.3

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

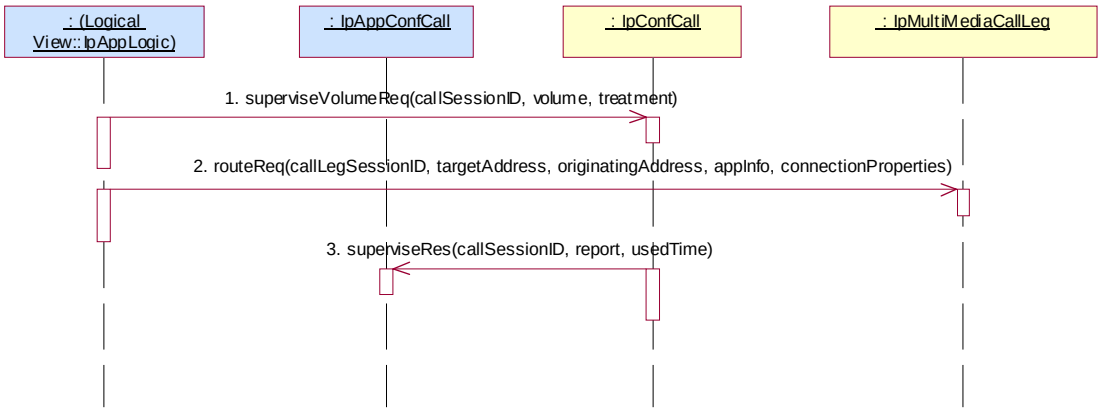
Condition: superviseVolumeReq method is supported.

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



Test CCC _ IpConfCall _10

Summary: IpConfCall, getInfoReq, successful

Reference: ES 202 915-4-3 [3], clauses 6.3 and 6.5

Preamble: Same as CCC _ IpConfCall _08

Condition: createCallLeg and getInfoReq methods are supported.

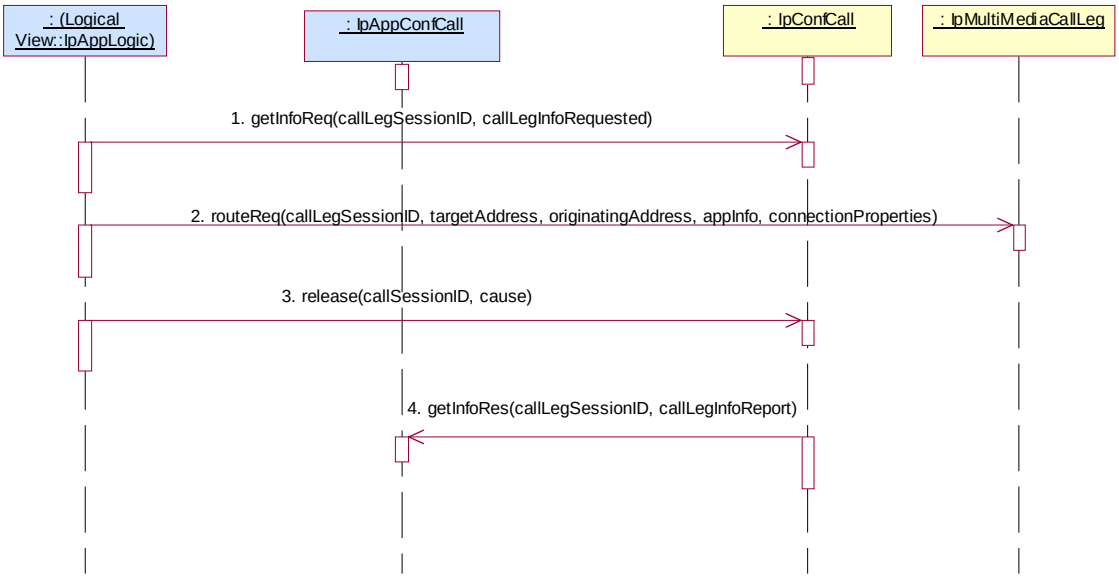
Test Sequence:

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

3. Method call **release()** on IpConfCall
   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) **IpAppConfCall** interface.
   Parameters: callSessionID given in 1., valid callInfoReport.



Test CCC _ IpConfCall _11

Summary: IpConfCall, setChargePlan, successful

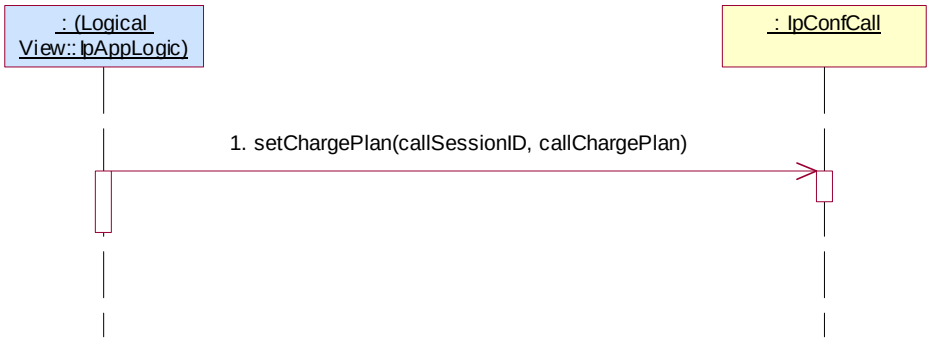
Reference: ES 202 915-4-3 [3], clause 6.3

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 _12

Summary: IpConfCall, setAdviceOfCharge, successful

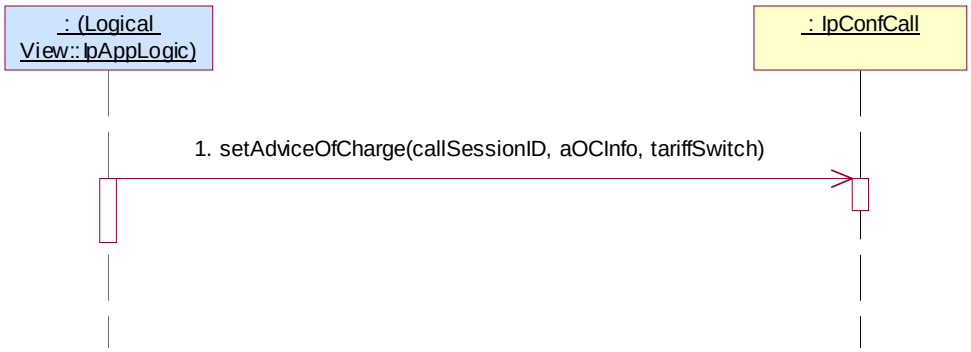
Reference: ES 202 915-4-3 [3], clause 6.3

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 _13

Summary: IpConfCall, superviseReq, successful

Reference: ES 202 915-4-3 [3], clauses 6.3 and 6.5

Preamble: Same as CCC _ IpConfCall _08

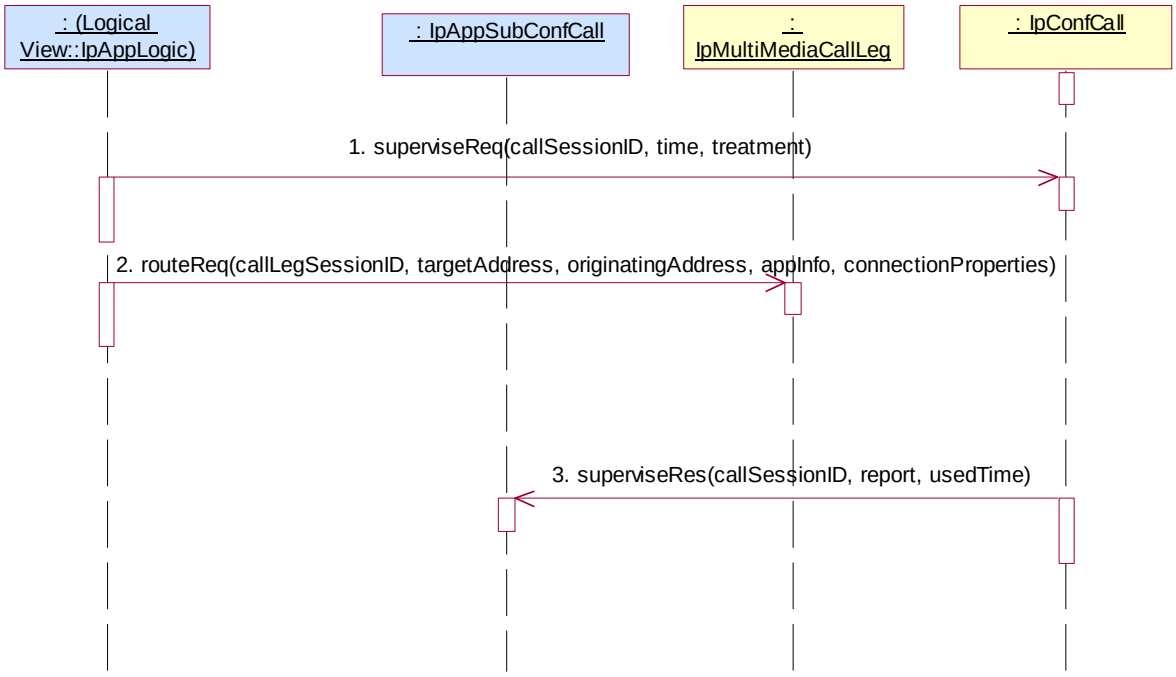
Condition: createCallLeg and superviseReq methods are supported.

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 _14

Summary: IpConfCall, all methods, successful

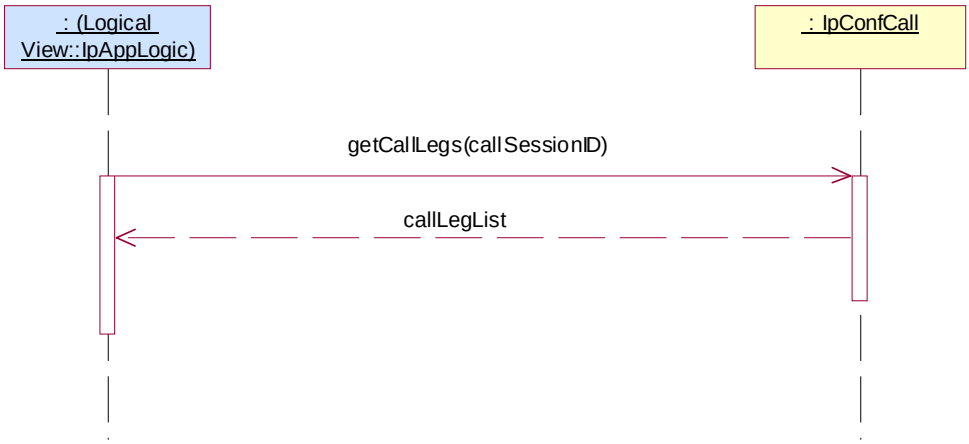
Reference: ES 202 915-4-3 [3], clause 6.3

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 _15

Summary: IpConfCall, getSubConferences, P_INVALID_SESSION_ID

Reference: ES 202 915-4-5 [5], clauses 6.1 and 6.3

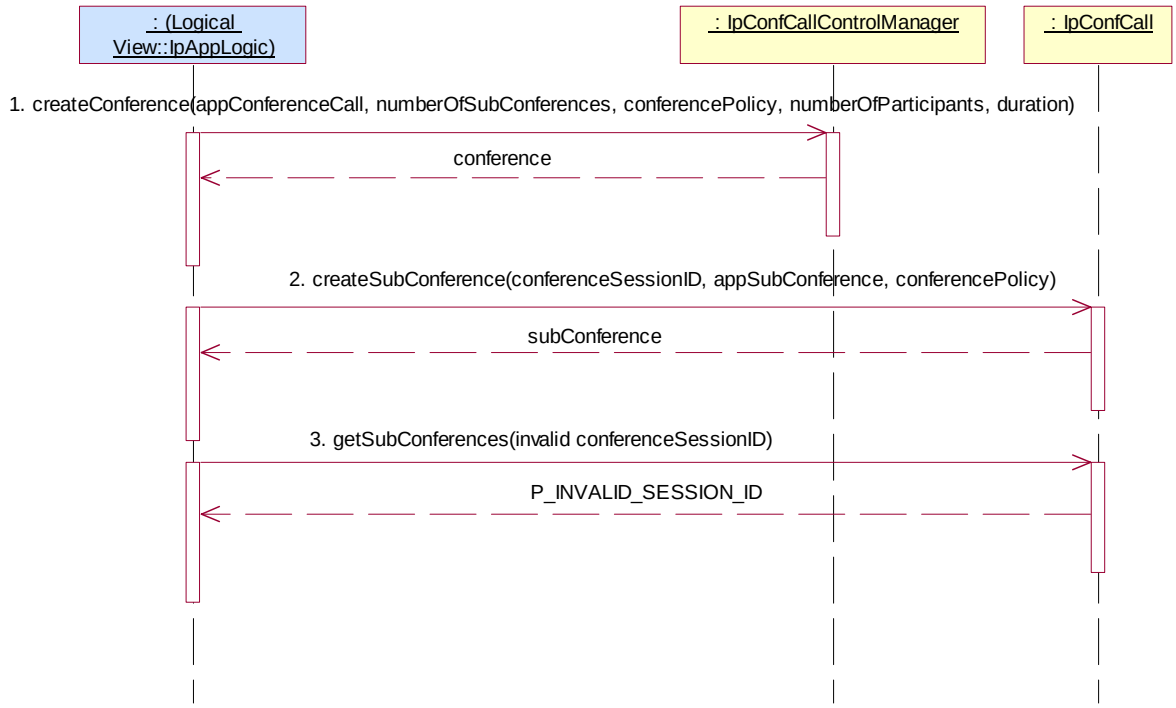
Preamble: Application has a reference interface used for callbacks.

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 is returned



Test CCC _ IpConfCall _16

Summary: IpConfCall, getConferenceAddress, P_INVALID_SESSION_ID

Reference: ES 202 915-4-5 [5], clauses 6.1 and 6.3

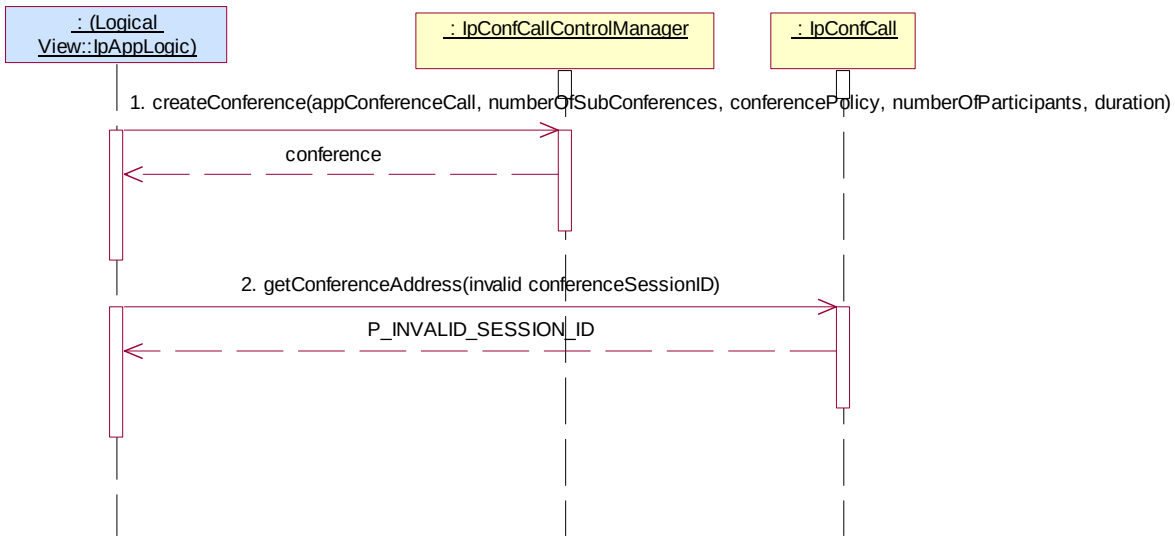
Preamble: Application has a reference interface used for callbacks.

Condition: getConferenceAddress 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 **getConferenceAddress()** on IpConfCall
   Parameters: invalid conferenceSessionID
   Check: P_INVALID_SESSION_ID is returned



Test CCC _ IpConfCall _17

Summary: IpConfCall, leaveMonitorReq, P_INVALID_SESSION_ID

Reference: ES 202 915-4-5 [5], clauses 6.1 and 6.3

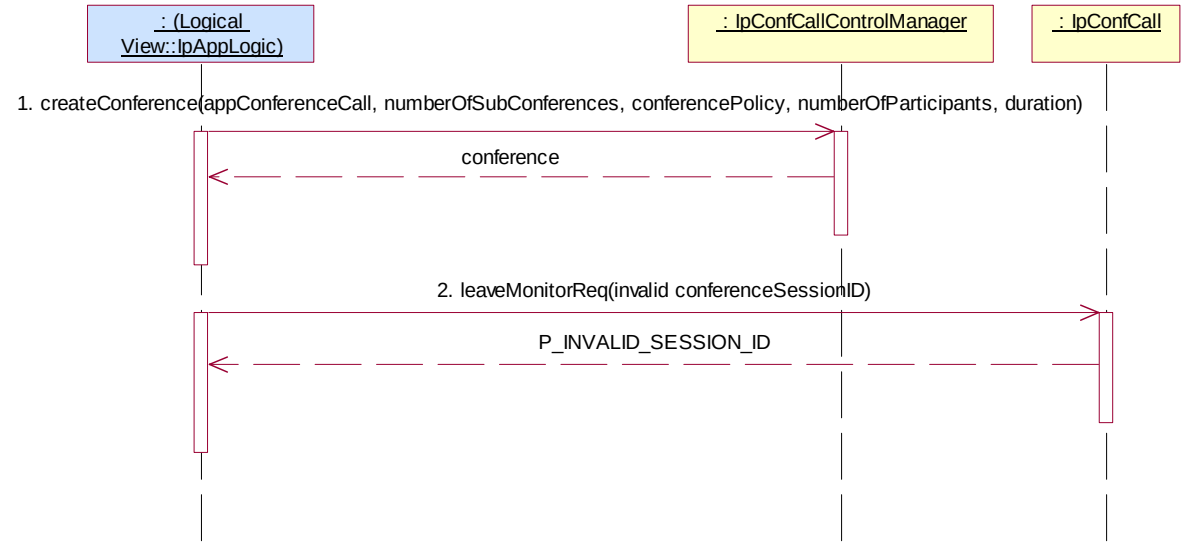
Preamble: Application has a reference interface used for callbacks.

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 is returned



Test CCC _ IpConfCall _18

Summary: IpConfCall, superviseVolumeReq, P_INVALID_SESSION_ID

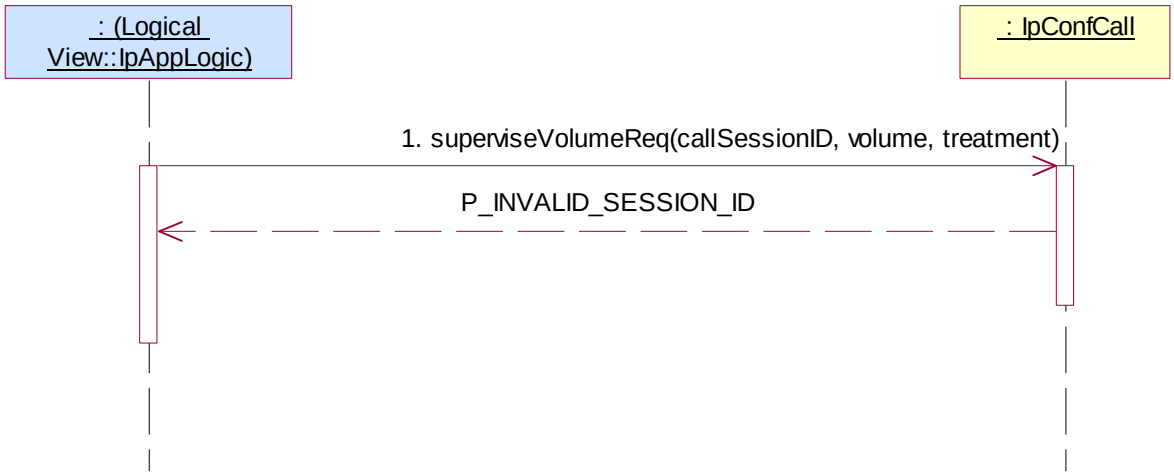
Reference: ES 202 915-4-3 [3], clause 6.3

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 is returned



Test CCC _ IpConfCall _19

Summary: IpConfCall, getInfoReq, P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.3

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 is returned

Test CCC _ IpConfCall _20

Summary: IpConfCall, setChargePlan, P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.3

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 is returned

Test CCC _ IpConfCall _21

Summary: IpConfCall, setAdviceOfCharge, P_INVALID_SESSION_ID

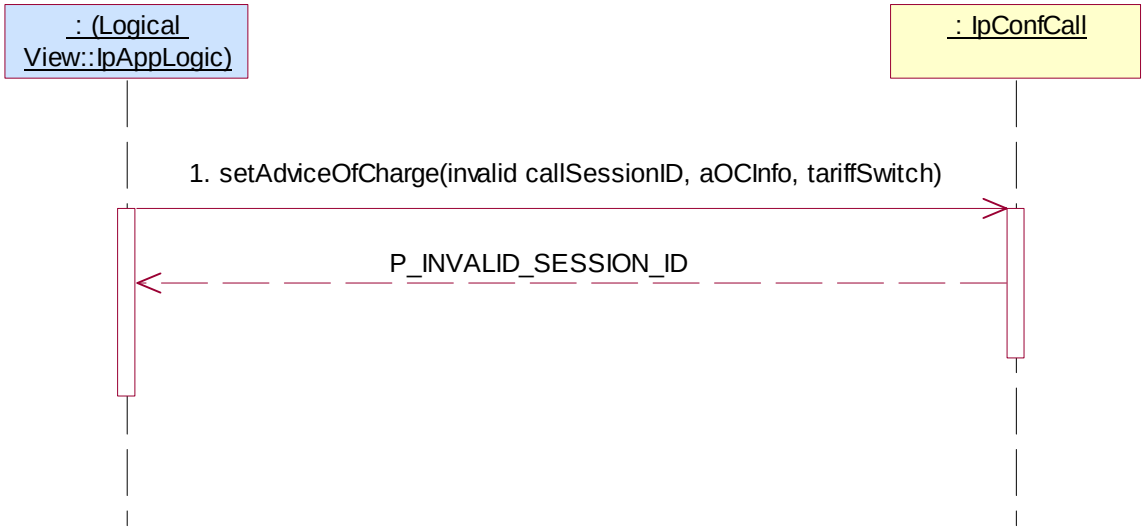
Reference: ES 202 915-4-3 [3], clause 6.3

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 is returned



Test CCC _ IpConfCall _22

Summary: IpConfCall, setAdviceOfCharge, P_INVALID_CURRENCY

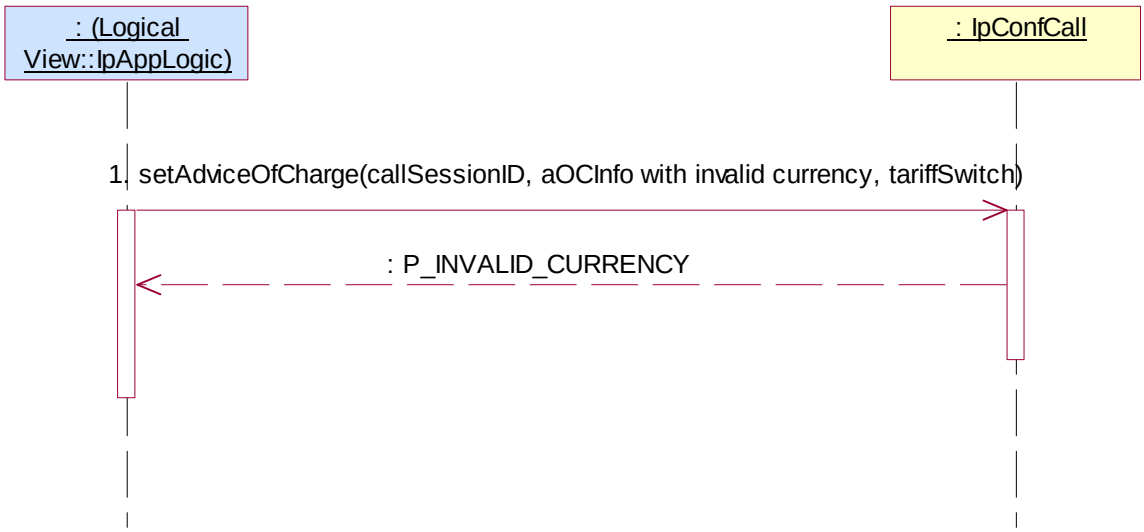
Reference: ES 202 915-4-3 [3], clause 6.3

Preamble: Same as CCC _ IpConfCall _02

Condition: setAdviceOfCharge method is supported.

Test Sequence:

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



Test CCC _ IpConfCall _23

Summary: IpConfCall, setAdviceOfCharge, P_INVALID_AMOUNT

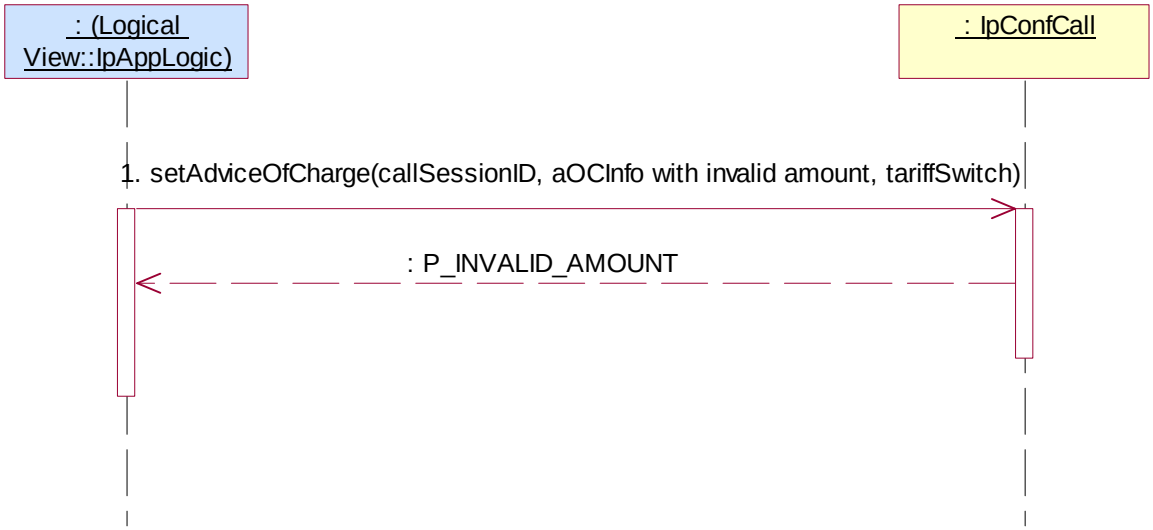
Reference: ES 202 915-4-3 [3], clause 6.3

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 is returned



Test CCC _ IpConfCall _24

Summary: IpConfCall, superviseReq, P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.3

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 is returned

Test CCC _ IpConfCall _25

Summary: IpConfCall, getCallLegs, P_INVALID_SESSION_ID

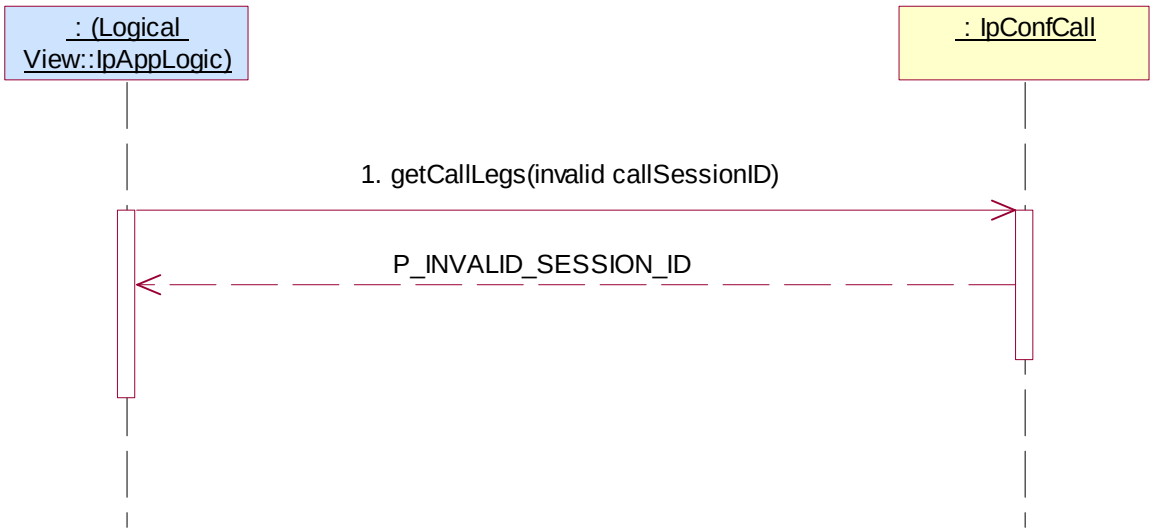
Reference: ES 202 915-4-3 [3], clause 6.3

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 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 202 915-4-5 [5], clauses 6.1, 6.3 and 6.5 and ES 202 915-4-3 [3], clause 6.3

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

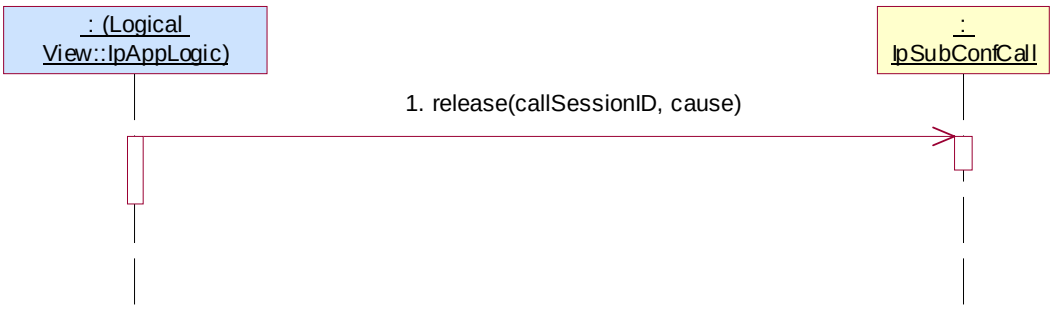
5. Method call **routeReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 3., valid targetAddress, valid appInfo, valid connectionProperties
   Check: no exception is returned

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

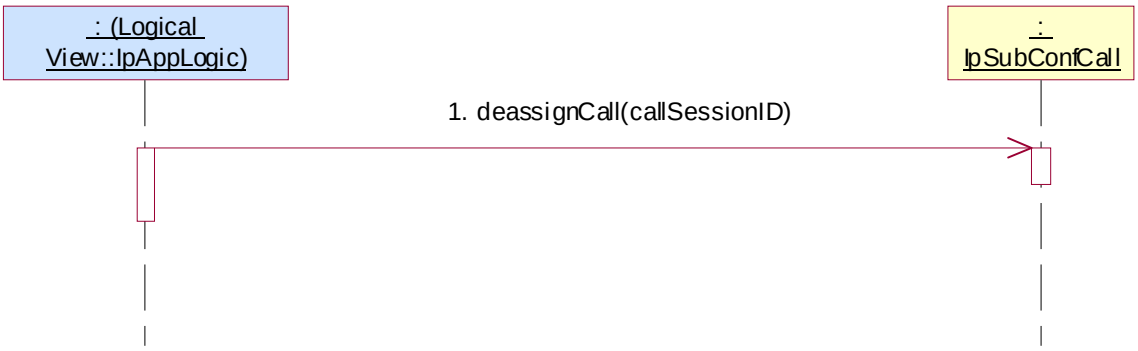
Summary: IpSubConfCall, all methods mandatory, successful

Reference: ES 202 915-4-3 [3], clause 6.3

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

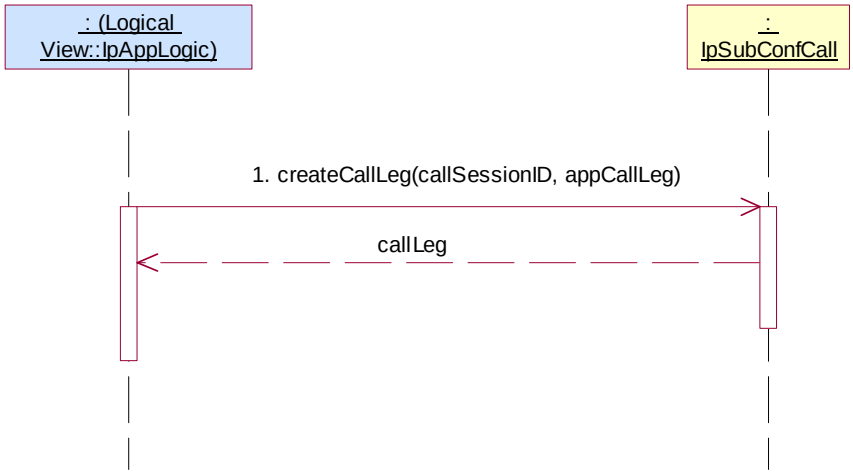
Reference: ES 202 915-4-3 [3], clause 6.3

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 202 915-4-3 [3], clause 6.3

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 202 915-4-5 [5], clauses 6.1, 6.3 and 6.5 and ES 202 915-4-3 [3], clause 6.5

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 with Interrupt event
   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 with Interrupt event, 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, eventInfo

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 with Interrupt event
   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 with Interrupt event, 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, eventInfo

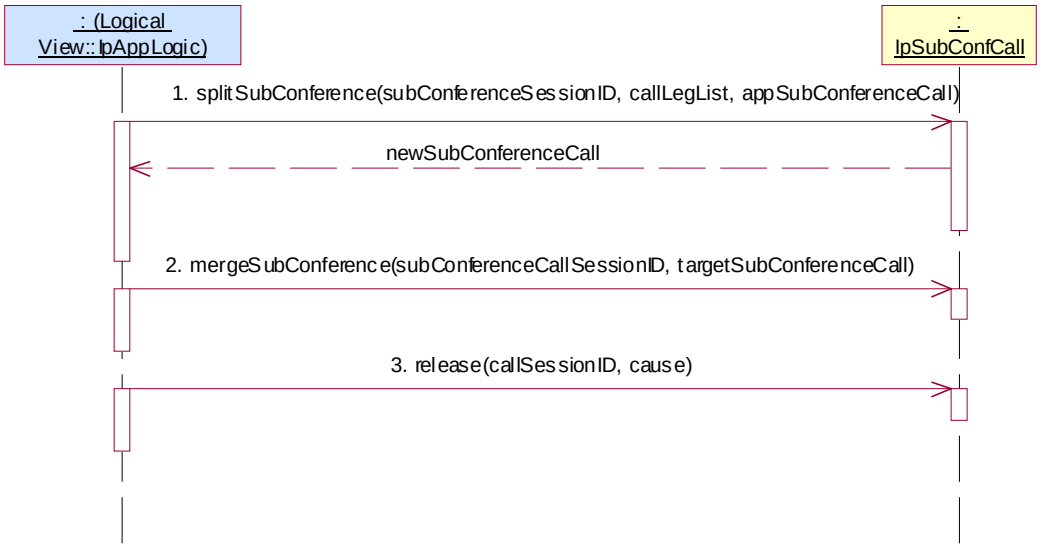
Condition: splitSubConference method is supported.

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 202 915-4-5 [5], clause 6.5

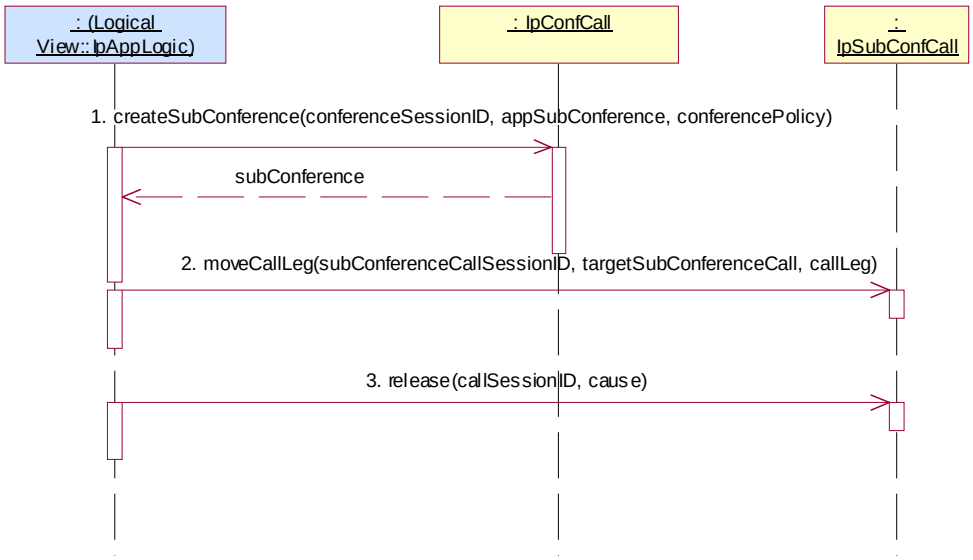
Preamble: Same as CCC _ IpSubConfCall _05

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

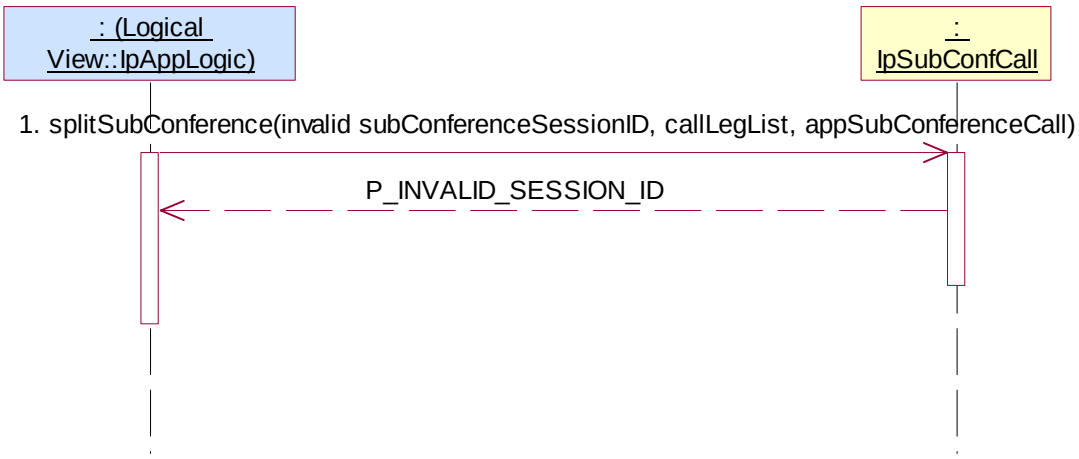
Summary: IpSubConfCall, splitSubConference, P_INVALID_SESSION_ID

Reference: ES 202 915-4-5 [5], clause 6.5

Preamble: Same as CCC _ IpSubConfCall _05

Test Sequence:

1. Method call **splitSubConference()**
   Parameters: invalid subConferenceSessionID, valid callLegList, valid appSubConferenceCall
   Check: P_INVALID_SESSION_ID is returned.



Test CCC _ IpSubConfCall _08

Summary: IpSubConfCall, mergeSubConference, P_INVALID_SESSION_ID

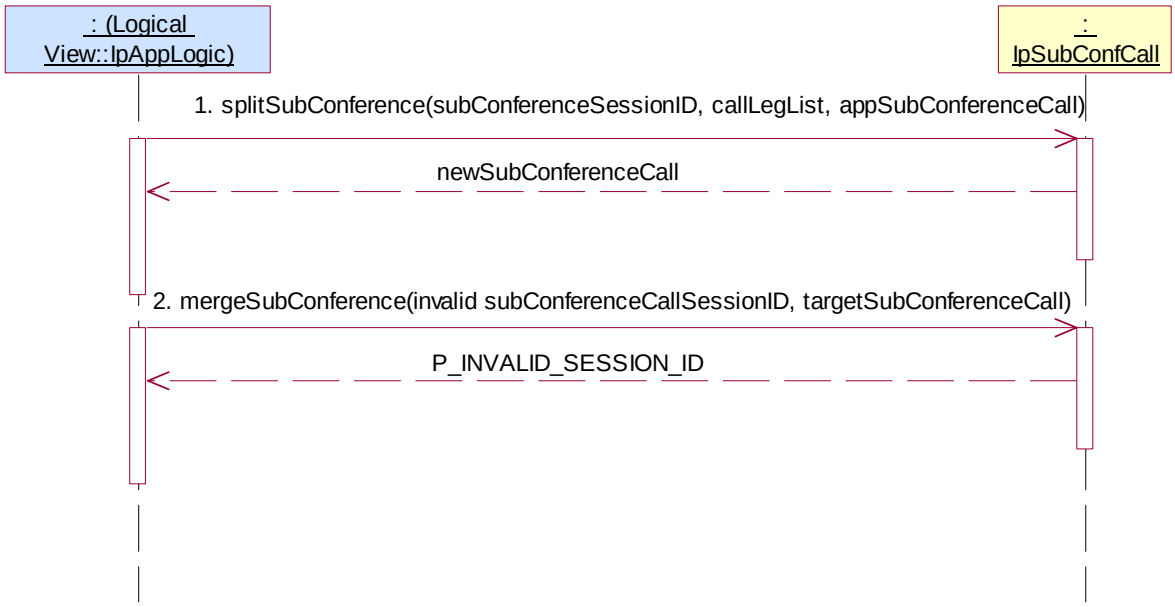
Reference: ES 202 915-4-5 [5], clause 6.5

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 is returned.



Test CCC _ IpSubConfCall _09

Summary: IpSubConfCall, mergeSubConference, P_INVALID_SESSION_ID

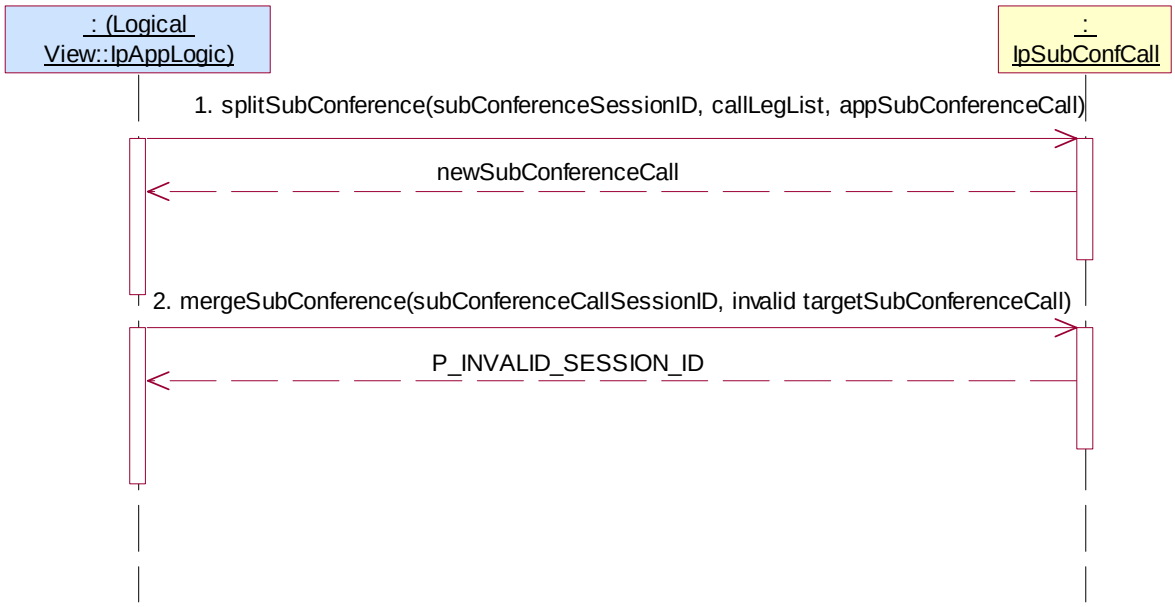
Reference: ES 202 915-4-5 [5], clause 6.5

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 is returned.



Test CCC _ IpSubConfCall _10

Summary: IpSubConfCall, moveCallLeg, P_INVALID_SESSION_ID

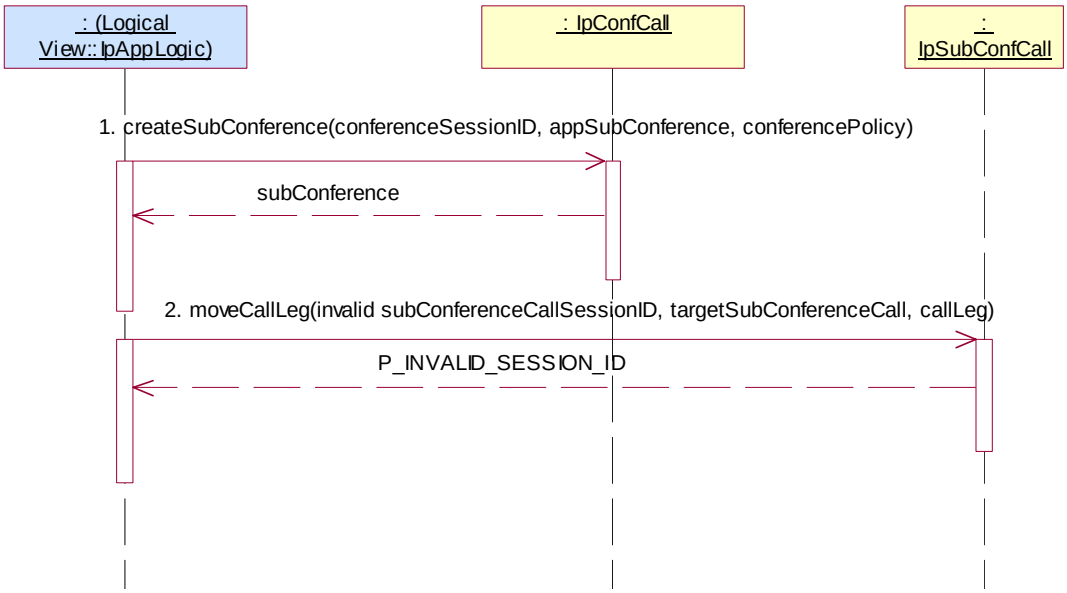
Reference: ES 202 915-4-5 [5], clauses 6.3 and 6.5

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 is returned.



Test CCC _ IpSubConfCall _11

Summary: IpSubConfCall, moveCallLeg, P_INVALID_SESSION_ID

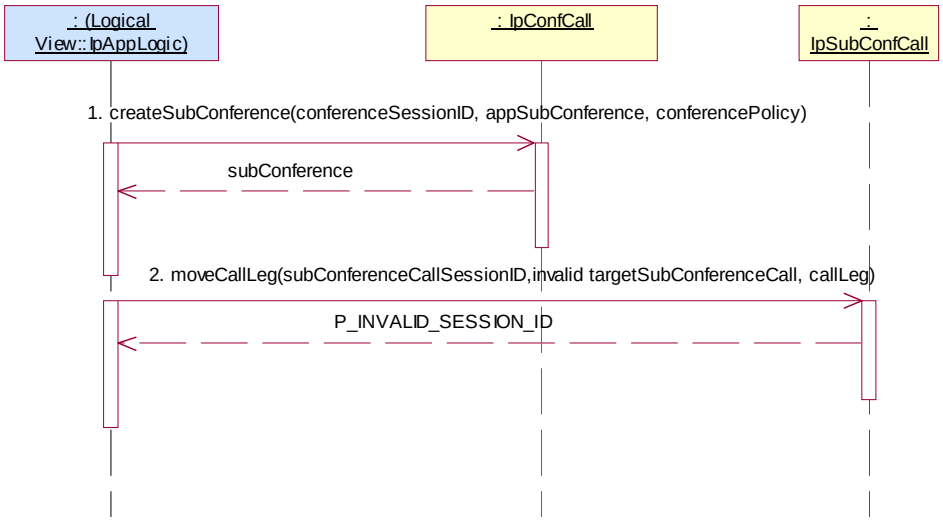
Reference: ES 202 915-4-5 [5], clauses 6.3 and 6.5

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 is returned.



Test CCC _ IpSubConfCall _12

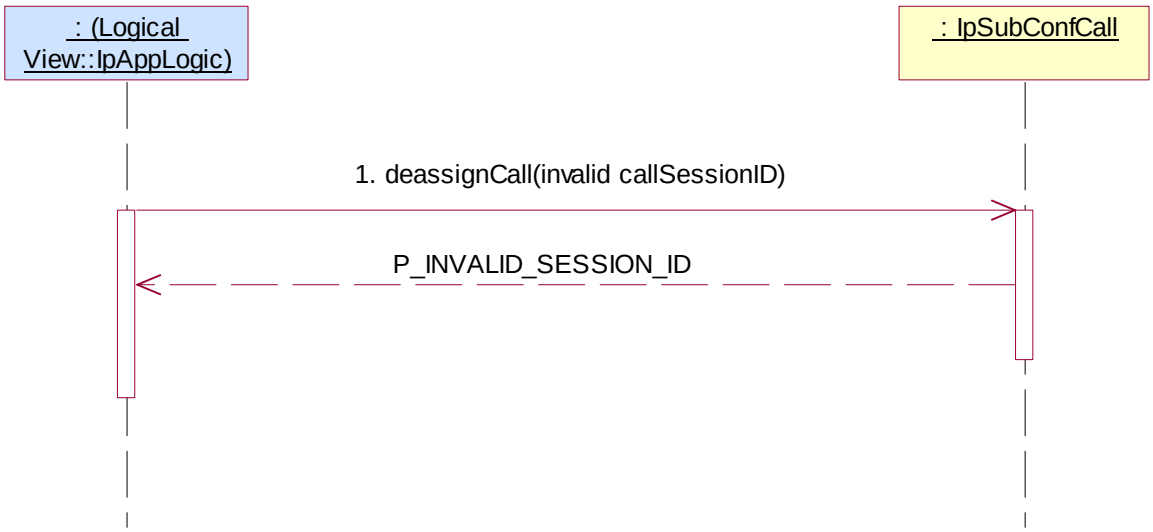
Summary: IpSubConfCall, deassignCall, P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.3

Preamble: Same as CCC _ IpSubConfCall _01

Test Sequence:

1. Method call **deassignCall()** on IpSubConfCall
   Parameters: invalid callSessionID
   Check: P_INVALID_SESSION_ID is returned



Test CCC _ IpSubConfCall _13

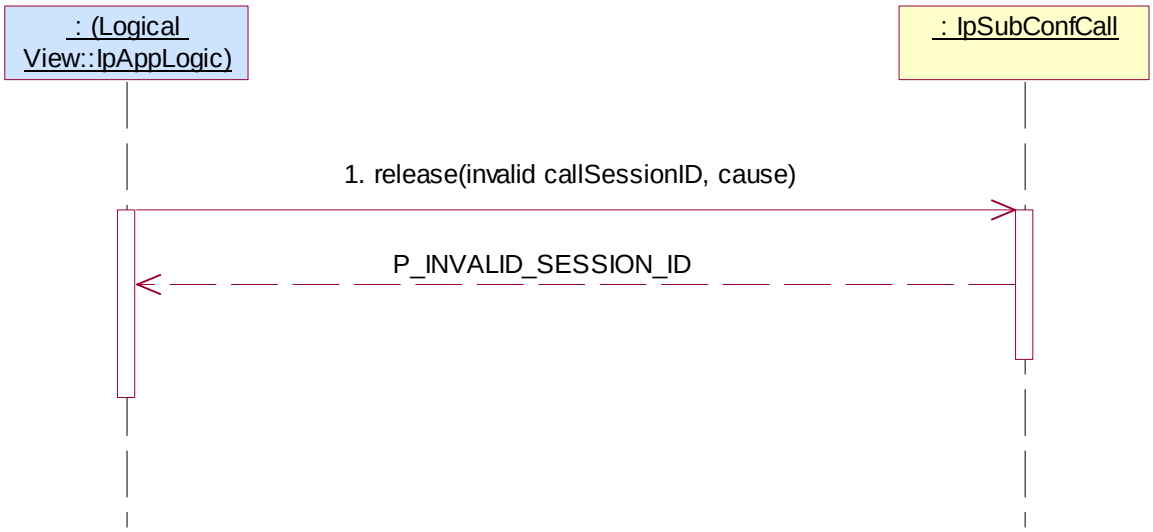
Summary: IpSubConfCall, release, P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.3

Preamble: Same as CCC _ IpSubConfCall _01

Test Sequence:

1. Method call **release()** on IpSubConfCall
   Parameters: invalid callSessionID, valid cause
   Check: P_INVALID_SESSION_ID is returned



Test CCC _ IpSubConfCall _14

Summary: IpSubConfCall, createCallLeg, P_INVALID_SESSION_ID

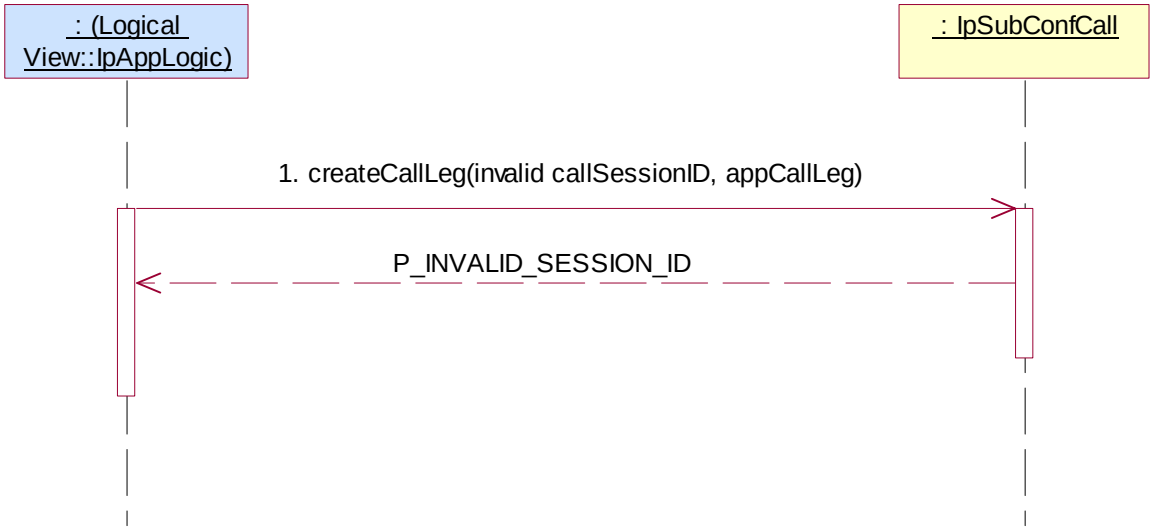
Reference: ES 202 915-4-3 [3], clause 6.3

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 is returned



Test CCC _ IpSubConfCall _15

Summary: IpSubConfCall, createCallLeg, P_INVALID_INTERFACE_TYPE

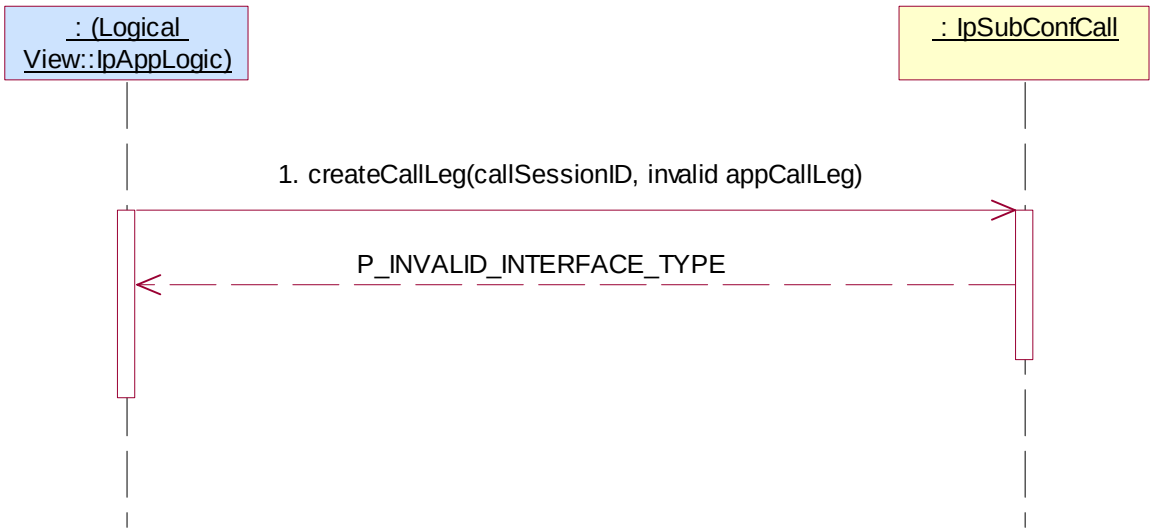
Reference: ES 202 915-4-3 [3], clause 6.3

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 is returned



Test CCC _ IpSubConfCall _16

Summary: IpSubConfCall, createAndRouteCallLegReq, P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.3

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 is returned

Test CCC _ IpSubConfCall _17

Summary: IpSubConfCall, createAndRouteCallLegReq, P_INVALID_INTERFACE_TYPE

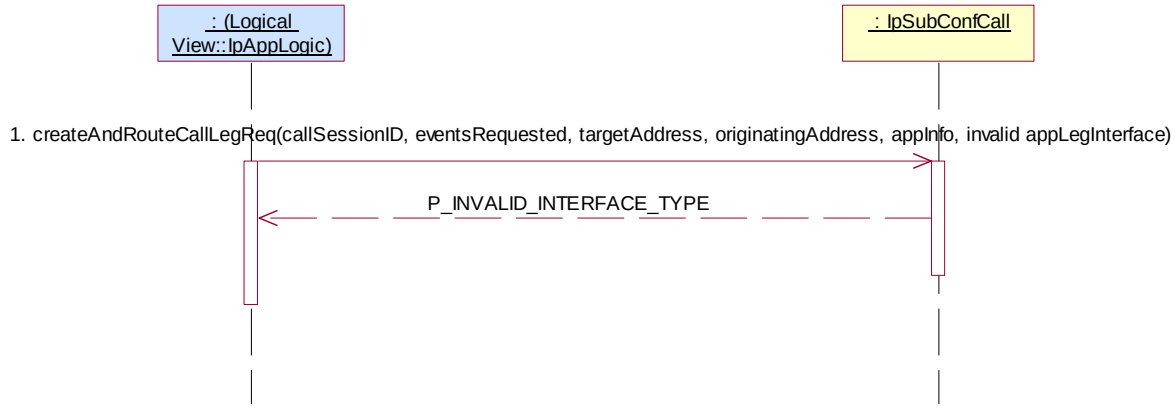
Reference: ES 202 915-4-3 [3], clause 6.3

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 is returned



Test CCC _ IpSubConfCall _18

Summary: IpSubConfCall, createAndRouteCallLegReq, P_INVALID_ADDRESS

Reference: ES 202 915-4-3 [3], clause 6.3

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 is returned

Test CCC _ IpSubConfCall _19

Summary: IpSubConfCall, createAndRouteCallLegReq, P_INVALID_ADDRESS

Reference: ES 202 915-4-3 [3], clause 6.3

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 is returned

Test CCC _ IpSubConfCall _20

Summary: IpSubConfCall, createAndRouteCallLegReq, P_INVALID_CRITERIA

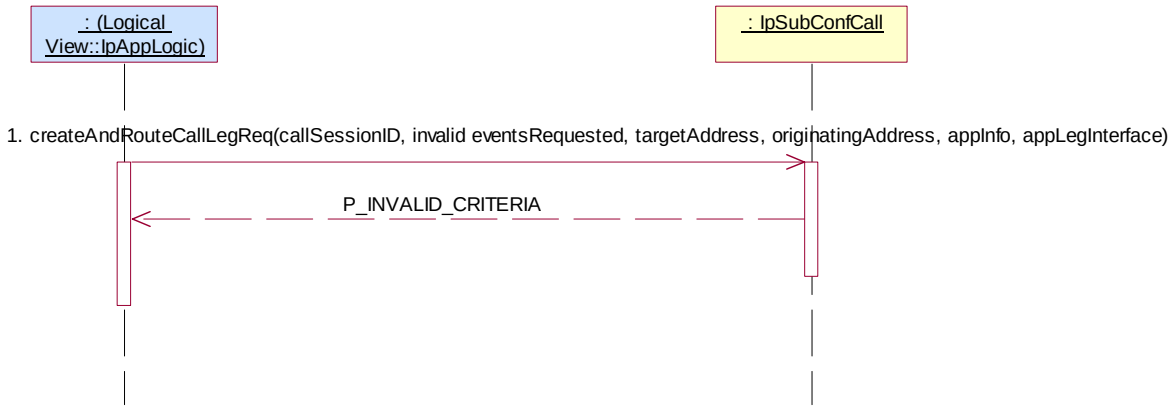
Reference: ES 202 915-4-3 [3], clause 6.3

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 is returned



5.2.4.3.3 Optional, valid behaviour

Test CCC _ IpSubConfCall _21

Summary: IpSubConfCall, all methods, successful

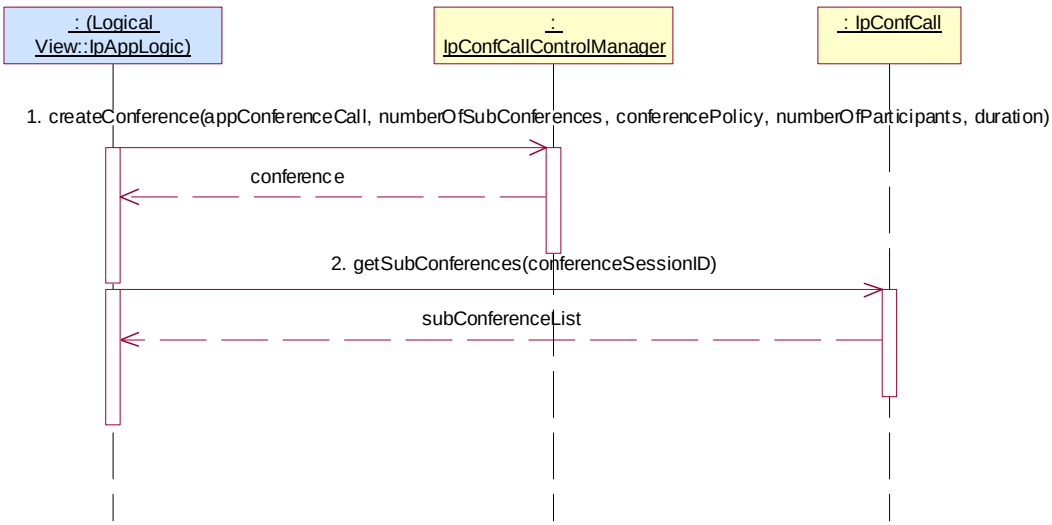
Reference: ES 202 915-4-5 [5], clauses 6.1 and 6.3

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

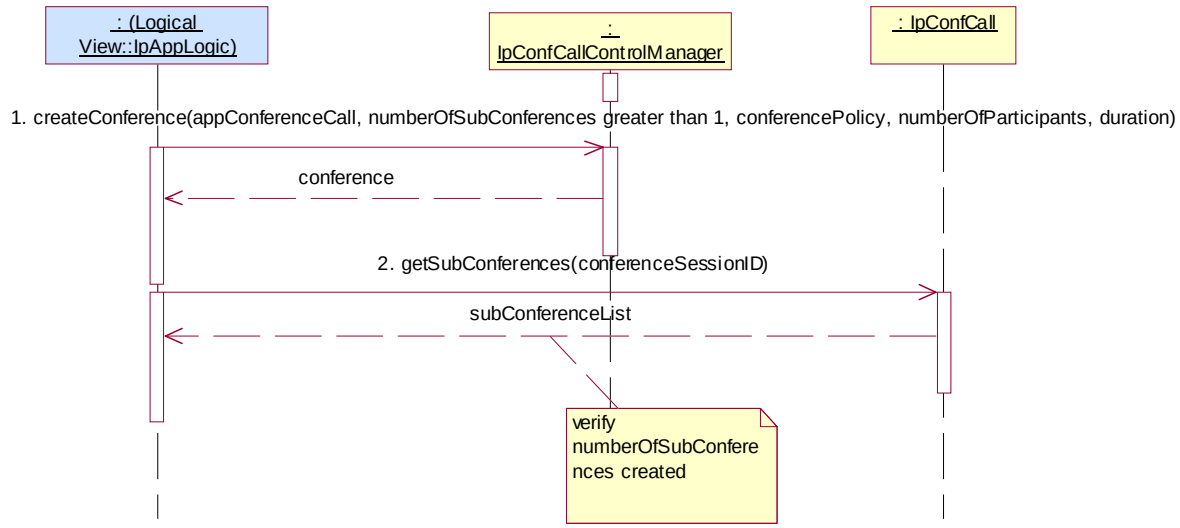
Reference: ES 202 915-4-5 [5], clauses 6.1 and 6.3

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 202 915-4-5 [5], clause 6.5

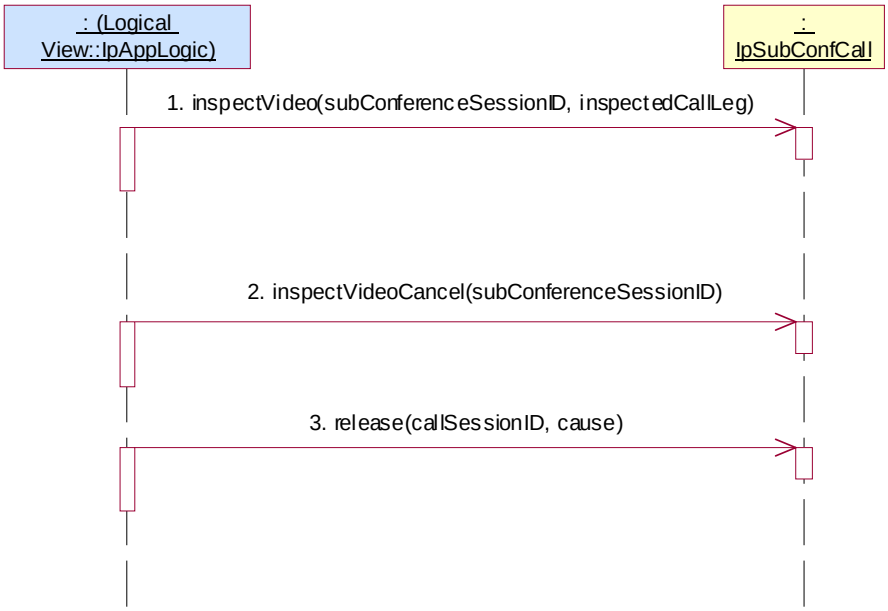
Preamble: Same as CCC _ IpSubConfCall _01

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 202 915-4-5 [5], clause 6.5 and ES 202 915-4-3 [3], clause 6.3

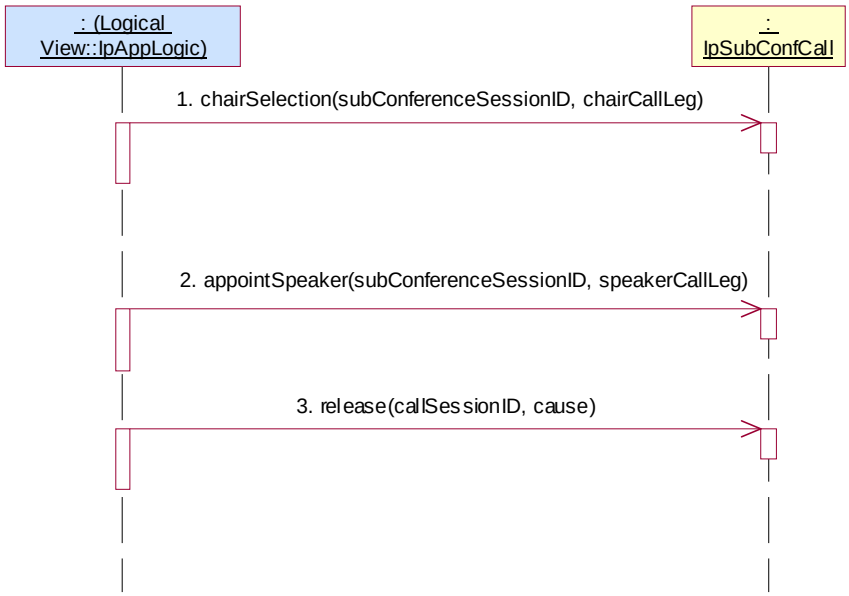
Preamble: Same as CCC _ IpSubConfCall _01

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

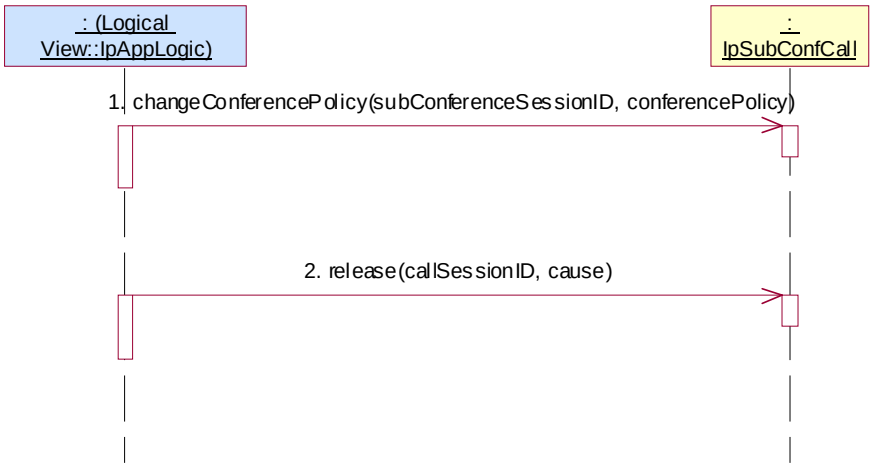
Reference: ES 202 915-4-5 [5], clause 6.5 and ES 202 915-4-3 [3], clause 6.3

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 202 915-4-3 [3], clause 6.3

Preamble: Same as CCC _ IpConfCall _08

Condition: createCallLeg and getInfoReq methods are supported.

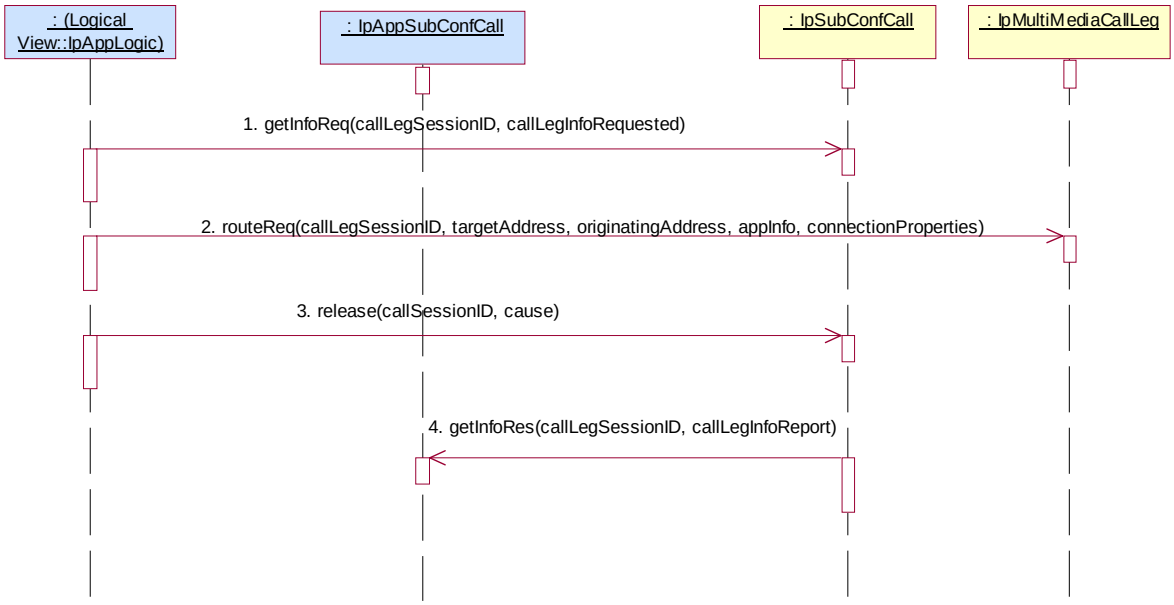
Test Sequence:

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

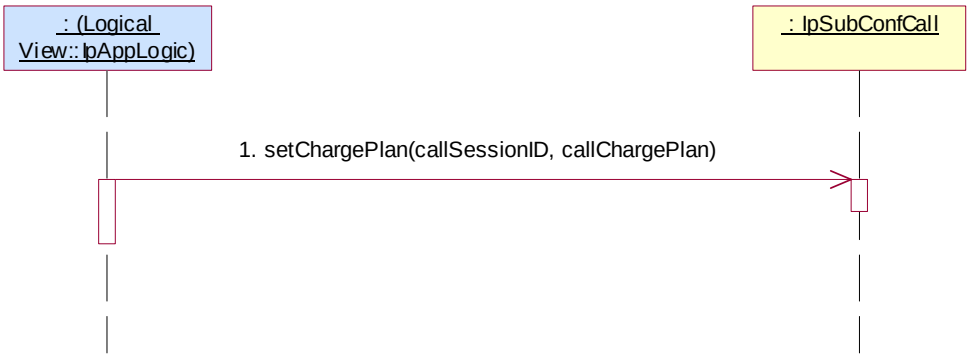
Reference: ES 202 915-4-3 [3], clause 6.3

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

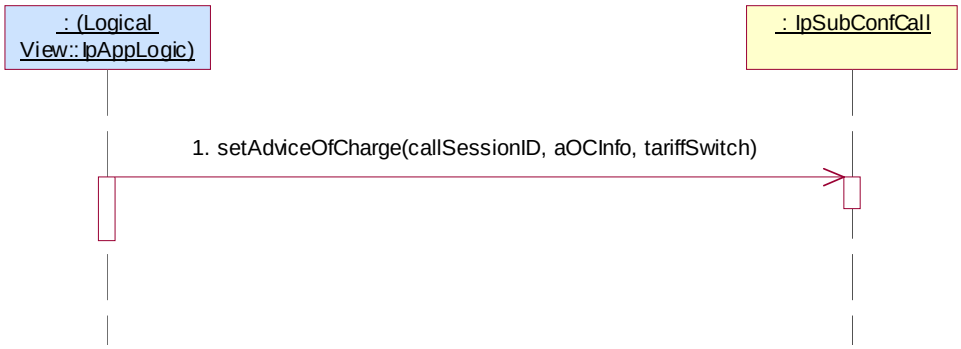
Reference: ES 202 915-4-3 [3], clause 6.3

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

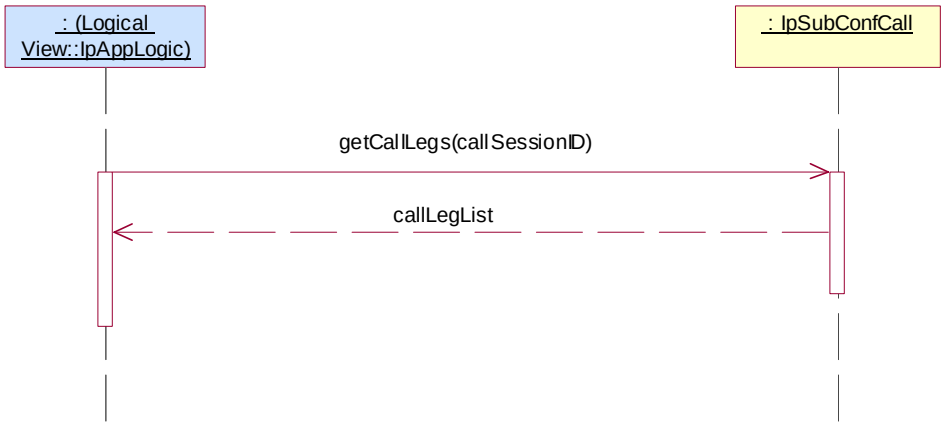
Reference: ES 202 915-4-3 [3], clause 6.3

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 202 915-4-5 [5], clause 6.5

Preamble: Same as CCC _ IpSubConfCall _01

Test Sequence:

1. Method call **inspectVideo()**
   Parameters: invalid subConferenceSessionID, valid inspectedCallLeg
   Check: P_INVALID_SESSION_ID is returned.

Test CCC _ IpSubConfCall _31

Summary: IpSubConfCall, inspectVideoCancel, P_INVALID_SESSION_ID

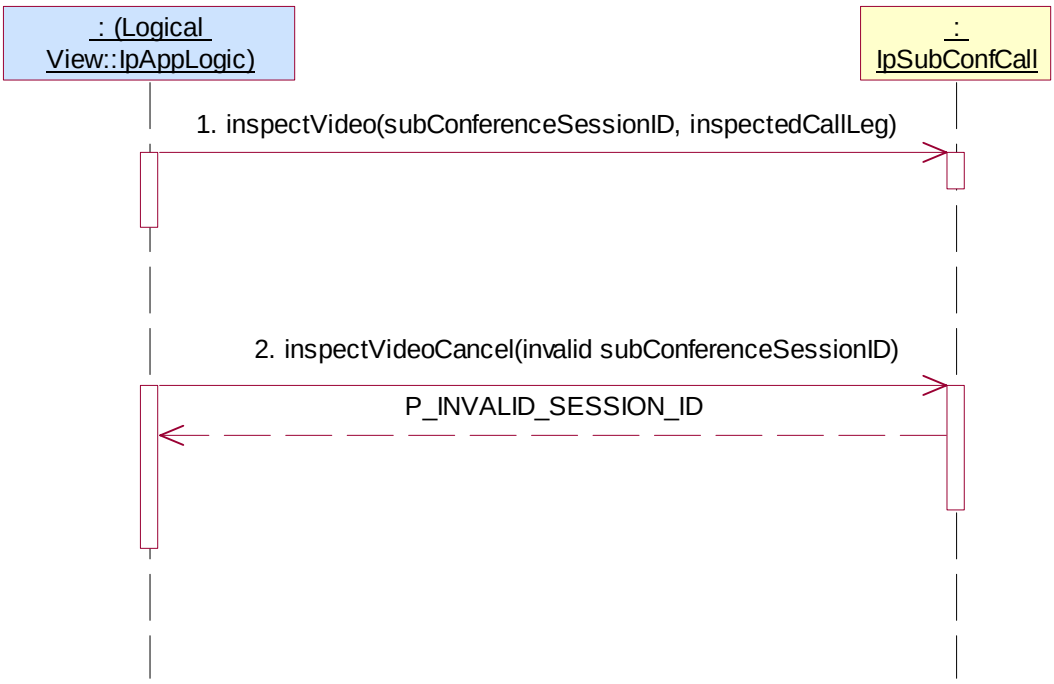
Reference: ES 202 915-4-5 [5], clause 6.5

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 is returned.



Test CCC _ IpSubConfCall _32

Summary: IpSubConfCall, appointSpeaker, P_INVALID_SESSION_ID

Reference: ES 202 915-4-5 [5], clause 6.5

Preamble: Same as CCC _ IpSubConfCall _01

Test Sequence:

1. Method call **appointSpeaker()**
   Parameters: invalid subConferenceSessionID, valid speakerCallLeg
   Check: P_INVALID_SESSION_ID is returned.

Test CCC _ IpSubConfCall _33

Summary: IpSubConfCall, chairSelection, P_INVALID_SESSION_ID

Reference: ES 202 915-4-5 [5], clause 6.5

Preamble: Same as CCC _ IpSubConfCall _01

Test Sequence:

1. Method call **chairSelection()**
   Parameters: invalid subConferenceSessionID, valid chairCallLeg
   Check: P_INVALID_SESSION_ID is returned.

Test CCC _ IpSubConfCall _34

Summary: IpSubConfCall, changeConferencePolicy, P_INVALID_SESSION_ID

Reference: ES 202 915-4-5 [5], clause 6.5

Preamble: Same as CCC _ IpSubConfCall _01

Test Sequence:

1. Method call **changeConferencePolicy()**
   Parameters: invalid subConferenceSessionID, valid conferencePolicy
   Check: P_INVALID_SESSION_ID is returned.

Test CCC _ IpSubConfCall _35

Summary: IpSubConfCall, getInfoReq, P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.3

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 is returned

Test CCC _ IpSubConfCall _36

Summary: IpSubConfCall, setChargePlan, P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.3

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 is returned

Test CCC _ IpSubConfCall _37

Summary: IpSubConfCall, setAdviceOfCharge, P_INVALID_SESSION_ID

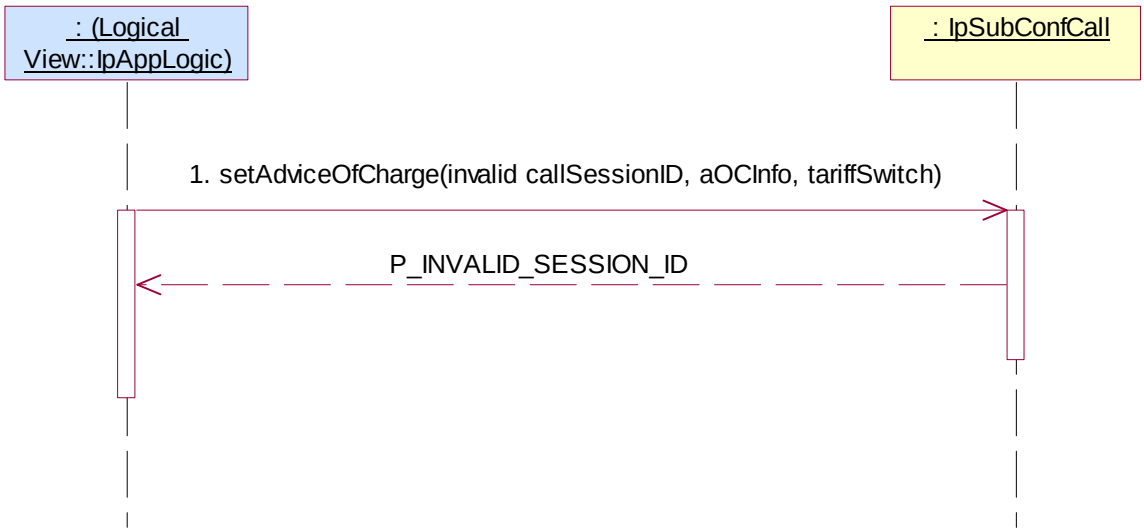
Reference: ES 202 915-4-3 [3], clause 6.3

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 is returned



Test CCC _ IpSubConfCall _38

Summary: IpSubConfCall, setAdviceOfCharge, P_INVALID_CURRENCY

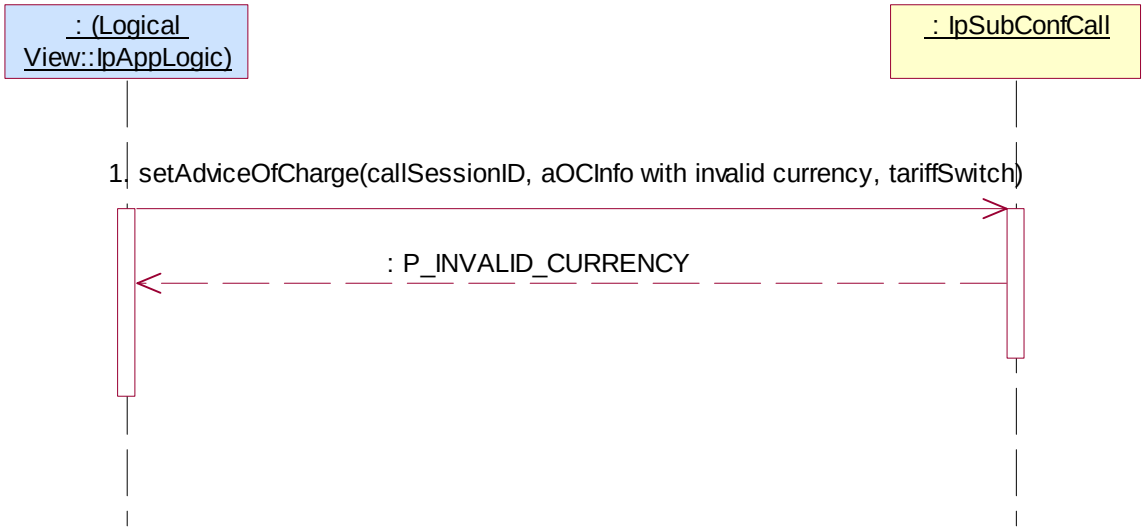
Reference: ES 202 915-4-3 [3], clause 6.3

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 is returned



Test CCC _ IpSubConfCall _39

Summary: IpSubConfCall, setAdviceOfCharge, P_INVALID_AMOUNT

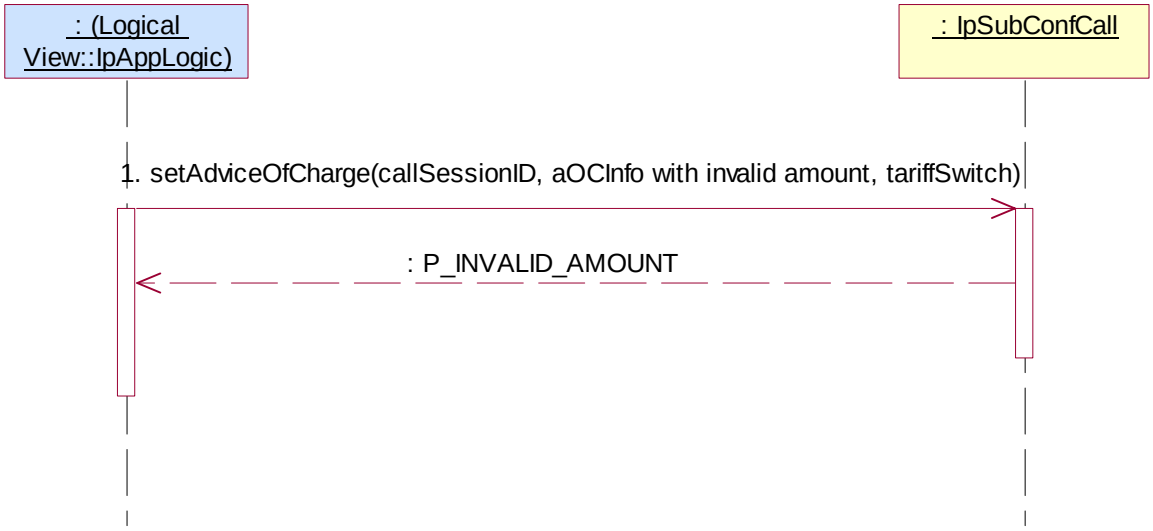
Reference: ES 202 915-4-3 [3], clause 6.3

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 is returned



Test CCC _ IpSubConfCall _40

Summary: IpSubConfCall, getCallLegs, P_INVALID_SESSION_ID

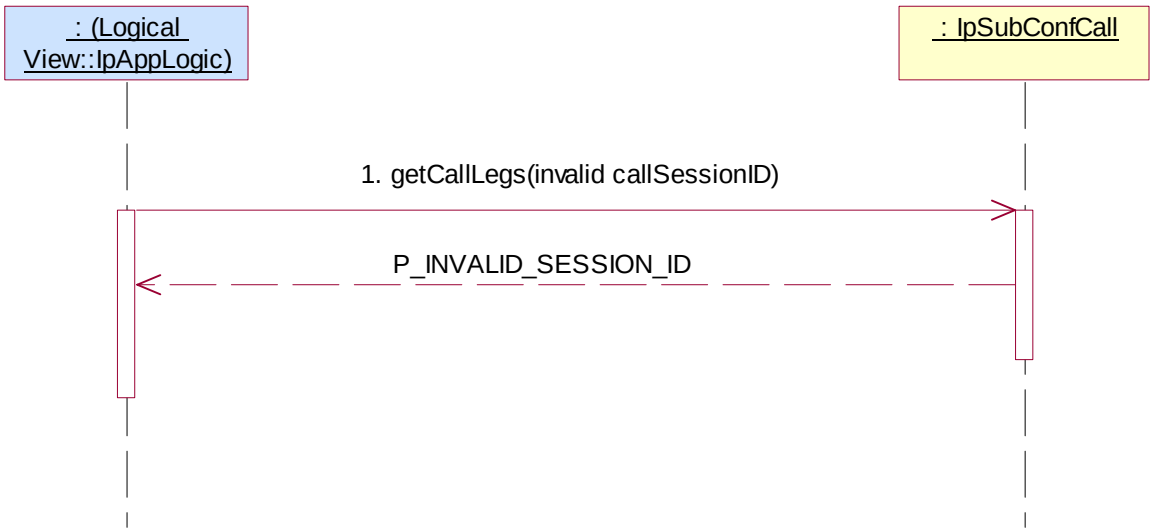
Reference: ES 202 915-4-3 [3], clause 6.3

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 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 202 915-4-4 [4], clause 6.5

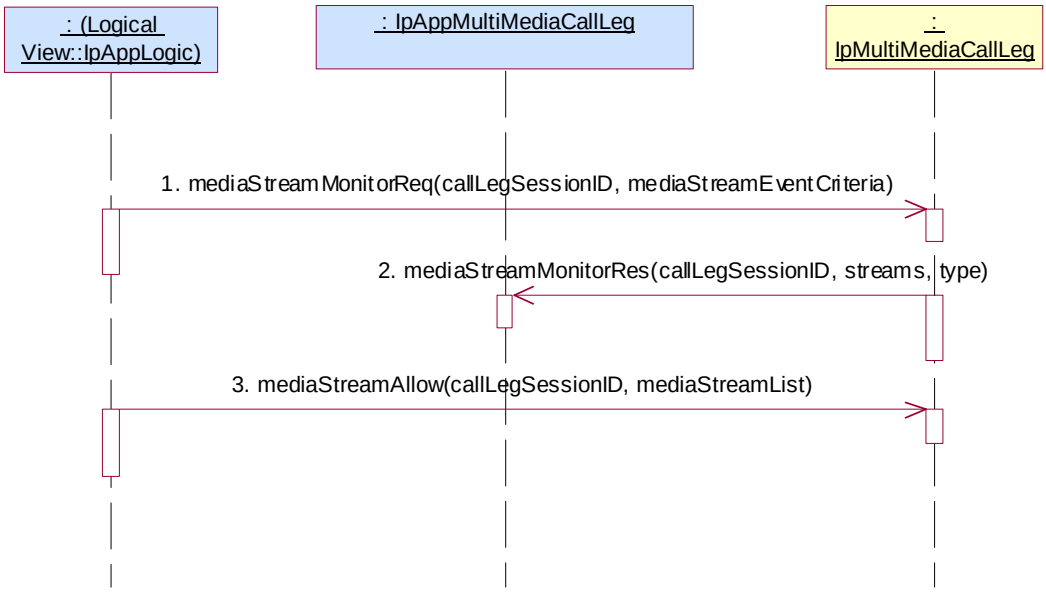
Preamble: Same as CCC _ IpSubConfCall_01

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 202 915-4-3 [3], clause 6.5

Preamble: Same as CCC _ IpConfCall _08

Condition: createCallLeg method is supported

Test Sequence:

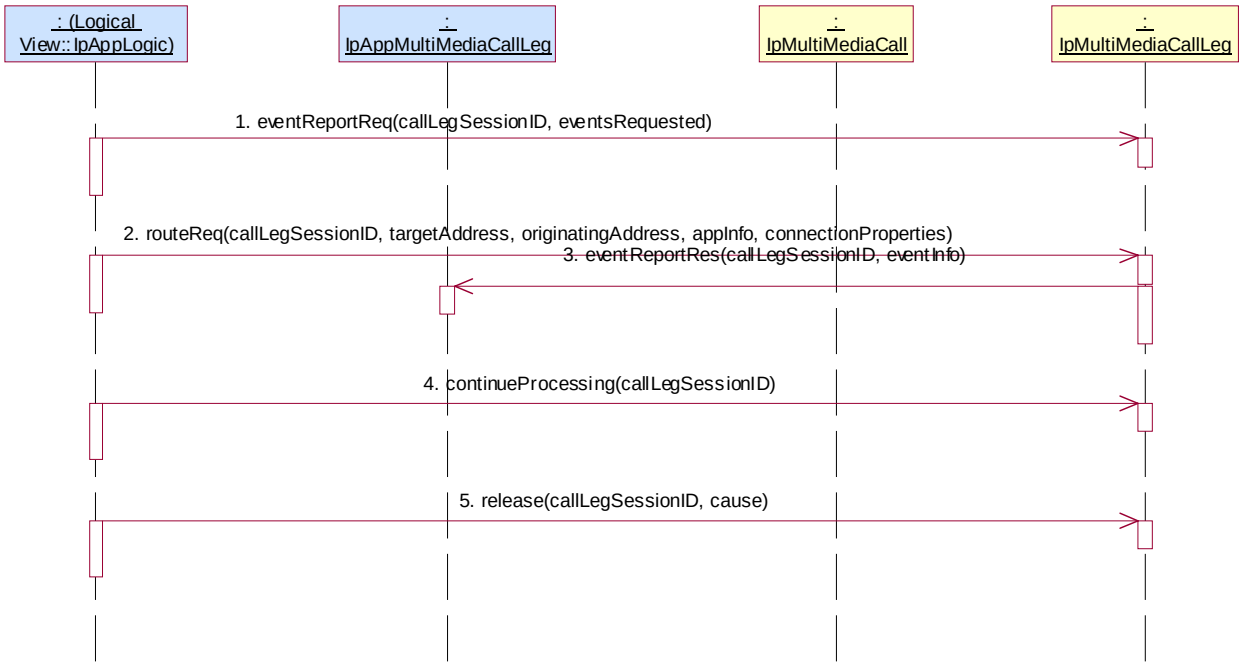
1. Method call **eventReportReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 1., valid eventsRequested with Interrupt event
   Check: no exception is returned

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, eventInfo

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 202 915-4-3 [3], clause 6.5 and ES 202 915-4-5 [5], clause 6.5

Preamble: Same as CCC_IpConfCall_02

Condition: createAndRouteCallLeg method is supported

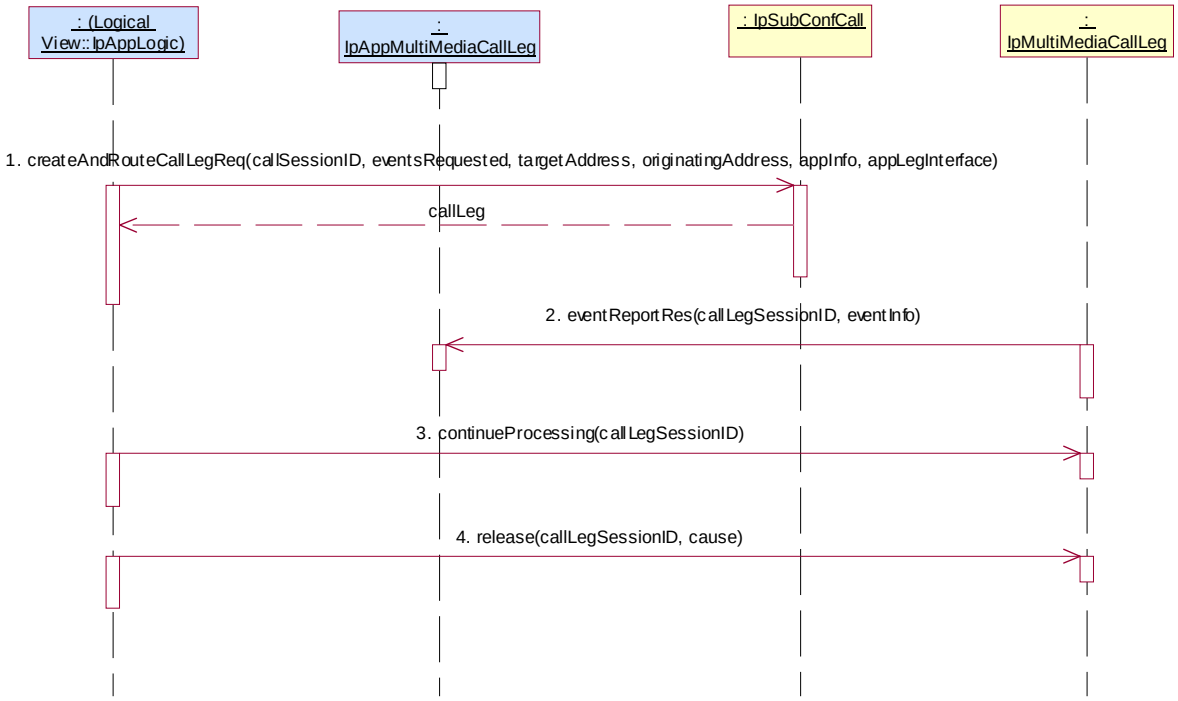
Test Sequence:

1. Method call **createAndRouteCallLeg()** on IpSubConfCall
   Parameters: valid callSessionID returned in preamble, valid eventsRequested with Interrupt event, 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., eventInfo

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 202 915-4-3 [3], clause 6.5

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 202 915-4-3 [3], clause 6.5 and ES 202 915-4-5 [5], clause 6.5

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 with Interrupt event
   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, eventInfo

5. Method call **continueProcessing()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID 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

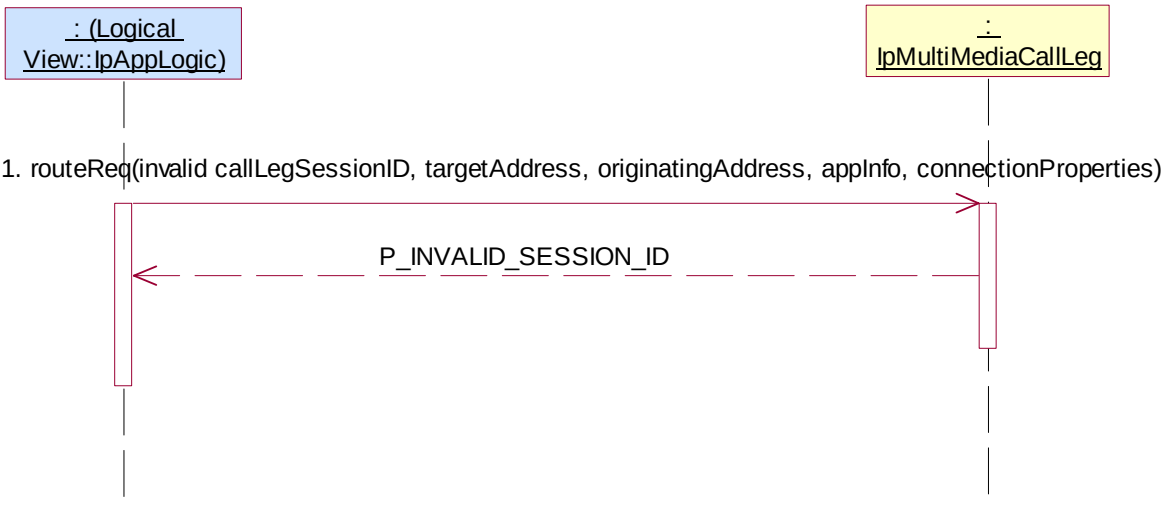
Summary: IpMultiMediaCallLeg, routeReq: P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.5

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 is returned



Test CCC _ IpMultiMediaCallLeg _07

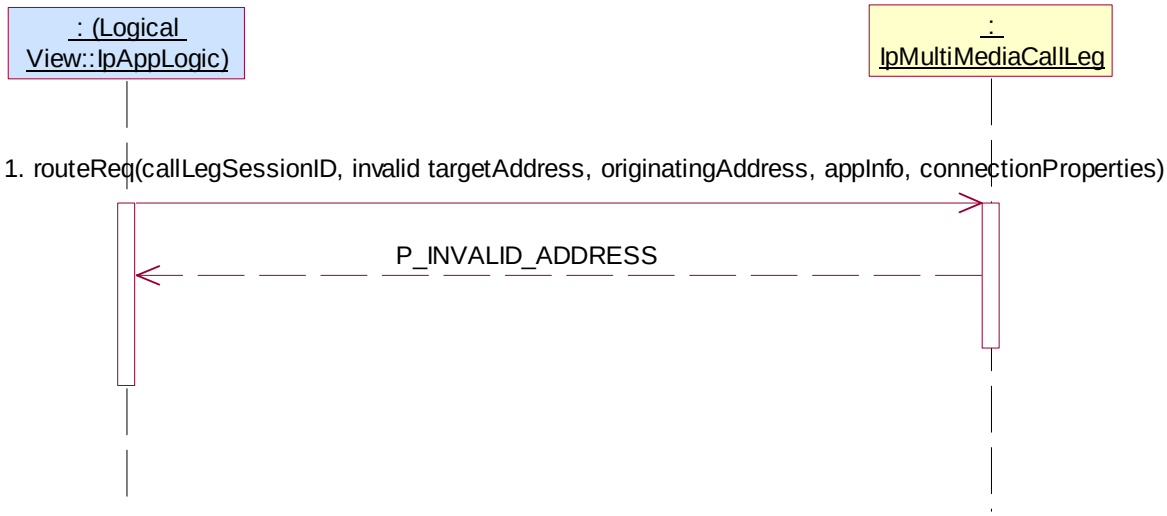
Summary: IpMultiMediaCallLeg, routeReq, P_INVALID_ADDRESS

Reference: ES 202 915-4-3 [3], clause 6.5

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 is returned



Test CCC _ IpMultiMediaCallLeg _08

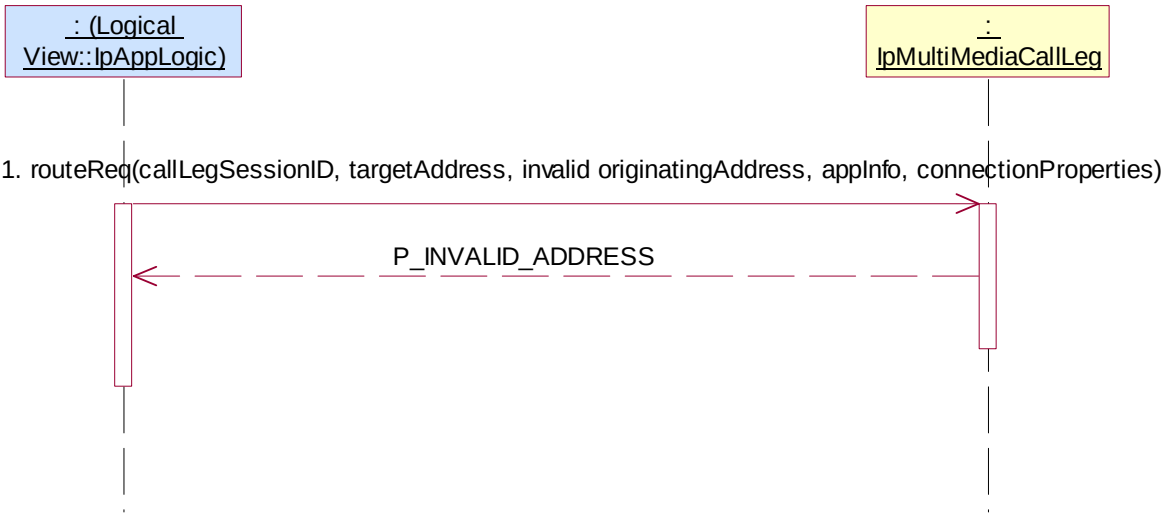
Summary: IpMultiMediaCallLeg, routeReq, P_INVALID_ADDRESS

Reference: ES 202 915-4-3 [3], clause 6.5

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 is returned



Test CCC _ IpMultiMediaCallLeg _09

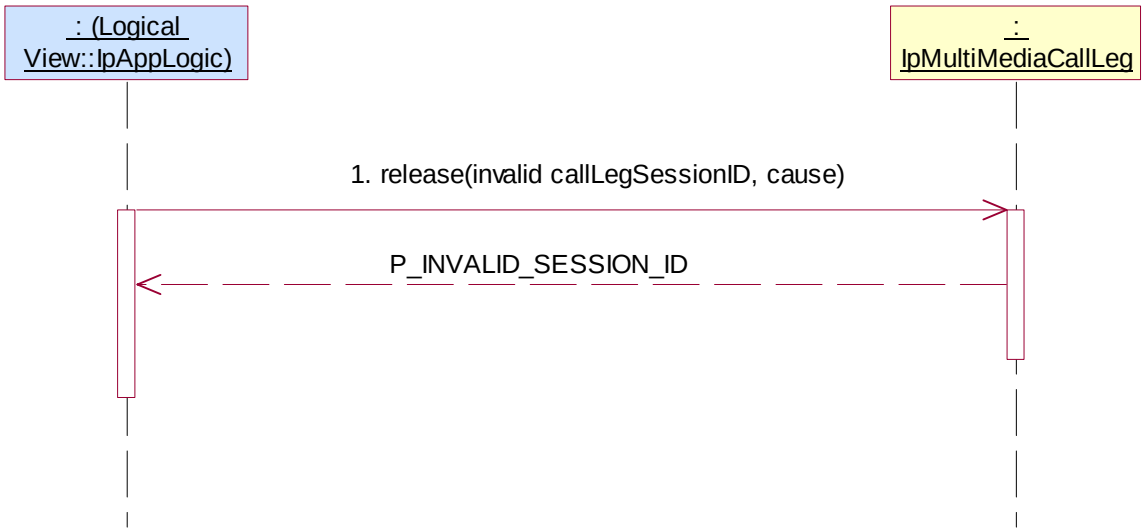
Summary: IpMultiMediaCallLeg, release**,** P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.5

Preamble: Same as CCC _ IpSubConfCall_01

Test Sequence:

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



Test CCC _ IpMultiMediaCallLeg _10

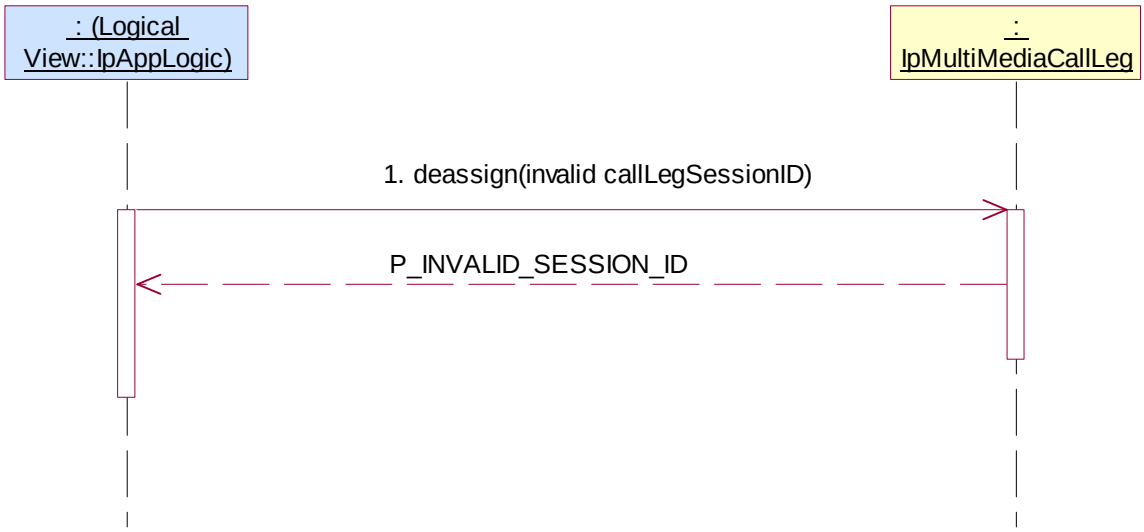
Summary: IpMultiMediaCallLeg, deassign**,** P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.5

Preamble: Same as CCC _ IpSubConfCall_01

Test Sequence:

1. Method call **deassign()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID is returned



Test CCC _ IpMultiMediaCallLeg _11

Summary: IpMultiMediaCallLeg, mediaStreamAllow, P_INVALID_SESSION_ID

Reference: ES 202 915-4-4 [4], clause 6.5

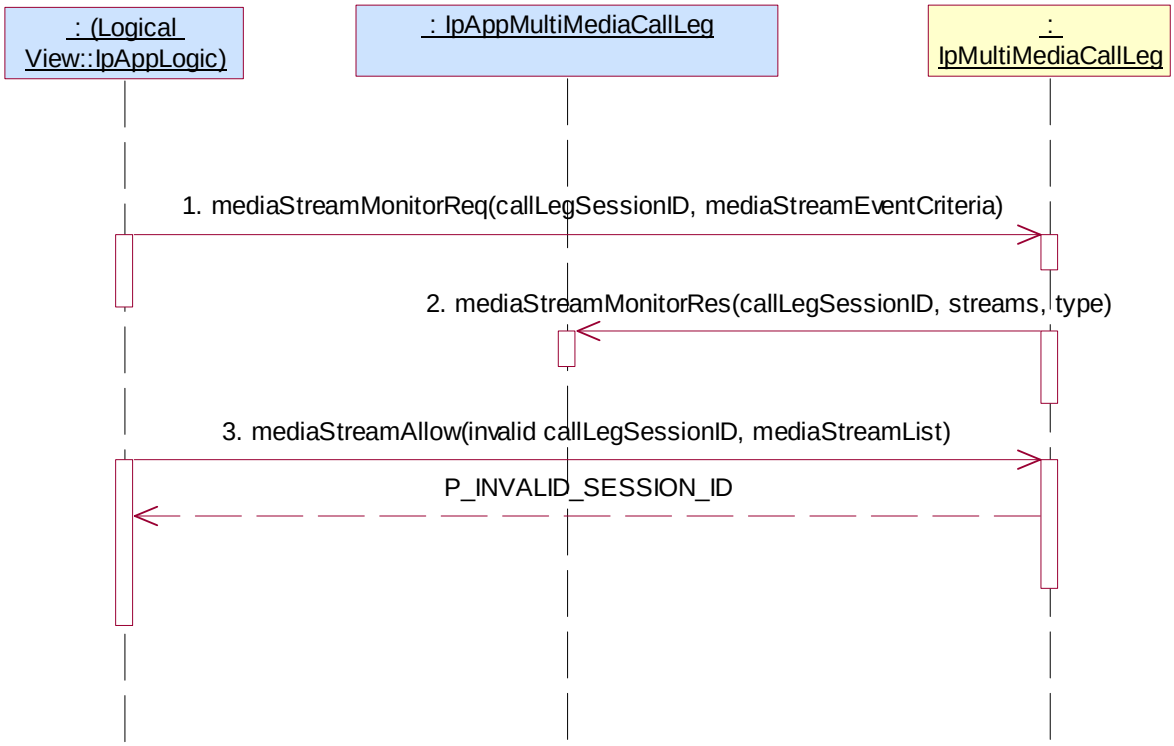
Preamble: Same as CCC _ IpSubConfCall_01

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 is returned



Test CCC _ IpMultiMediaCallLeg _12

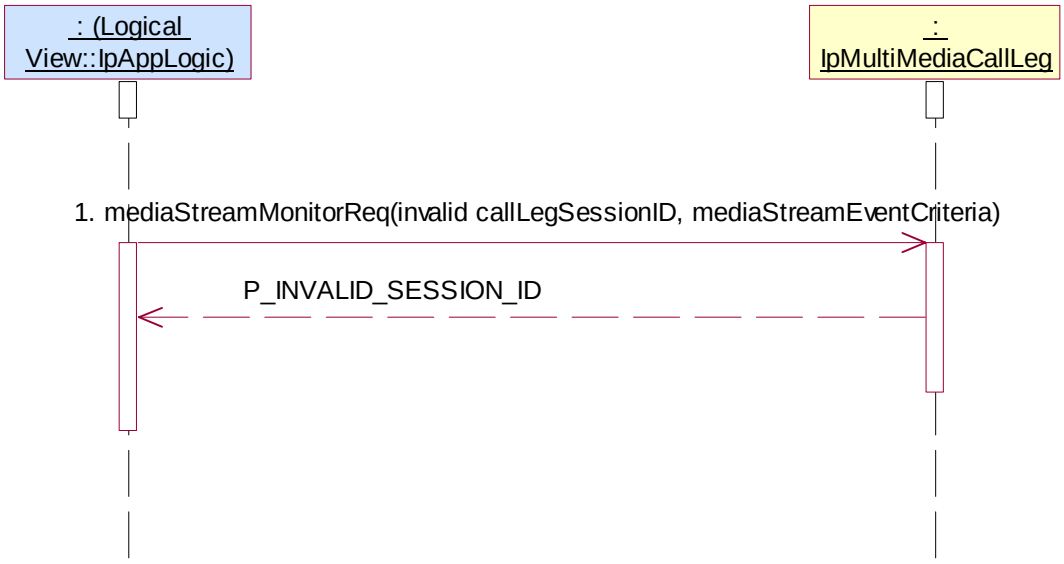
Summary: IpMultiMediaCallLeg, mediaStreamMonitorReq, P_INVALID_SESSION_ID

Reference: ES 202 915-4-4 [4], clause 6.5

Preamble: Same as CCC _ IpSubConfCall_01

Test Sequence:

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



Test CCC _ IpMultiMediaCallLeg _13

Summary: IpMultiMediaCallLeg, mediaStreamMonitorReq, P_INVALID_CRITERIA

Reference: ES 202 915-4-4 [4], clause 6.5

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 is returned

5.2.4.4.3 Optional, valid behaviour

Test CCC _ IpMultiMediaCallLeg _14

Summary: IpMultiMediaCallLeg, getInfoReq, successful

Reference: ES 202 915-4-3 [3], clause 6.5

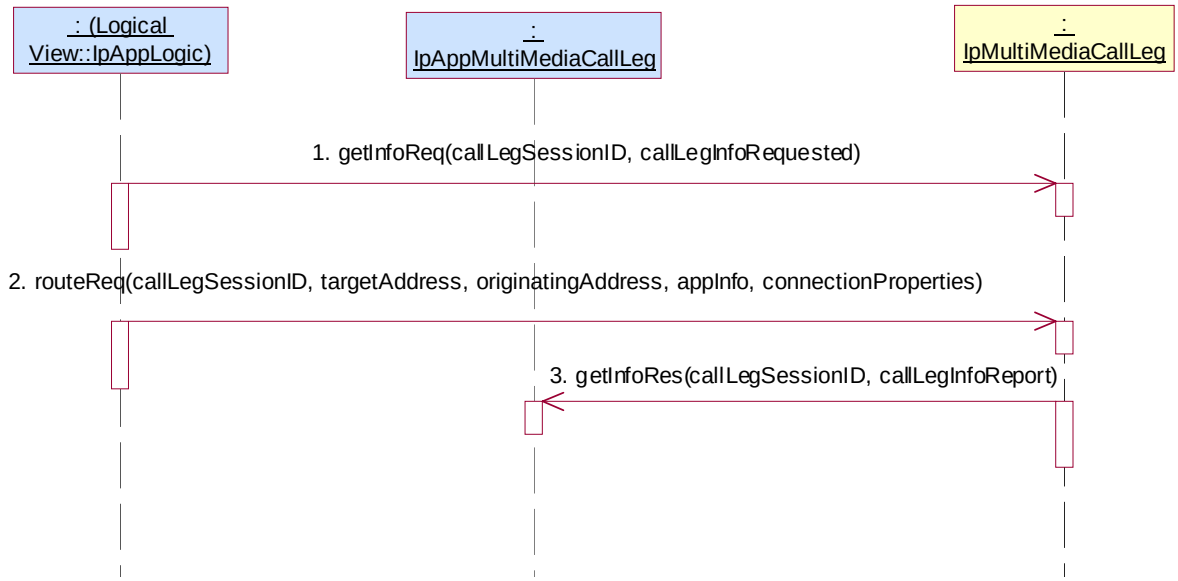
Preamble: Same as CCC _ IpConfCall _08

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 202 915-4-5 [5], clauses 6.1 and 6.3, ES 202 915-4-4 [4], clause 6.5 and ES 202 915-4-3 [3], clause 6.3

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.

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.

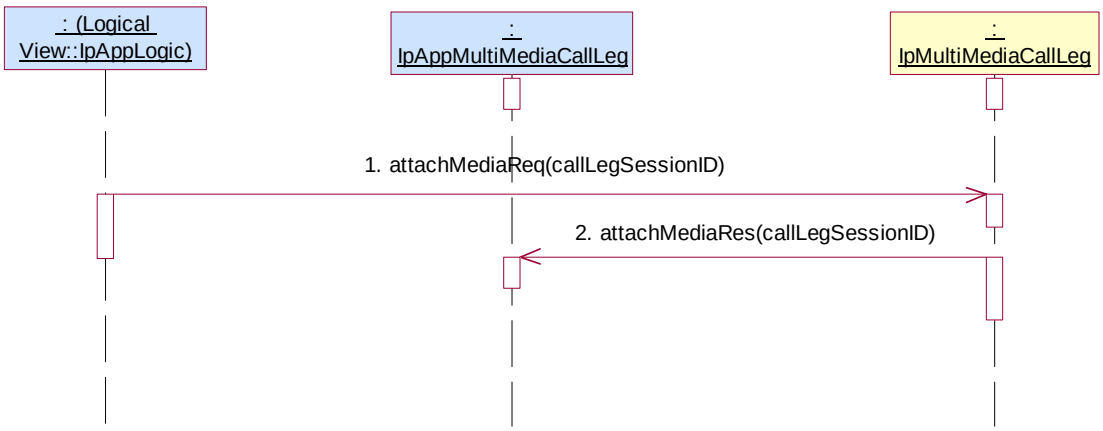
4. Method call **createCallLeg()** on IpSubConfCall
   Parameters: valid callSessionID reported in 2., valid appCallLeg
   Check: valid value of TpCallLegIdentifier is returned

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 202 915-4-4 [4], clause 6.5

Preamble: Same as CCC _ IpMultiMediaCallLeg_15

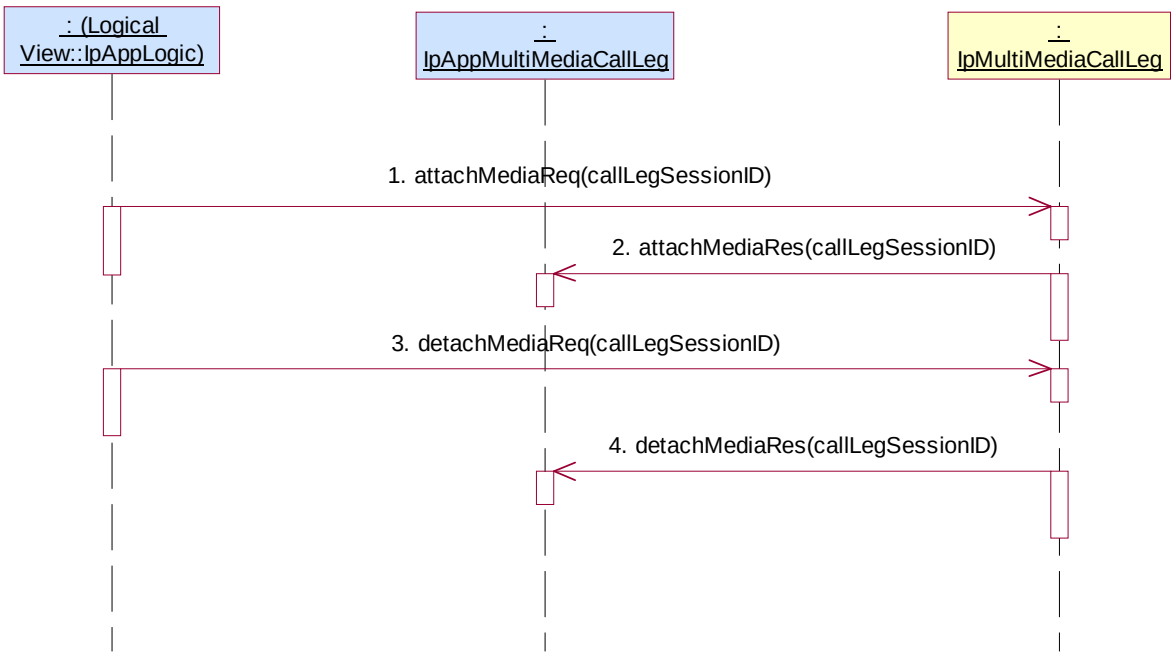
Test Sequence:

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 202 915-4-3 [3], clause 6.5

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

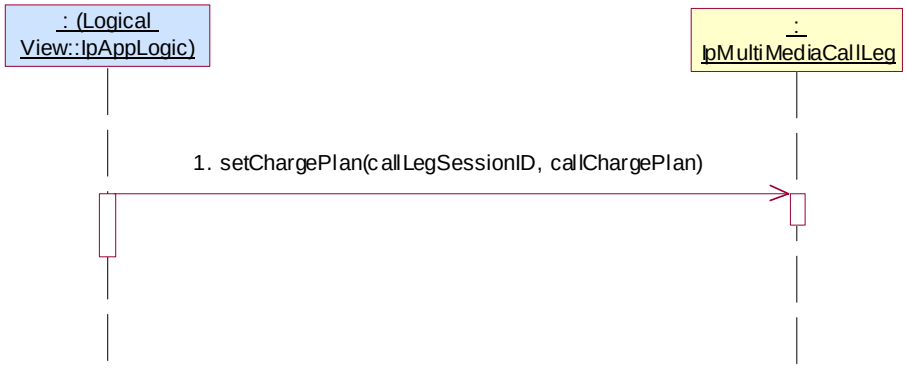
Summary: IpMultiMediaCallLeg, setChargePlan, successful

Reference: ES 202 915-4-3 [3], clause 6.5

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

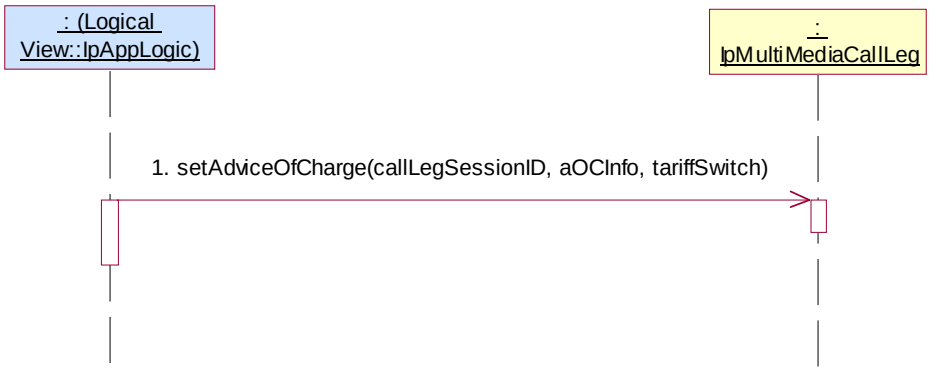
Summary: IpMultiMediaCallLeg, setAdviceOfCharge, successful

Reference: ES 202 915-4-3 [3], clause 6.5

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 202 915-4-3 [3], clause 6.5

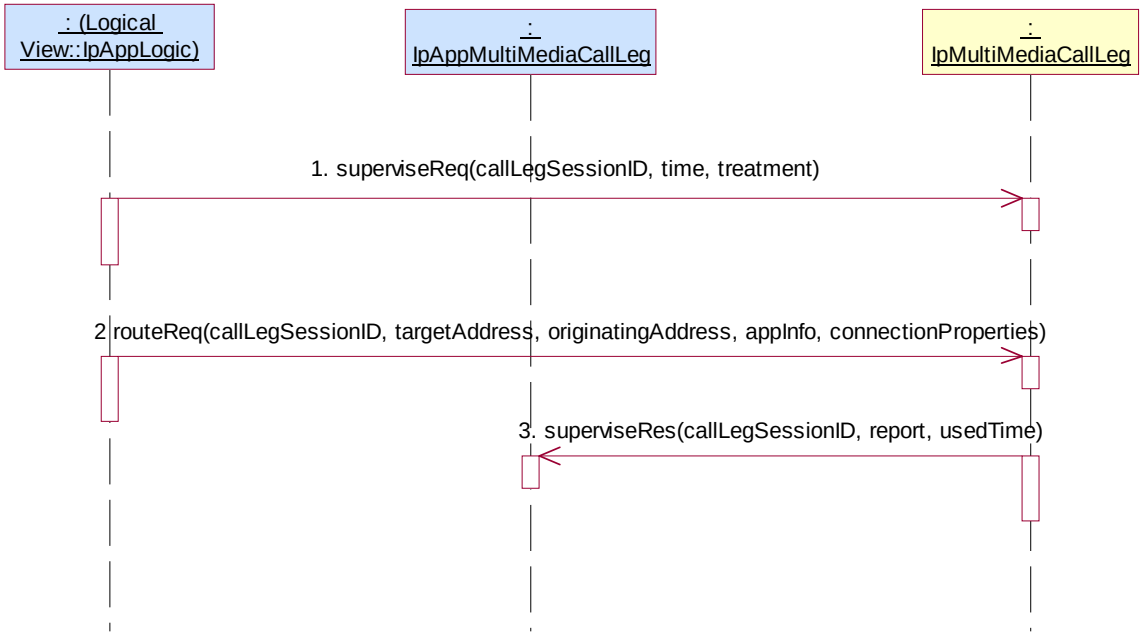
Preamble: Same as CCC _ IpConfCall _08

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

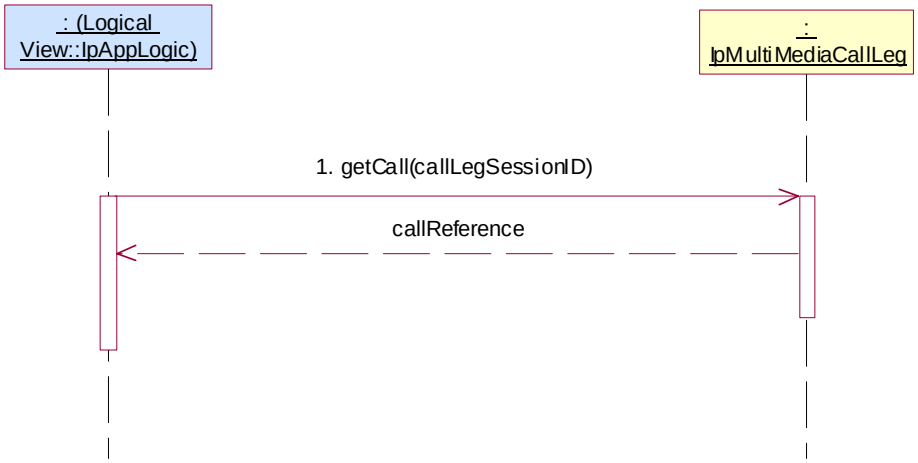
Summary: IpMultiMediaCallLeg, getCall, successful

Reference: ES 202 915-4-3 [3], clause 6.5

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 202 915-4-3 [3], clause 6.5

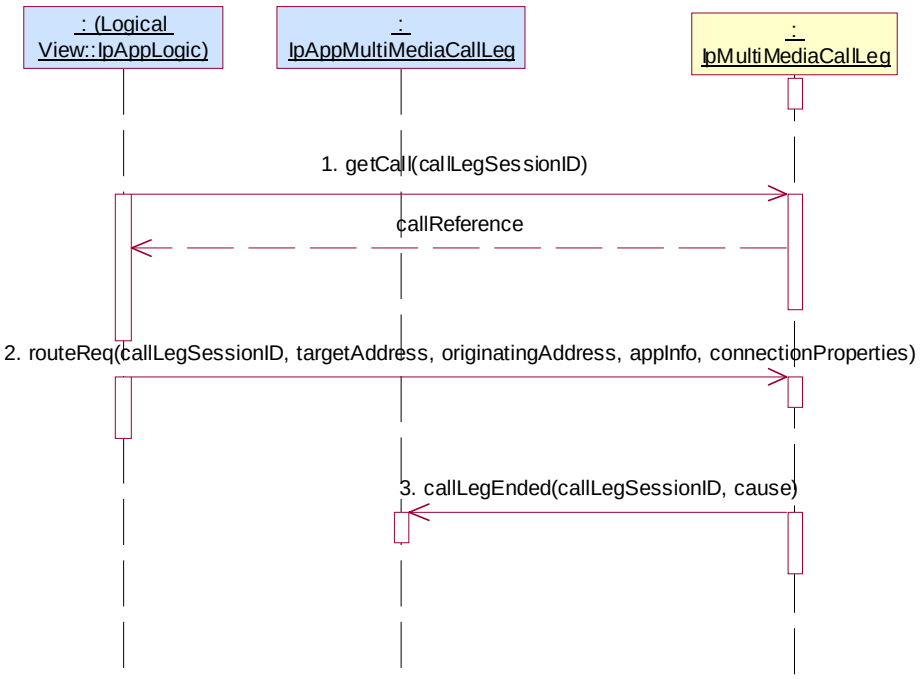
Preamble: Same as CCC _ IpConfCall _08

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 202 915-4-3 [3], clause 6.5

Preamble: Same as CCC _ IpConfCall _08

Test Sequence:

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

Test CCC _ IpMultiMediaCallLeg _24

Summary: IpMultiMediaCallLeg, attachMediaReq, : P_INVALID_SESSION_ID

Reference: ES 202 915-4-4 [4], clause 6.5

Preamble: Same as CCC _ IpMultiMediaCallLeg_15

Test Sequence:

1. Method call **attachMediaReq()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID is returned

Test CCC _ IpMultiMediaCallLeg _25

Summary: IpMultiMediaCallLeg, detachMediaReq, : P_INVALID_SESSION_ID

Reference: ES 202 915-4-4 [4], clause 6.5

Preamble: Same as CCC _ IpSubConfCall_01

Test Sequence:

1. Method call **detachMediaReq()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID is returned

Test CCC _ IpMultiMediaCallLeg _26

Summary: IpMultiMediaCallLeg, getCurrentDestinationAddress, : P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.5

Preamble: Same as CCC _ IpSubConfCall_01

Test Sequence:

1. Method call **getCurrentDestinationAddress()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID is returned

Test CCC _ IpMultiMediaCallLeg _27

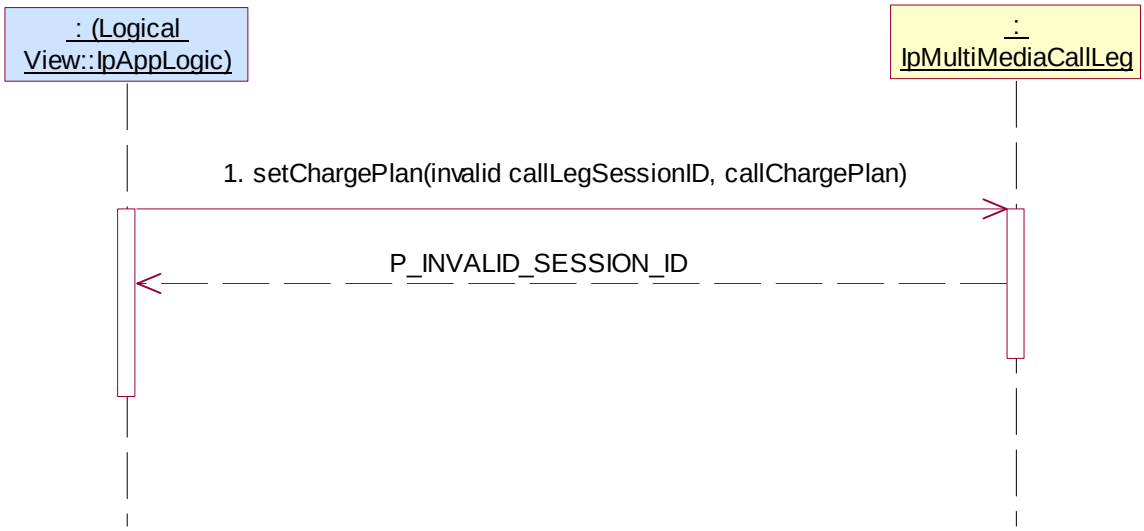
Summary: IpMultiMediaCallLeg, setChargePlan, : P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.5

Preamble: Same as CCC _ IpConfCall _08

Test Sequence:

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



Test CCC _ IpMultiMediaCallLeg _28

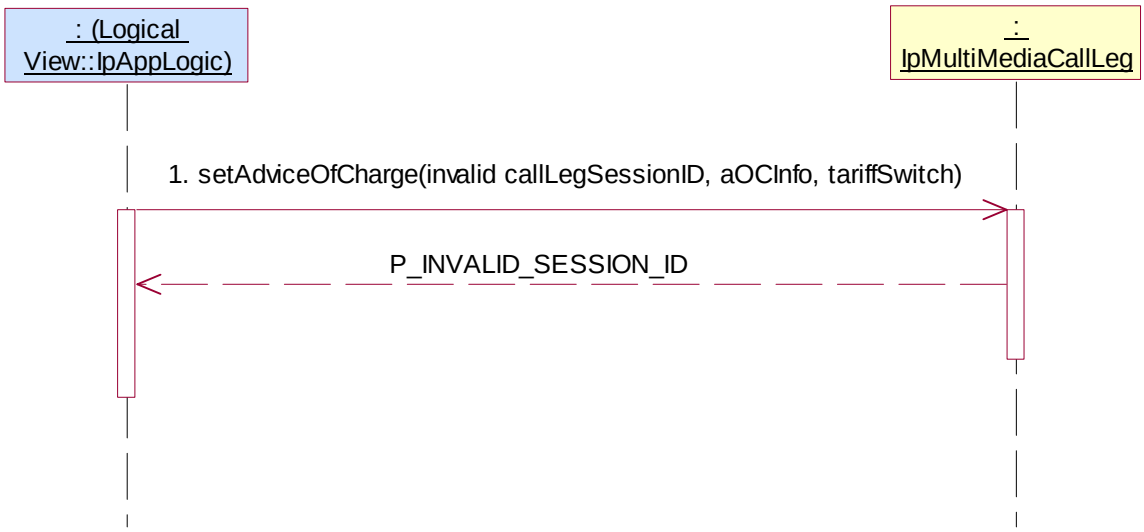
Summary: IpMultiMediaCallLeg, setAdviceOfCharge, : P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.5

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 is returned



Test CCC _ IpMultiMediaCallLeg _29

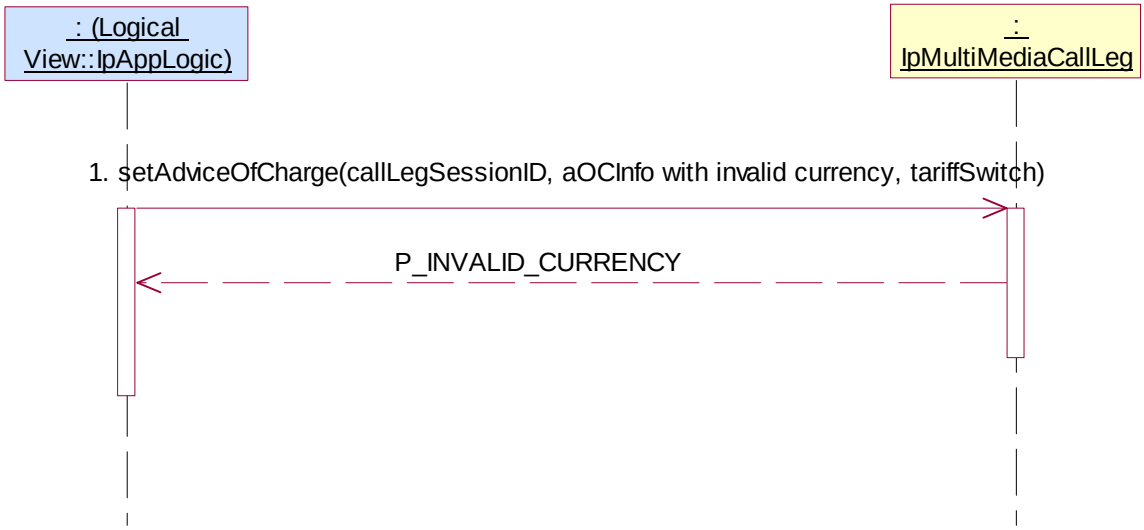
Summary: IpMultiMediaCallLeg, setAdviceOfCharge, P_INVALID_CURRENCY

Reference: ES 202 915-4-3 [3], clause 6.5

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 is returned



Test CCC _ IpMultiMediaCallLeg _30

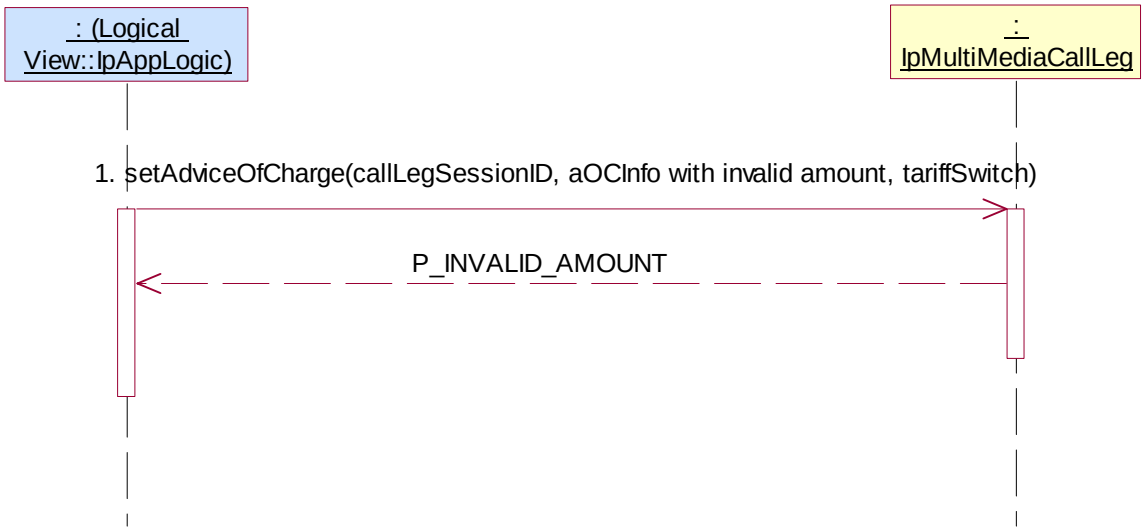
Summary: IpMultiMediaCallLeg, setAdviceOfCharge, P_INVALID_AMOUNT

Reference: ES 202 915-4-3 [3], clause 6.5

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 is returned



Test CCC _ IpMultiMediaCallLeg _31

Summary: IpMultiMediaCallLeg, superviseReq, : P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.5

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 is returned

Test CCC _ IpMultiMediaCallLeg _32

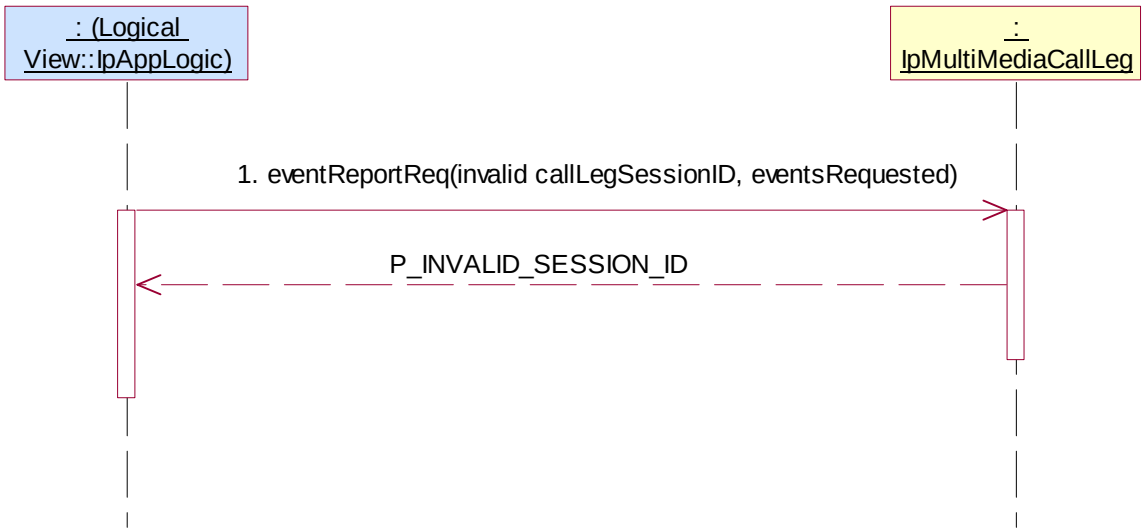
Summary: IpMultiMediaCallLeg, eventReportReq**,** P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.5

Preamble: Same as CCC _ IpConfCall _08

Test Sequence:

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



Test CCC _ IpMultiMediaCallLeg _33

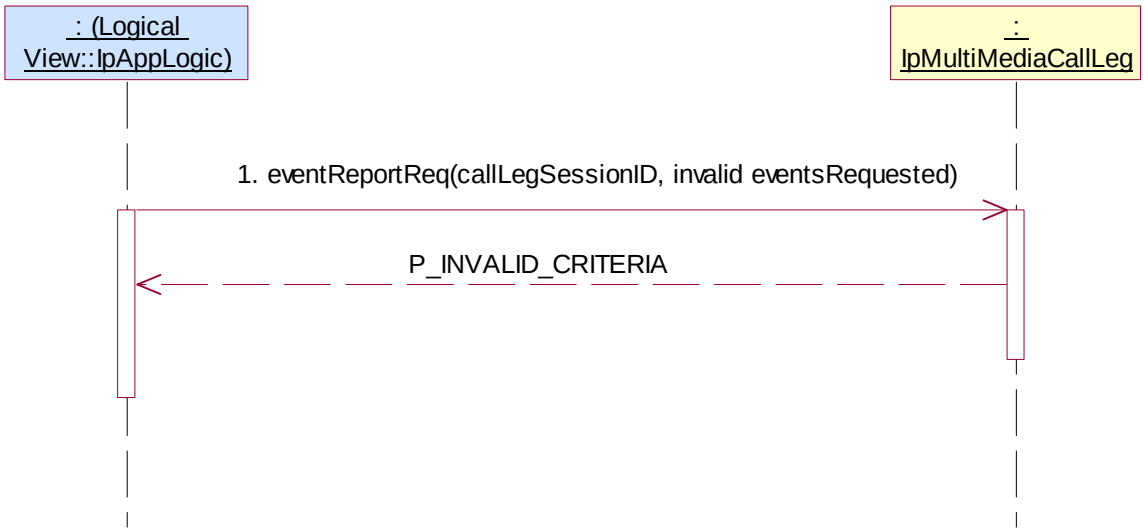
Summary: IpMultiMediaCallLeg, eventReportReq, P_INVALID_CRITERIA

Reference: ES 202 915-4-3 [3], clause 6.5

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 is returned



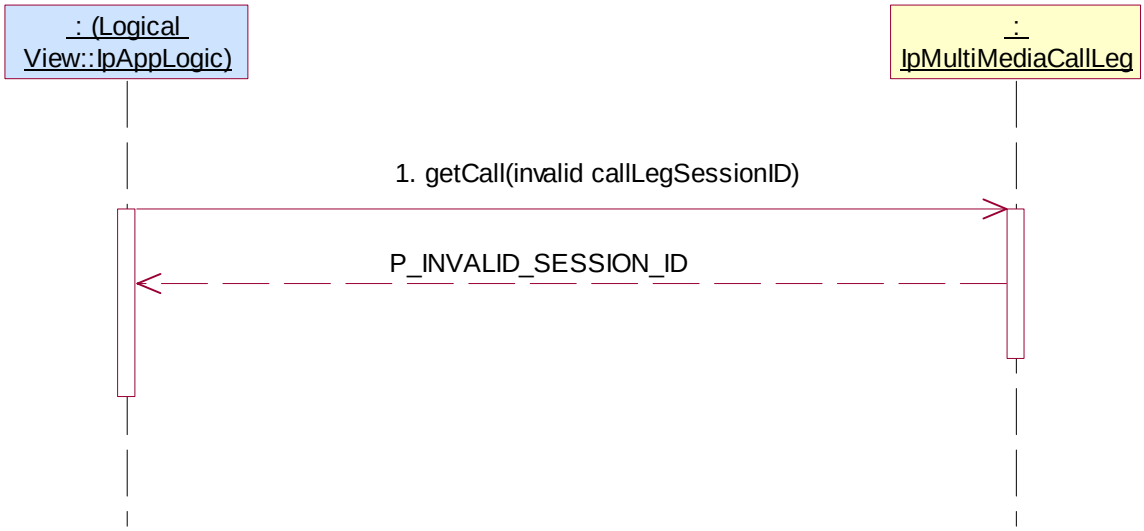
Test CCC _ IpMultiMediaCallLeg _34

Summary: IpMultiMediaCallLeg, getCall**,** P_INVALID_SESSION_ID

Reference: ES 202 915-4-3 [3], clause 6.5

Preamble: Same as CCC _ IpConfCall _08

1. Method call **getCall()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID is returned



Test CCC _ IpMultiMediaCallLeg _35

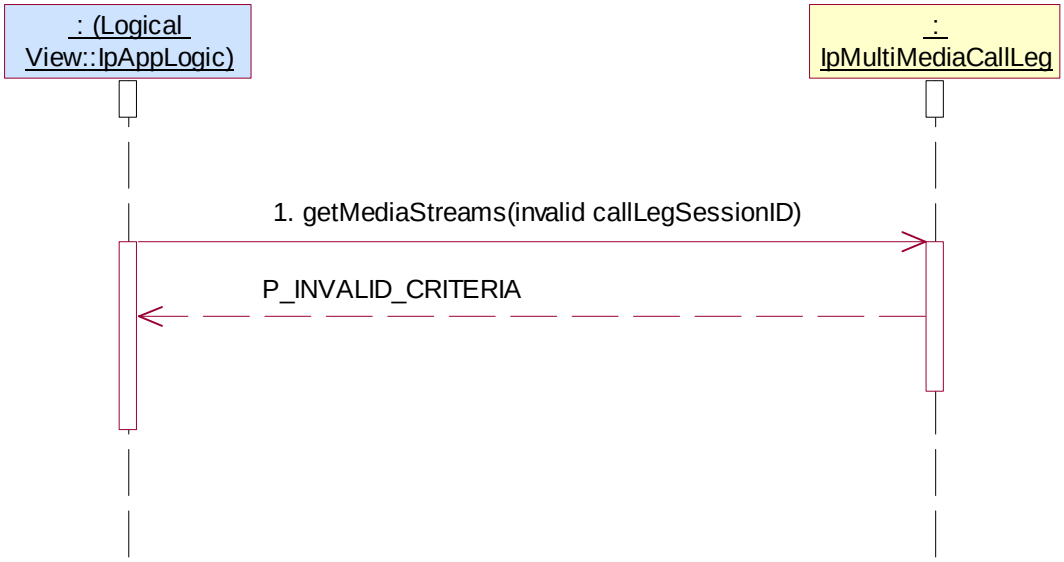
Summary: IpMultiMediaCallLeg, getMediaStreams, P_INVALID_SESSION_ID

Reference: ES 202 915-4-4 [4], clause 6.5

Preamble: Same as CCC _ IpSubConfCall_01

Test Sequence:

1. Method call **getMediaStreams()** on IpMultiMediaCallLeg
   Parameters: invalid callLegSessionID
   Check: P_INVALID_SESSION_ID 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 202 915-4-5 [5], clauses 6.1 and 6.3 and ES 202 915-4-4 [4], clause 6.5 and ES 202 915-4-3 [3], clause 6.5

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 **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

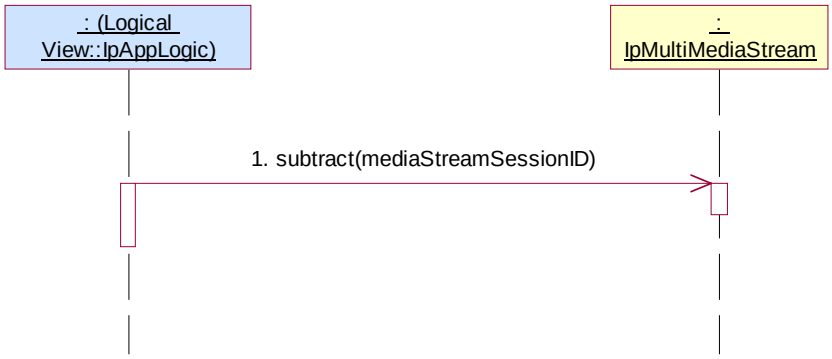
5. Method call **mediaStreamMonitorReq()** on IpMultiMediaCallLeg
   Parameters: valid callLegSessionID returned in 3., valid mediaStreamEventCriteria
   Check: no exception is returned

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

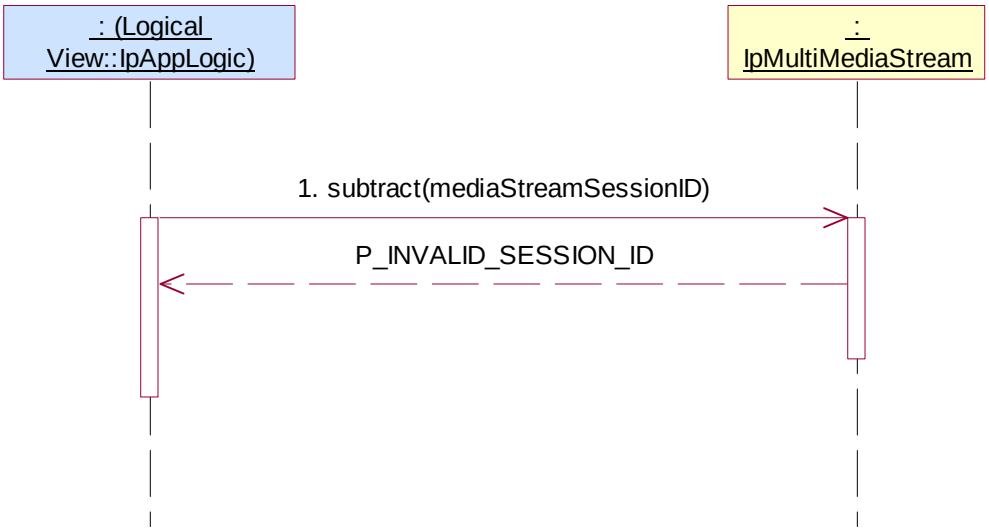
Summary: IpMultiMediaStream, subtract, P_INVALID_SESSION_ID

Reference: ES 202 915-4-4 [4], clause 6.7

Preamble: Same as Test CCC_ IpMultiMediaStream _01

Test Sequence:

1. Method call **subtract()** on IpMultiMediaStream
   Parameters: invalid mediaStreamSessionID
   Check: P_INVALID_SESSION_ID is returned



6 Test Suite Structure (TSS) - Application

Call Control (CC):

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

7 Test Purposes (TP) - Application

7.1 Introduction

For each test requirement a TP is defined.

7.1.1 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 2).

Table 2: 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:	field representing the group reference according to TSS
<nn>	= sequential number:	(01-99)

7.1.2 Source of TP definition

The TPs are based on ES 202 915-4-1 [1], ES 202 915-4-2 [2], ES 202 915-4-3 [3], ES 202 915-4-4 [4] and ES 202 915-4-5 [5].

7.1.3 Test strategy

As the base standards ES 202 915-4-1 [1], ES 202 915-4-2 [2], ES 202 915-4-3 [3], ES 202 915-4-4 [4] and ES 202 915-4-5 [5] contain no explicit requirements for testing, the TPs were generated as a result of an analysis of the base standards and the ICS specification ES 202 363 [6].

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 [9]).

7.2 TPs for the application using the Call Control SCF

All ICS items referred to in this clause are as specified in ES 202 363 [6] unless indicated otherwise by another numbered reference.

All parameters specified in method calls are valid unless specified.

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

In performing the tests for the application, it may be necessary to permit the application to perform any valid set of method exchanges with the tester in between the triggered actions or method calls indicated in the tests below. The tester shall respond to the application's method calls in conformance to the API specification and as required by the application. The requirements of the application should be made known to the tester's operator in advance.

7.2.1 Generic Call Control

The TPs in this clause are based on ES 202 915-4-2 [2].

7.2.1.1 IpAppCallControlManager

Test GCC_IPAPPCALLCONTROLMANAGER_01

Summary: create call object

Reference: ES 202 915-4-2 [2], clause 6.1

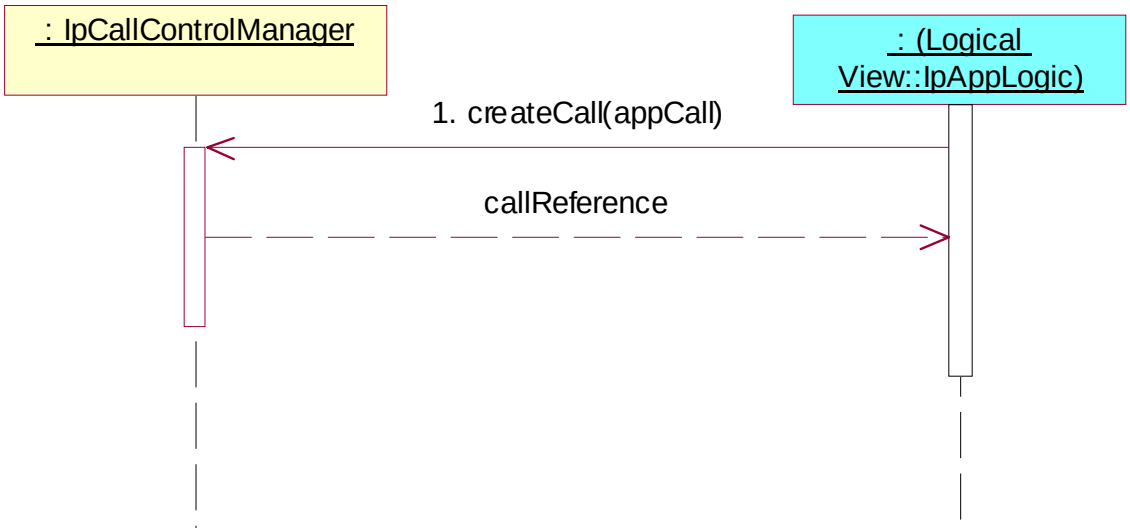
Precondition: IUT capable of invoking **createCall()**

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

The application is permitted to provide its IpAppCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **createCall()** method on the tester's (SCF's) IpCallControlManager interface.
   Parameters: appCall



Test GCC_IPAPPCALLCONTROLMANAGER_02

Summary: create call object and accept abort

Reference: ES 202 915-4-2 [2], clauses 6.1 and 6.2

Precondition: IUT capable of invoking **createCall()**; **callAborted()** implemented

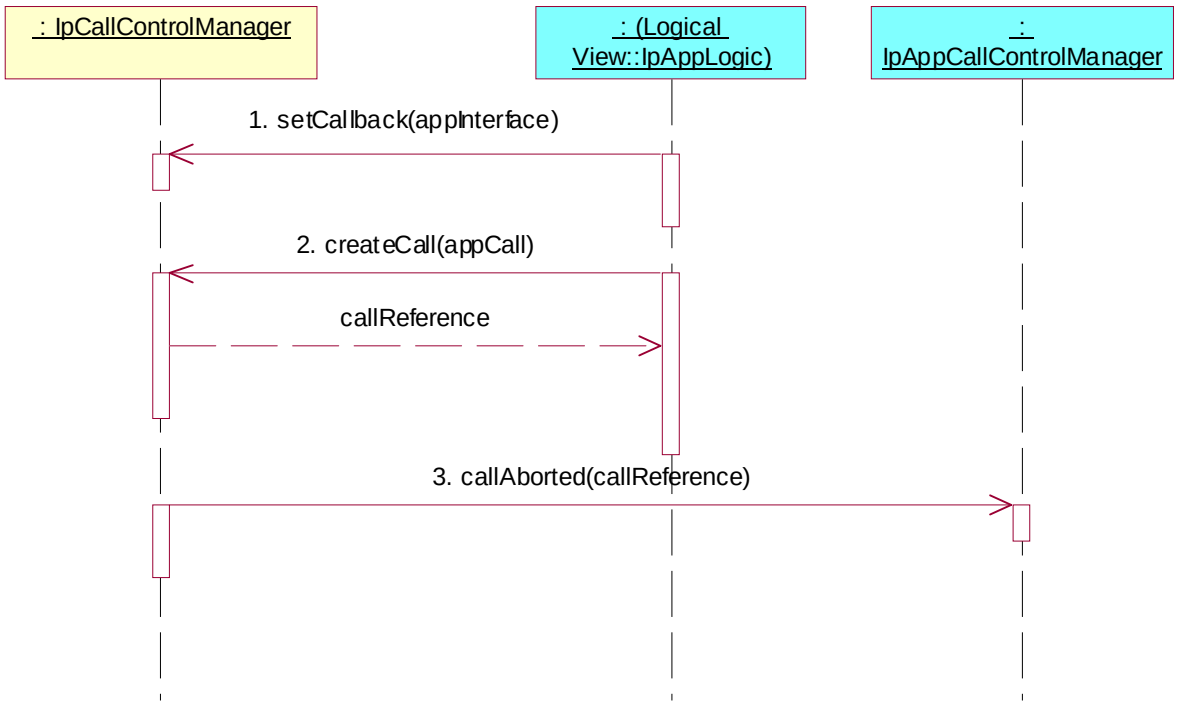
Preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpCallControlManager interface through selecting that service and signing the required service agreement.

Test Sequence:

1. Triggered Action: cause IUT to call **setCallBack()** method on the tester's (SCF's) IpCallControlManager interface.
   Parameters: valid, non NULL, value of appInterface

2. Triggered Action: cause IUT to call **createCall()** method on the tester's (SCF's) IpCallControlManager interface.
   Parameters: appCall

3. Method call **callAborted()**
   Parameters: callReference
   Check: no exception is returned



Test GCC_IPAPPCALLCONTROLMANAGER_03

Summary: enable and accept call notifications

Reference: ES 202 915-4-2 [2], clauses 6.1 and 6.2

Precondition: IUT capable of invoking **enableCallNotification()**, **callEventNotify()** implemented

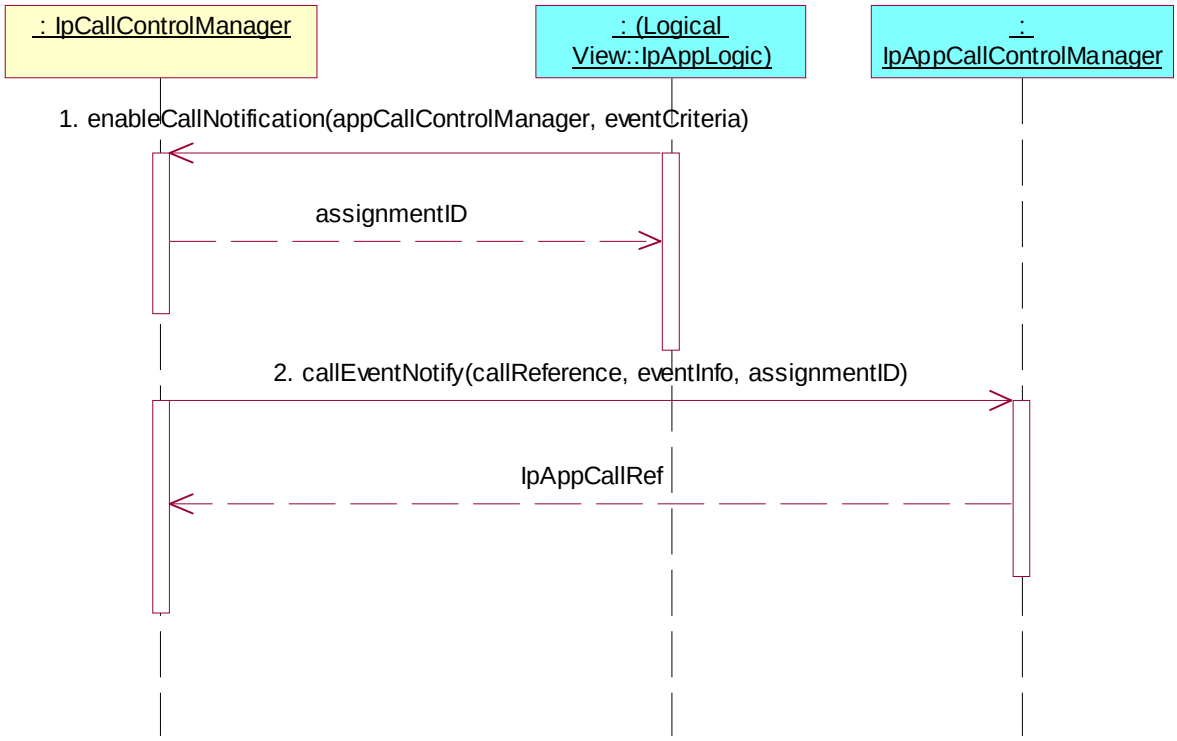
Preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **enableCallNotification()** method on the tester's (SCF's) IpCallControlManager interface.
   Parameters: appCallControlManager, eventCriteria

2. Method call **callEventNotify()**
   Parameters: callReference, eventInfo, assignmentID
   Check: valid value of IpAppCallRef is returned



Test GCC_IPAPPCALLCONTROLMANAGER_04

Summary: interrupt and continue call notifications

Reference: ES 202 915-4-2 [2], clauses 6.1 and 6.2

Precondition: IUT capable of invoking **enableCallNotification()**, **callNotificationInterrupted(), callNotificationContinued()** and **callEventNotify()** implemented

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

The application is permitted to provide its IpAppCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **enableCallNotification()** method on the tester's (SCF's) IpCallControlManager interface.
   Parameters: appCallControlManager, eventCriteria

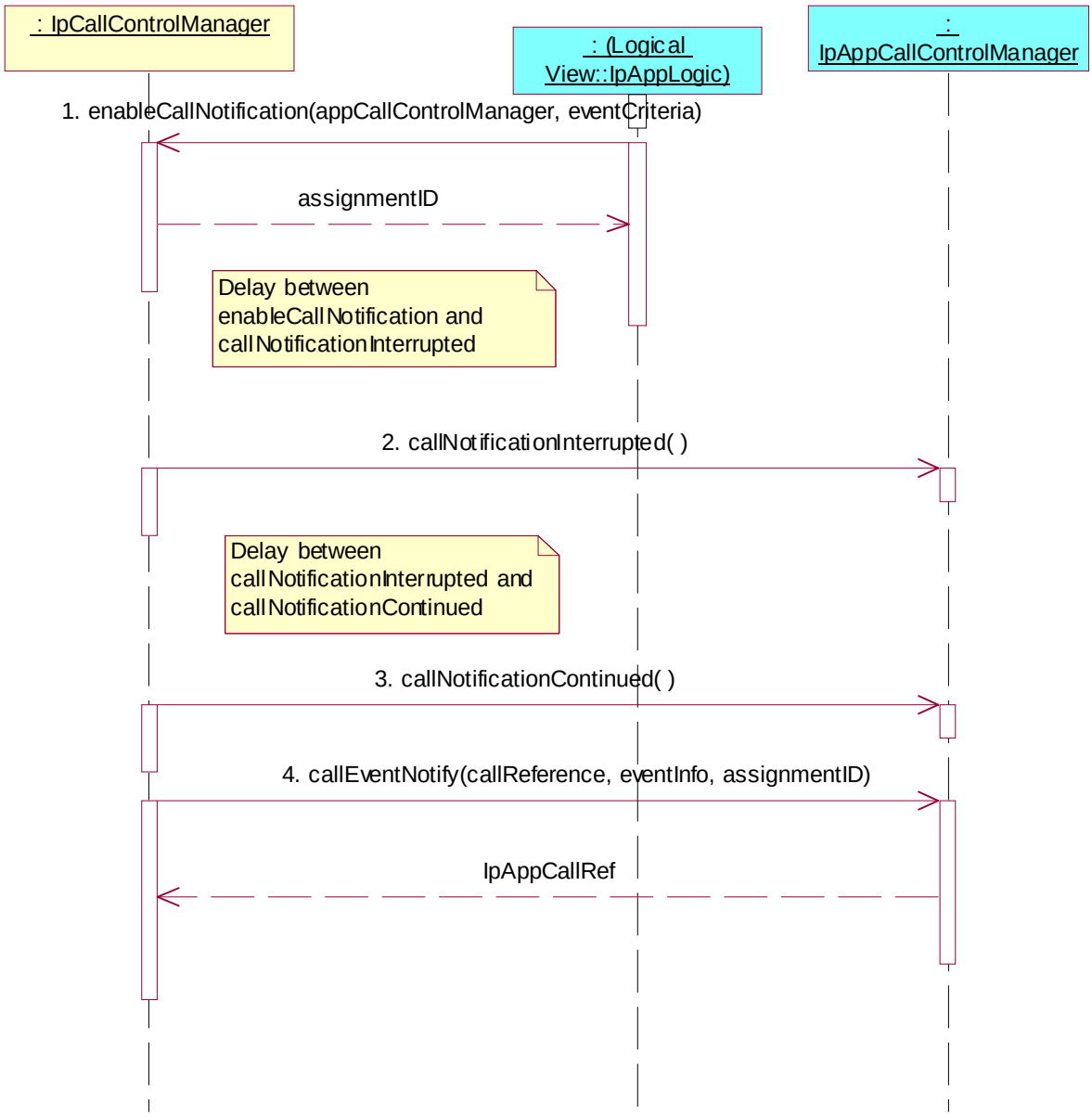
Delay between enableCallNotification and callNotificationInterrupted

2. Method call **callNotificationInterrupted()**
   Parameters: none
   Check: no exception is returned

Delay between callNotificationInterrupted and callNotificationContinued

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

4. Method call **callEventNotify()**
   Parameters: callReference, eventInfo, assignmentID
   Check: valid value of IpAppCallRef is returned



Test GCC_IPAPPCALLCONTROLMANAGER_05

Summary: enable and change call notifications

Reference: ES 202 915-4-2 [2], clauses 6.1 and 6.2

Precondition: IUT capable of invoking **enableCallNotification()** and **changeCallNotification()**, **callEventNotify()** implemented

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

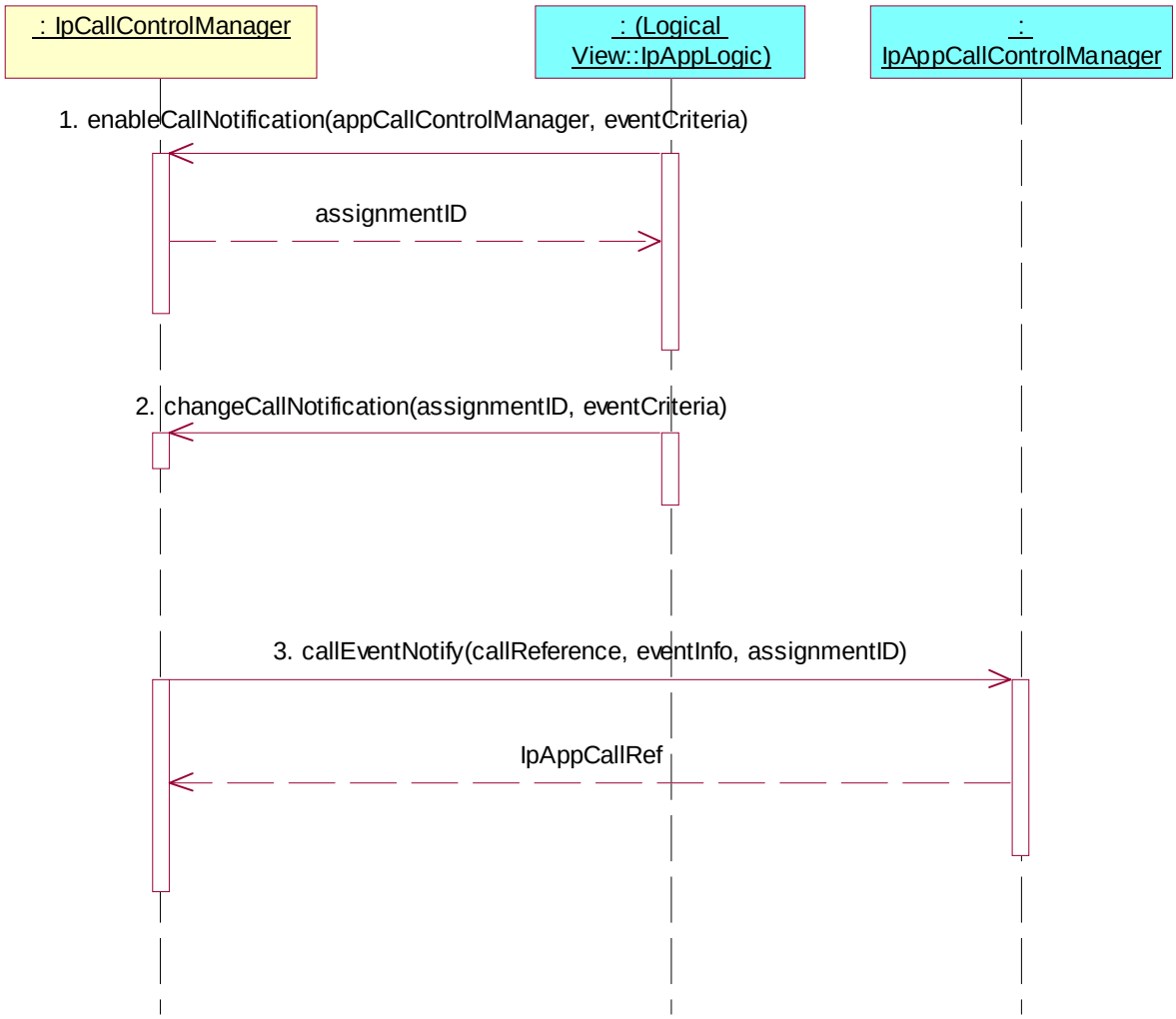
The application is permitted to provide its IpAppCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **enableCallNotification()** method on the tester's (SCF's) IpCallControlManager interface.
   Parameters: appCallControlManager, eventCriteria

2. Triggered Action: cause IUT to call **changeCallNotification()** method on the tester's (SCF's) IpCallControlManager interface.
   Parameters: assignmentID, eventCriteria

3. Method call **callEventNotify()**
   Parameters: callReference, eventInfo, assignmentID
   Check: valid value of IpAppCallRef is returned



Test GCC_IPAPPCALLCONTROLMANAGER_06

Summary: enable and disable call notifications

Reference: ES 202 915-4-2 [2], clauses 6.1 and 6.2

Precondition: IUT capable of invoking **enableCallNotification()** and **disableCallNotification()**

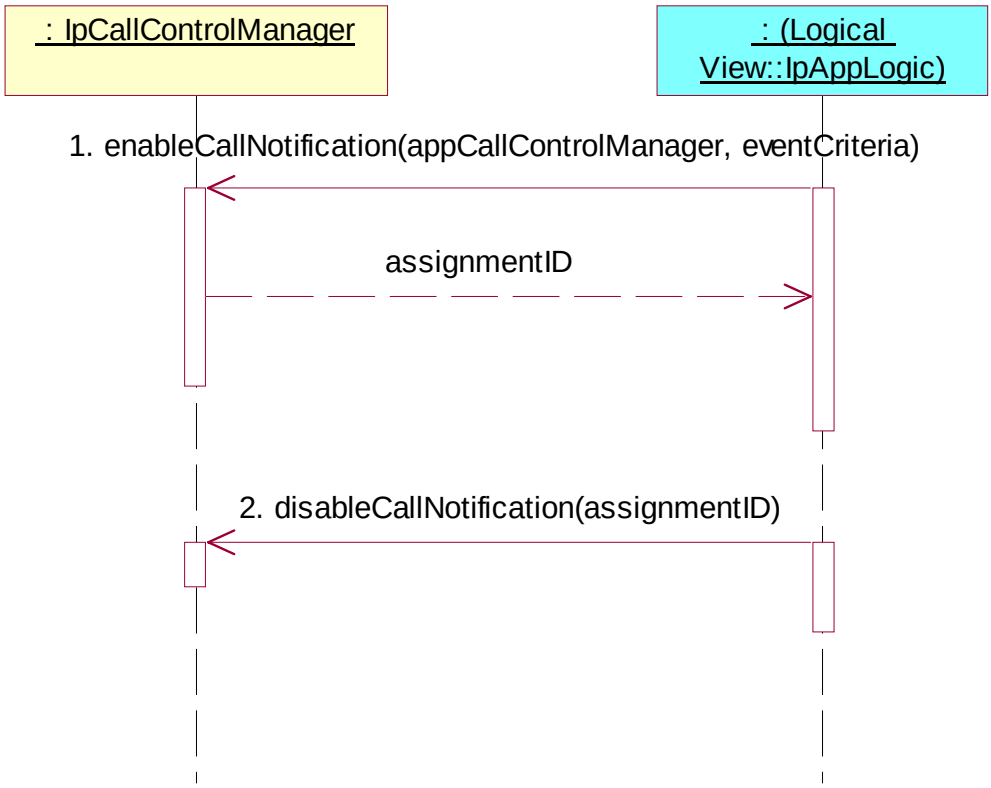
Preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **enableCallNotification()** method on the tester's (SCF's) IpCallControlManager interface.
   Parameters: appCallControlManager, eventCriteria

2. Triggered Action: cause IUT to call **disableCallNotification()** method on the tester's (SCF's) IpCallControlManager interface.
   Parameters: assignmentID



Test GCC_IPAPPCALLCONTROLMANAGER_07

Summary: enablecall notifications and get criteria

Reference: ES 202 915-4-2 [2], clauses 6.1 and 6.2

Precondition: IUT capable of invoking **enableCallNotification()** and **getCriteria()**

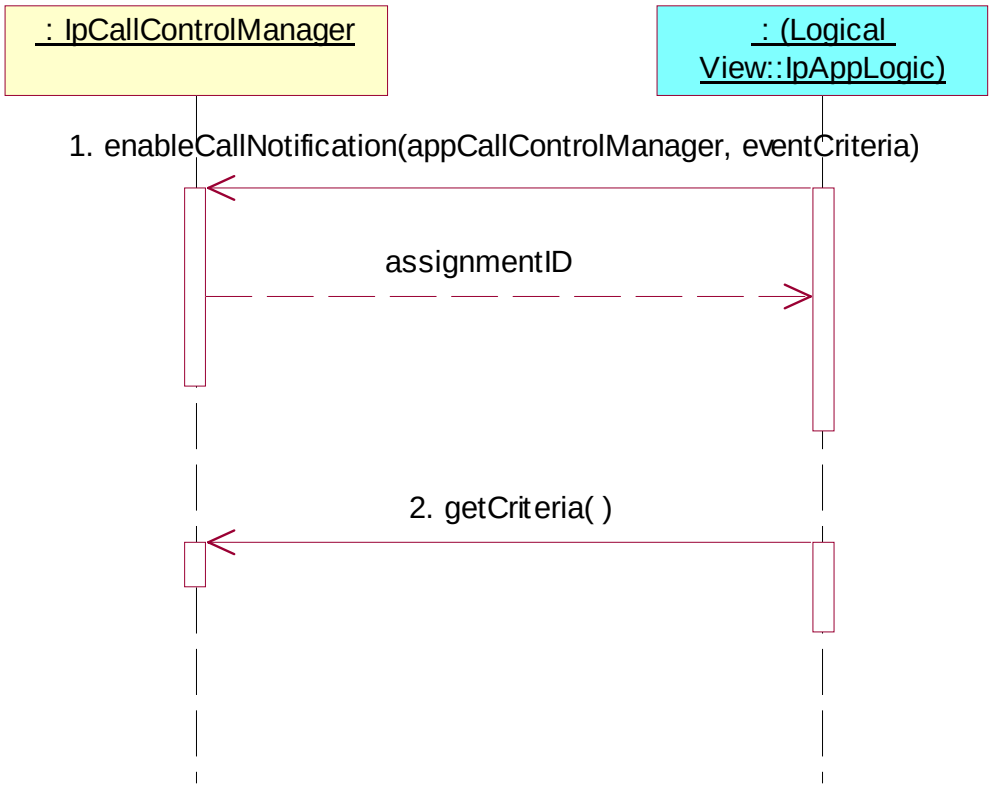
Preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **enableCallNotification()** method on the tester's (SCF's) IpCallControlManager interface.
   Parameters: appCallControlManager, eventCriteria

2. Triggered Action: cause IUT to call **getCriteria()** method on the tester's (SCF's) IpCallControlManager interface.
   Parameters: none



Test GCC_IPAPPCALLCONTROLMANAGER_08

Summary: enable load control and accept overload notifications

Reference: ES 202 915-4-2 [2], clauses 6.1 and 6.2

Precondition: IUT capable of invoking **setCallLoadControl()**, **callOverloadEncountered()** and **callOverloadCeased()** implemented

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

The application is permitted to provide its IpAppCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

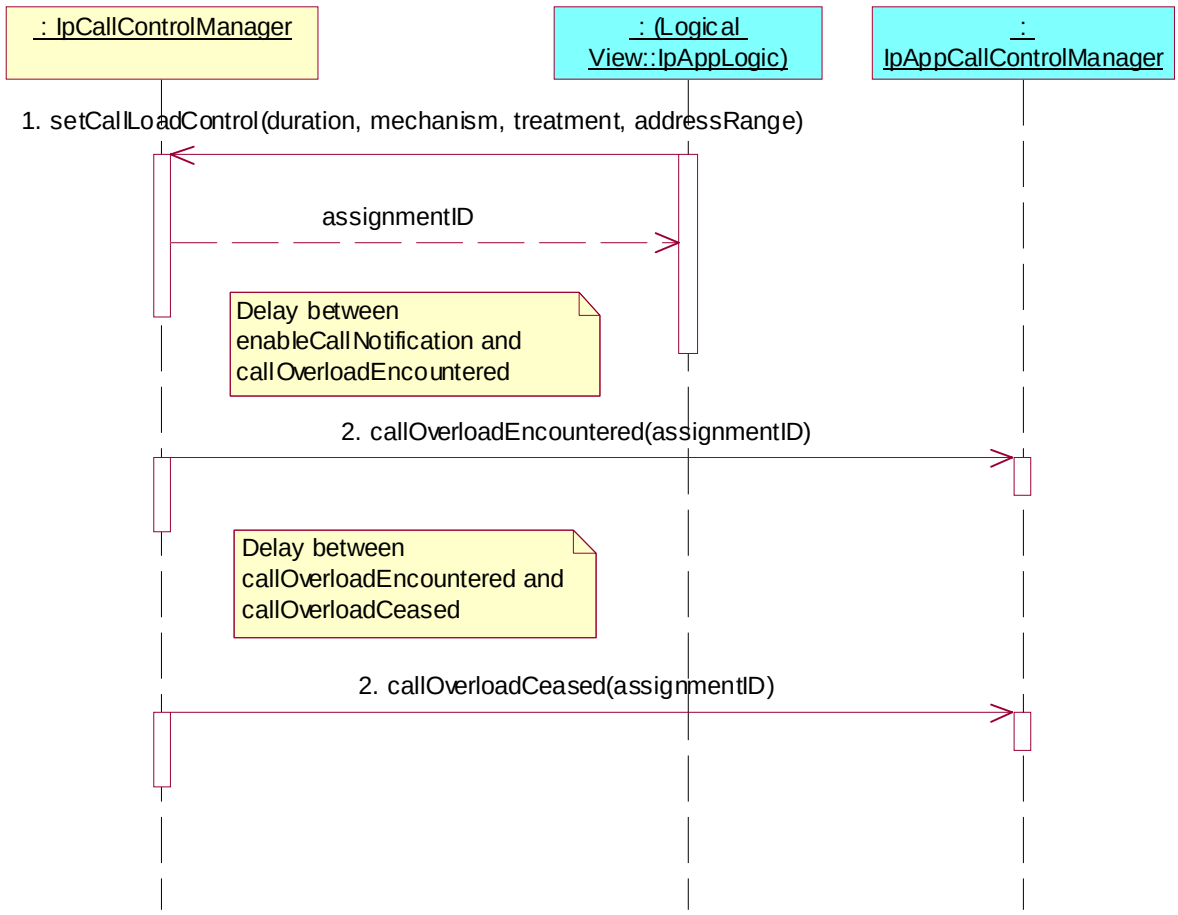
1. Triggered Action: cause IUT to call **setCallLoadControl()** method on the tester's (SCF's) IpCallControlManager interface.
   Parameters: duration, mechanism, treatment, addressRange

Delay between setCallLoadControl and callOverloadEncountered

2. Method call **callOverloadEncountered()**
   Parameters: assignmentID
   Check: no exception is returned

Delay between callOverloadEncountered and callOverloadCeased

3. Method call **callOverloadCeased()**
   Parameters: assignmentID
   Check: no exception is returned



7.2.1.2 IpAppCall

Applications need not be capable of performing each of the sequences below, even if they support the methods indicated below.

Reference: ES 202 915-4-2 [2], clause 7.2

7.2.1.2.1 No Parties state

Precondition: IUT capable of invoking **createCall()**

Preamble GCC_IPAPPCALL_No_Parties

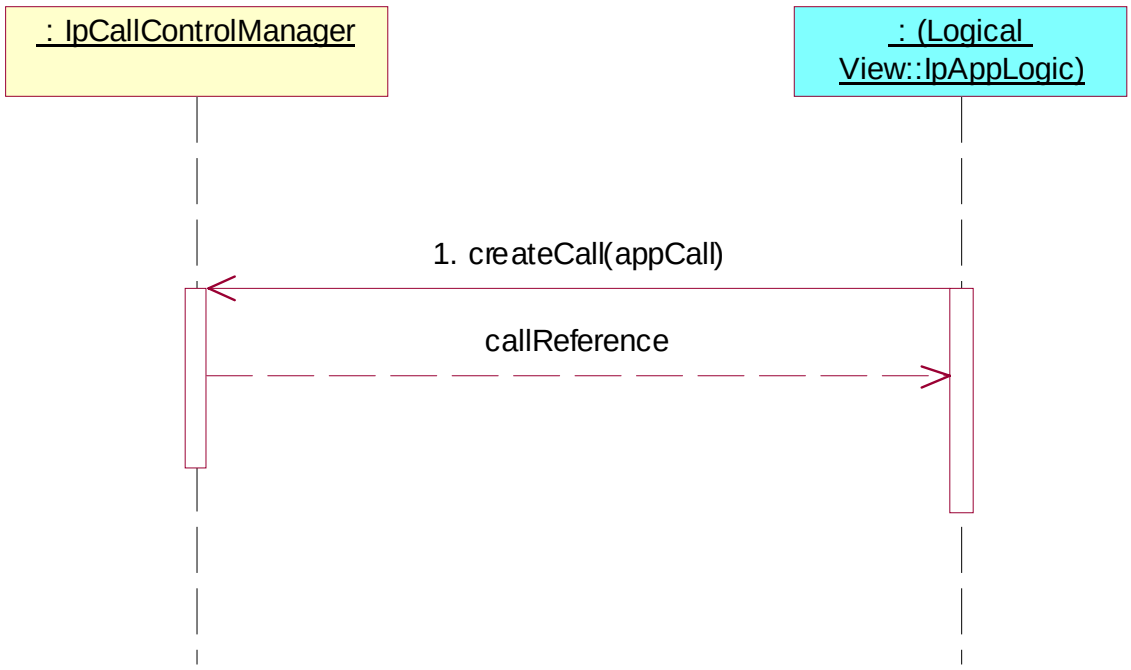
Reference: ES 202 915-4-2 [2], clause 7.2

Pre-preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Preamble Sequence:

1. Triggered Action: cause IUT to call **createCall()** method on the tester's (SCF's) IpCallControlManager interface.
   Parameters: appCall



Test GCC_IPAPPCALL_01

Summary: application initiated call, request connection to 1st call leg

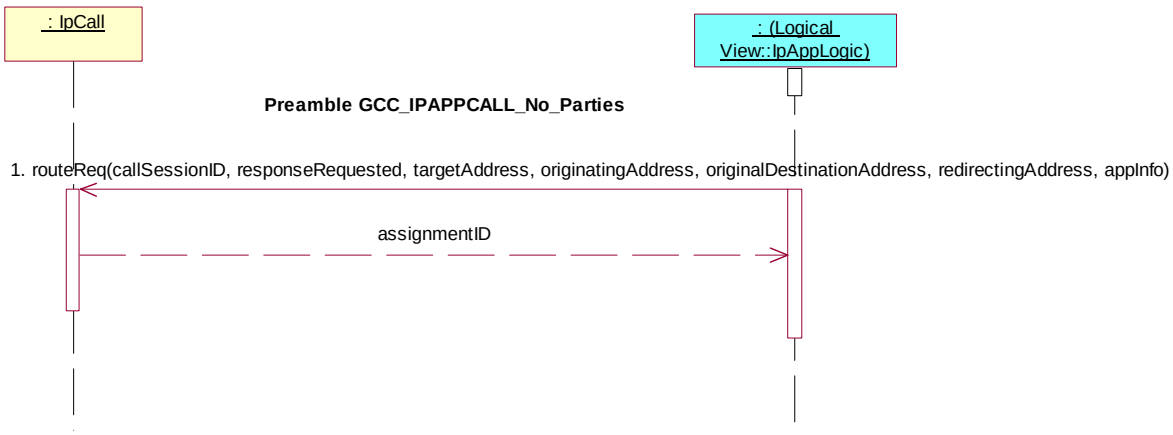
Reference: ES 202 915-4-2 [2], clauses 4.3, 6.3 and 6.4

Precondition: IUT capable of invoking **routeReq()**

Preamble: GCC_IPAPPCALL_No_Parties

Test Sequence:

1. Triggered Action: cause IUT to call **routeReq()** method on the tester's (SCF's) IpCall interface, i.e. connect to call leg to subscriber A.
   Parameters: callSessionID, responseRequested, targetAddress, originatingAddress, originalDestinationAddress, redirectingAddress, appInfo



Test GCC_IPAPPCALL_02

Summary: set supervision of call

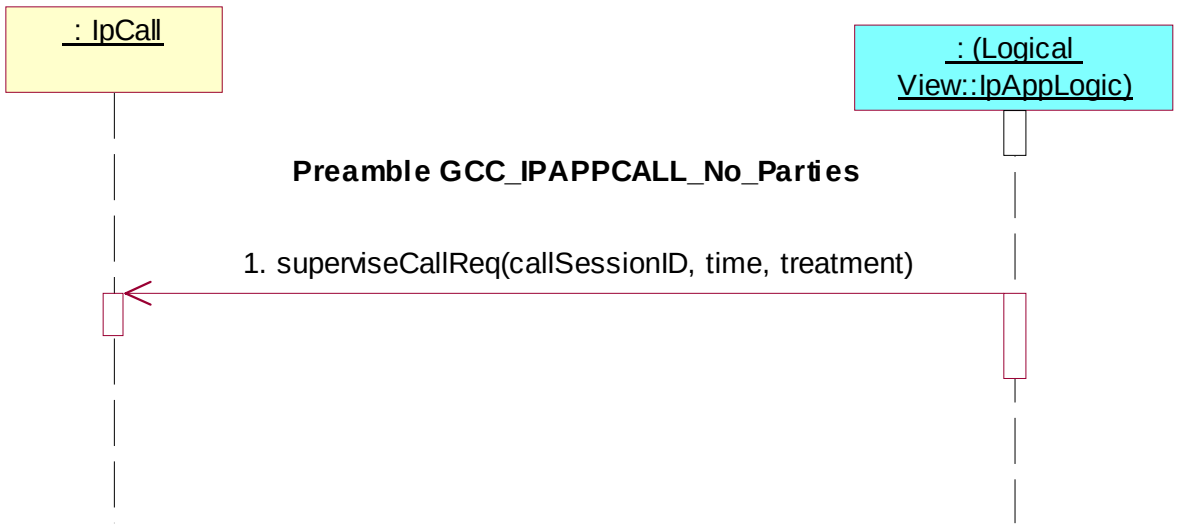
Reference: ES 202 915-4-2 [2], clause 6.3

Precondition: IUT capable of invoking **superviseCallReq()**

Preamble: GCC_IPAPPCALL_No_Parties

Test Sequence:

1. Triggered Action: cause IUT to call **superviseCallReq()** method on the tester's (SCF's) IpCall interface.
   Parameters: callSessionID, time, treatment



Test GCC_IPAPPCALL_03

Summary: request information about call

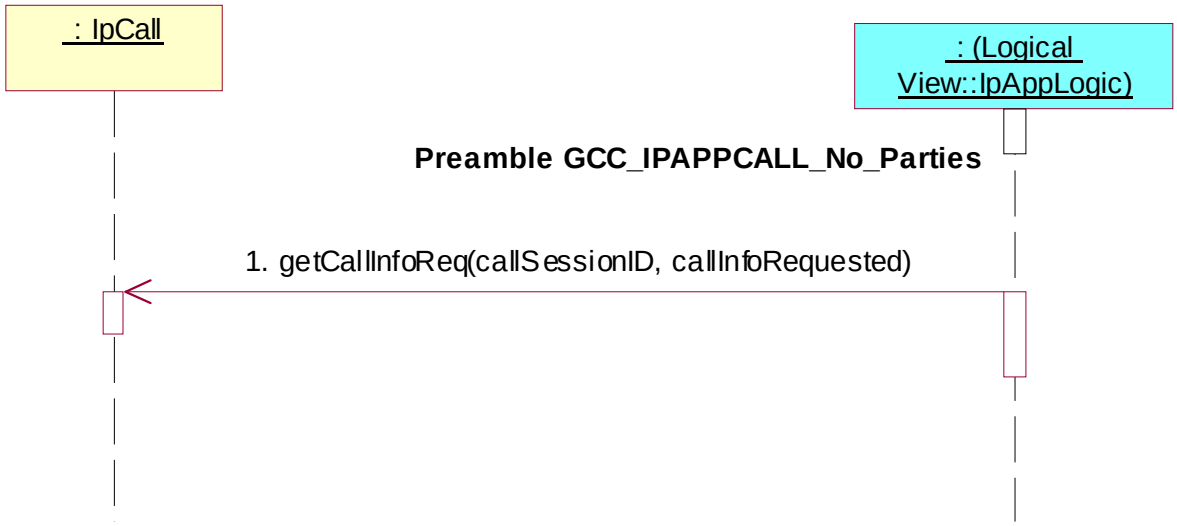
Reference: ES 202 915-4-2 [2], clause 6.3

Precondition: IUT capable of invoking **getCallInfoReq()**

Preamble: GCC_IPAPPCALL_No_Parties

Test Sequence:

1. Triggered Action: cause IUT to call **getCallInfoReq()** method on the tester's (SCF's) IpCall interface.
   Parameters: callSessionID, callInfoRequested



Test GCC_IPAPPCALL_04

Summary: specify charge plan for call

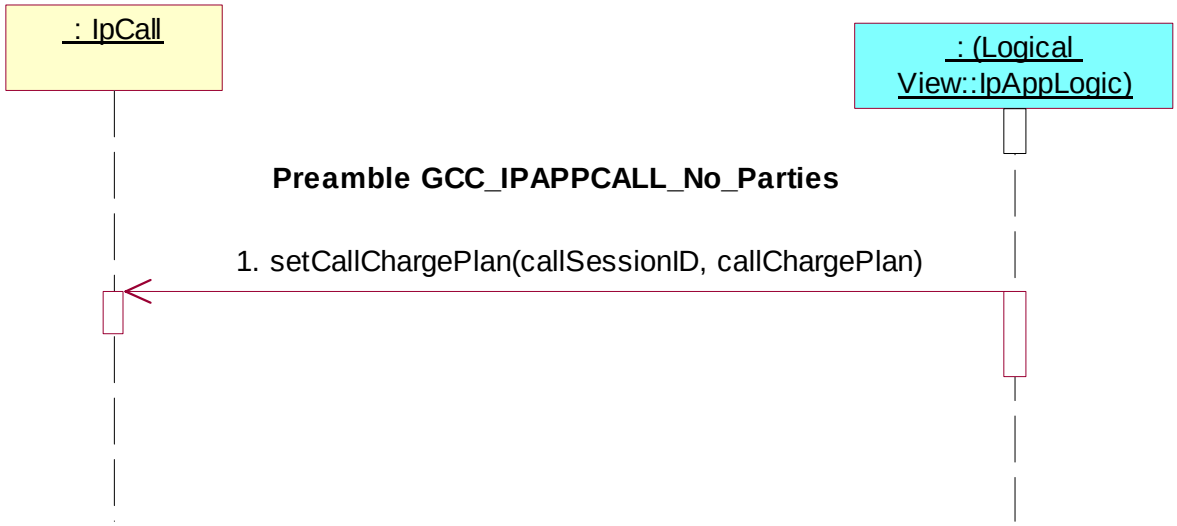
Reference: ES 202 915-4-2 [2], clause 6.3

Precondition: IUT capable of invoking **setCallChargePlan()**

Preamble: GCC_IPAPPCALL_No_Parties

Test Sequence:

1. Triggered Action: cause IUT to call **setCallChargePlan()** method on the tester's (SCF's) IpCall interface.
   Parameters: callSessionID, callChargePlan



Test GCC_IPAPPCALL_05

Summary: allow advice of charge information for call

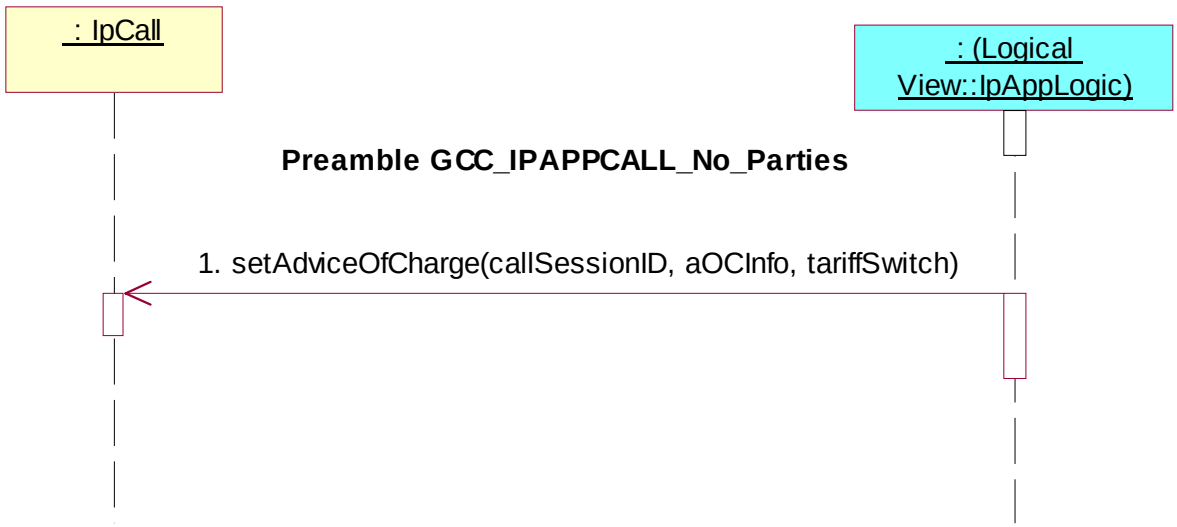
Reference: ES 202 915-4-2 [2], clause 6.3

Precondition: IUT capable of invoking **setAdviceOfCharge()**

Preamble: GCC_IPAPPCALL_No_Parties

Test Sequence:

1. Triggered Action: cause IUT to call **setAdviceOfCharge()** method on the tester's (SCF's) IpCall interface.
   Parameters: callSessionID, aOCInfo, tariffSwitch



7.2.1.2.2 General tests, Active state

NOTE: Tests GCC_IPAPPCALL_06 to 13 do not specify the values of CallEventName field in the eventCriteria parameter of the enableCallNotification() method expected from the IUT. This has been done intentionally to keep these basic tests simple. Note that the tester has to answer with parameter values that make sense in relation to the parameter values received from the IUT.

Precondition: IUT capable of invoking **enableCallNotification()**

Preamble GCC_IPAPPCALL_Active

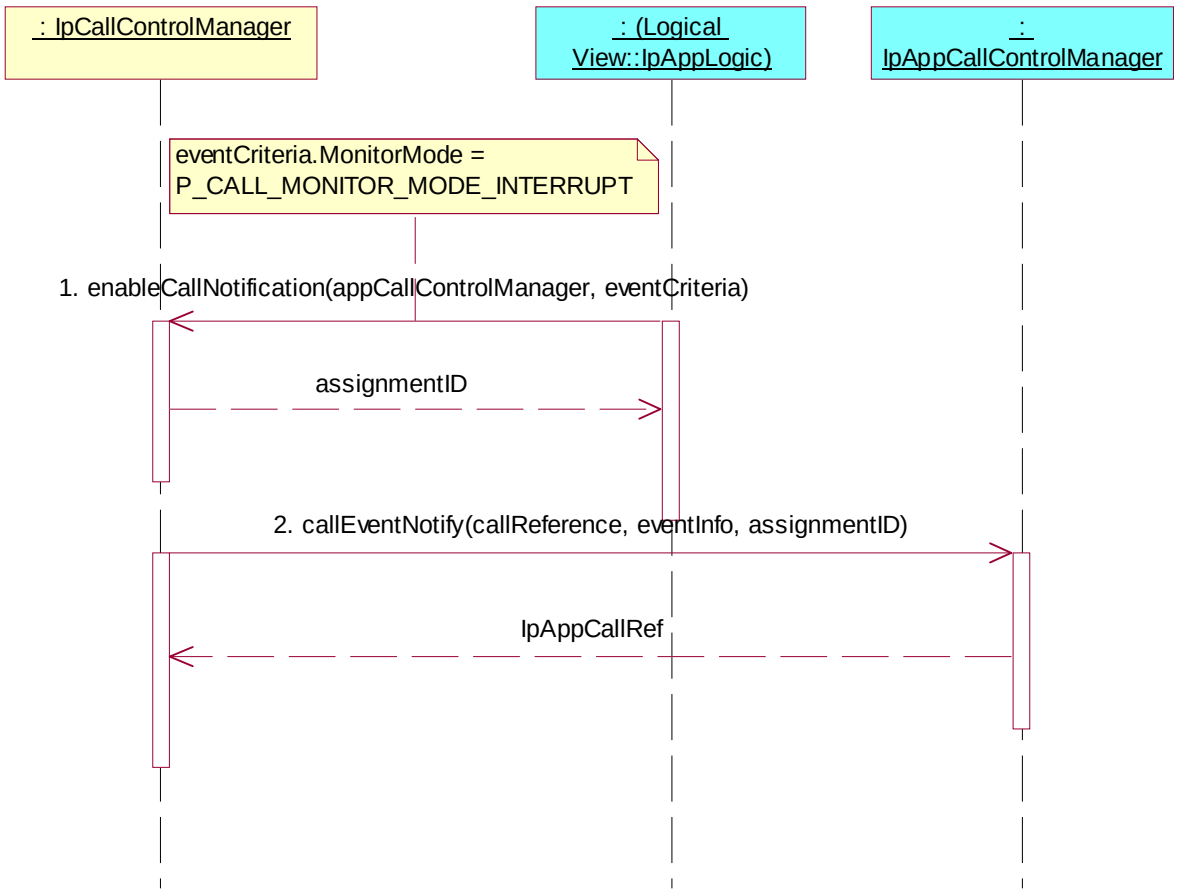
Reference: ES 202 915-4-2 [2], clause 7.2

Pre-preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

1. Triggered Action: cause IUT to call **enableCallNotification()** method on the tester's (SCF's) IpCallControlManager interface.
   Parameters: appCallControlManager, eventCriteria
   eventCriteria.MonitorMode = P_CALL_MONITOR_MODE_INTERRUPT

2. Method call **callEventNotify()**
   Parameters: callReference, eventInfo, assignmentID
   Check: valid value of IpAppCallRef is returned



Test GCC_IPAPPCALL_06

Summary: set supervision of call, successful

Reference: ES 202 915-4-2 [2], clause 6.3

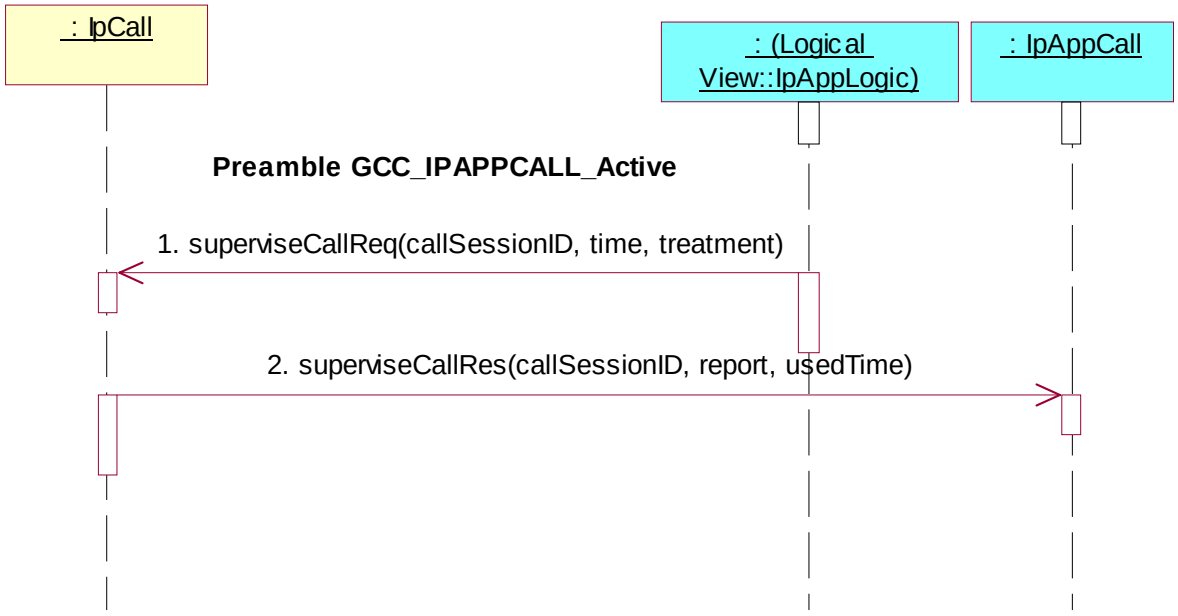
Precondition: IUT capable of invoking **superviseCallReq()**

Preamble: GCC_IPAPPCALL_Active

Test Sequence:

1. Triggered Action: cause IUT to call **superviseCallReq()** method on the tester's (SCF's) IpCall interface.
   Parameters: callSessionID, time, treatment

2. Method call **superviseCallRes()**
   Parameters: callSessionID, report, usedTime
   Check: no exception is returned



Test GCC_IPAPPCALL_07

Summary: set supervision of call, unsuccessful

Reference: ES 202 915-4-2 [2], clause 6.3

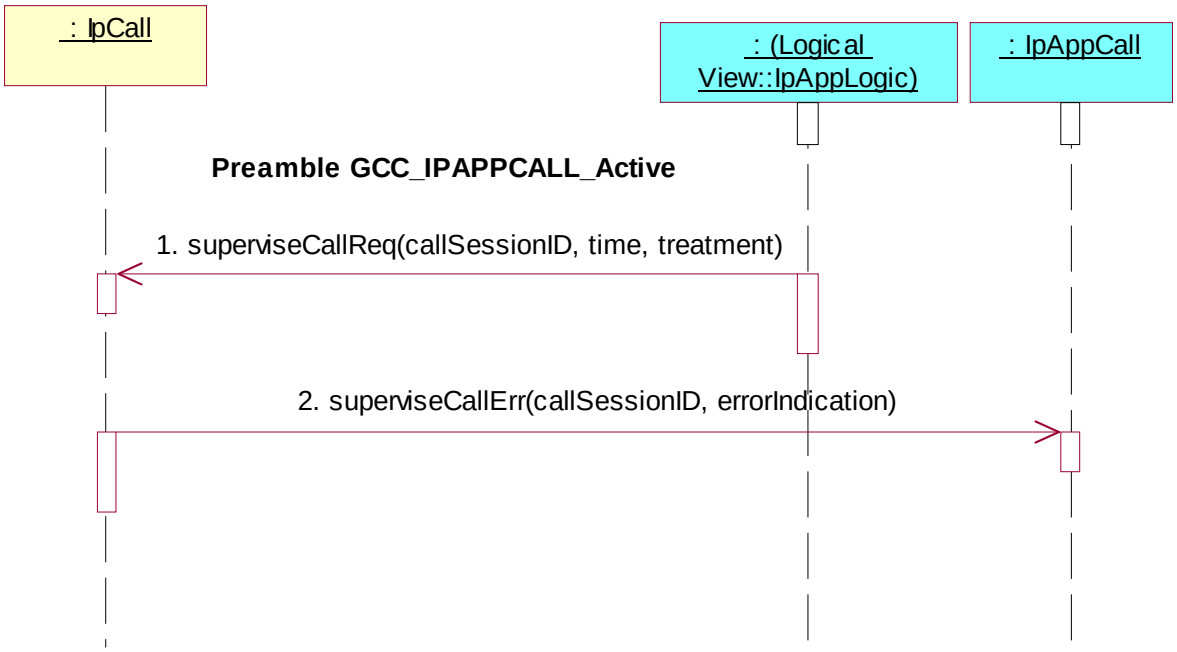
Precondition: IUT capable of invoking **superviseCallReq()**

Preamble: GCC_IPAPPCALL_Active

Test Sequence:

1. Triggered Action: cause IUT to call **superviseCallReq()** method on the tester's (SCF's) IpCall interface.
   Parameters: callSessionID, time, treatment

2. Method call **superviseCallErr()**
   Parameters: callSessionID, errorIndication
   Check: no exception is returned



Test GCC_IPAPPCALL_08

Summary: specify charge plan for call

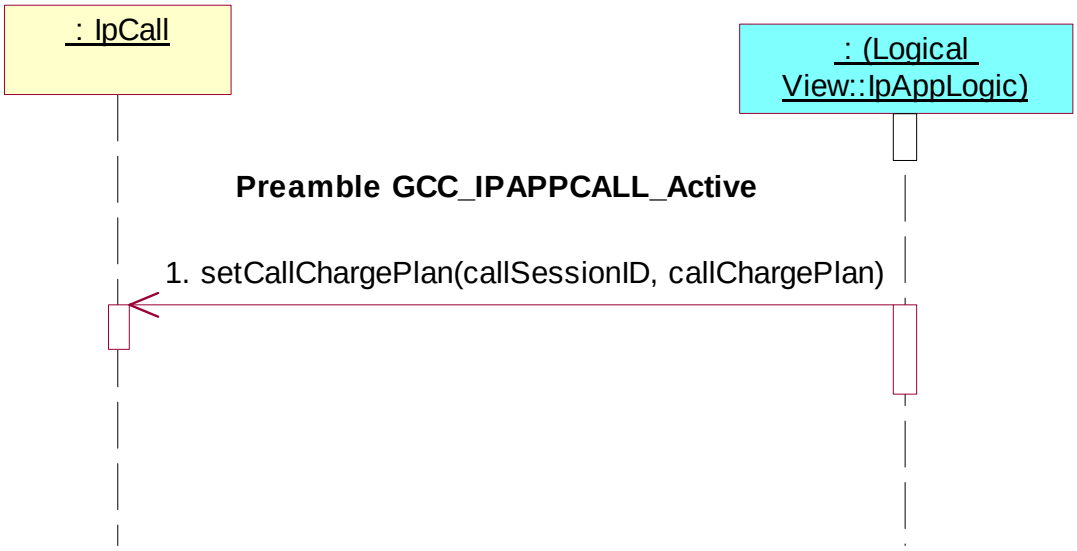
Reference: ES 202 915-4-2 [2], clause 6.3

Precondition: IUT capable of invoking **setCallChargePlan()**

Preamble: GCC_IPAPPCALL_Active

Test Sequence:

1. Triggered Action: cause IUT to call **setCallChargePlan()** method on the tester's (SCF's) IpCall interface.
   Parameters: callSessionID, callChargePlan



Test GCC_IPAPPCALL_09

Summary: allow advice of charge information for call

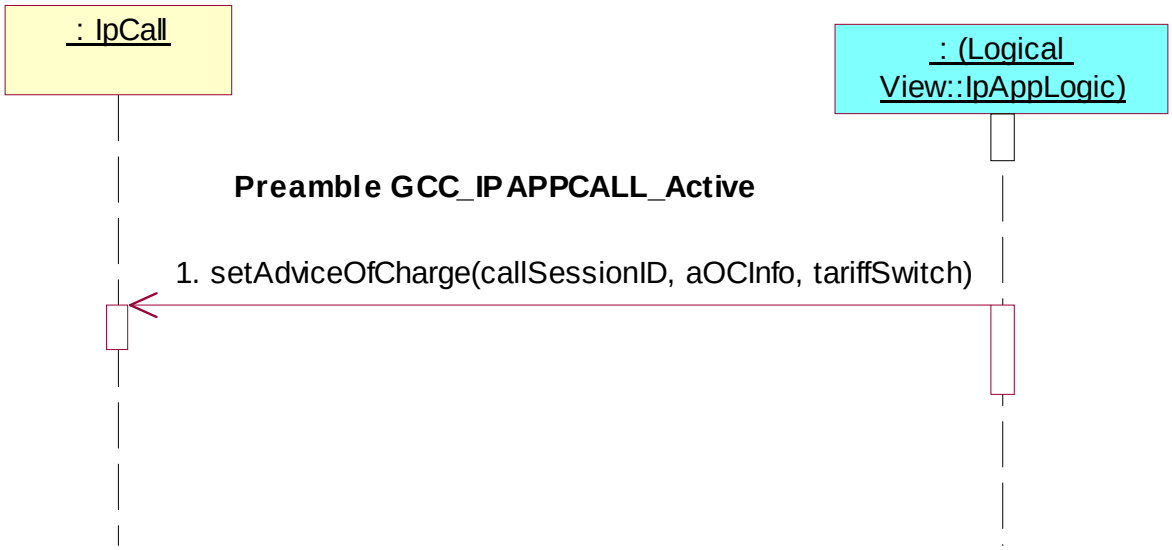
Reference: ES 202 915-4-2 [2], clause 6.3

Precondition: IUT capable of invoking **setAdviceOfCharge()**

Preamble: GCC_IPAPPCALL_Active

Test Sequence:

1. Triggered Action: cause IUT to call **setAdviceOfCharge()** method on the tester's (SCF's) IpCall interface.
   Parameters: callSessionID, aOCInfo, tariffSwitch



Test GCC_IPAPPCALL_10

Summary: release call

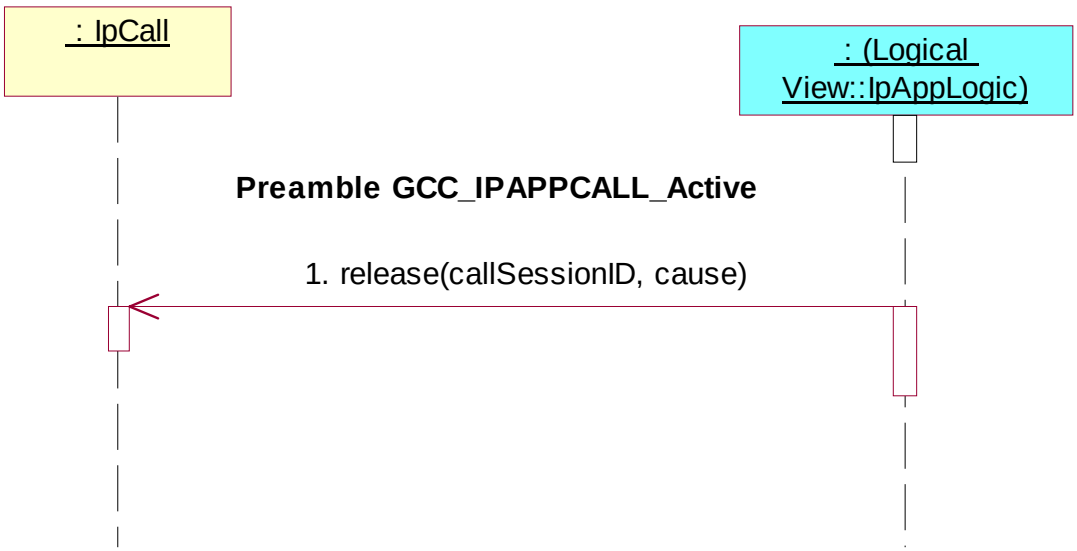
Reference: ES 202 915-4-2 [2], clause 6.3

Precondition: IUT capable of invoking **release()**

Preamble: GCC_IPAPPCALL_Active

Test Sequence:

1. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) IpCall interface.
   Parameters: callSessionID, reason



Test GCC_IPAPPCALL_11

Summary: indication of a call fault

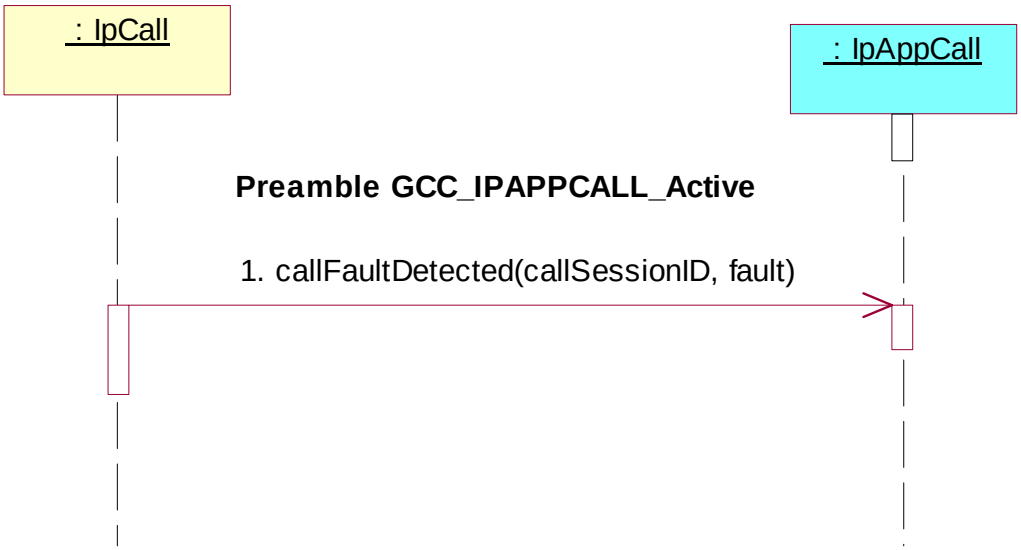
Reference: ES 202 915-4-2 [2], clause 6.4

Precondition: **callFaultDetected()** implemented

Preamble: GCC_IPAPPCALL_Active

Test Sequence:

1. Method call **callFaultDetected()**
   Parameters: callSessionID, fault
   Check: no exception is returned



Test GCC_IPAPPCALL_12

Summary: indication of a call ended

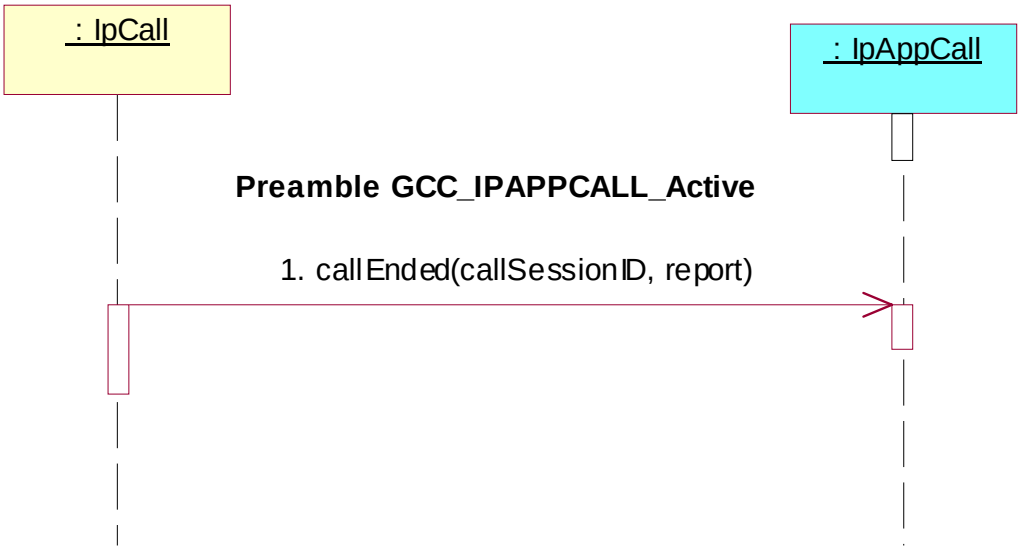
Reference: ES 202 915-4-2 [2], clause 6.4

Precondition: **callEnded()** implemented

Preamble: GCC_IPAPPCALL_Active

Test Sequence:

1. Method call **callEnded()**
   Parameters: callSessionID, report
   Check: no exception is returned



Test GCC_IPAPPCALL_13

Summary: continue processing for call

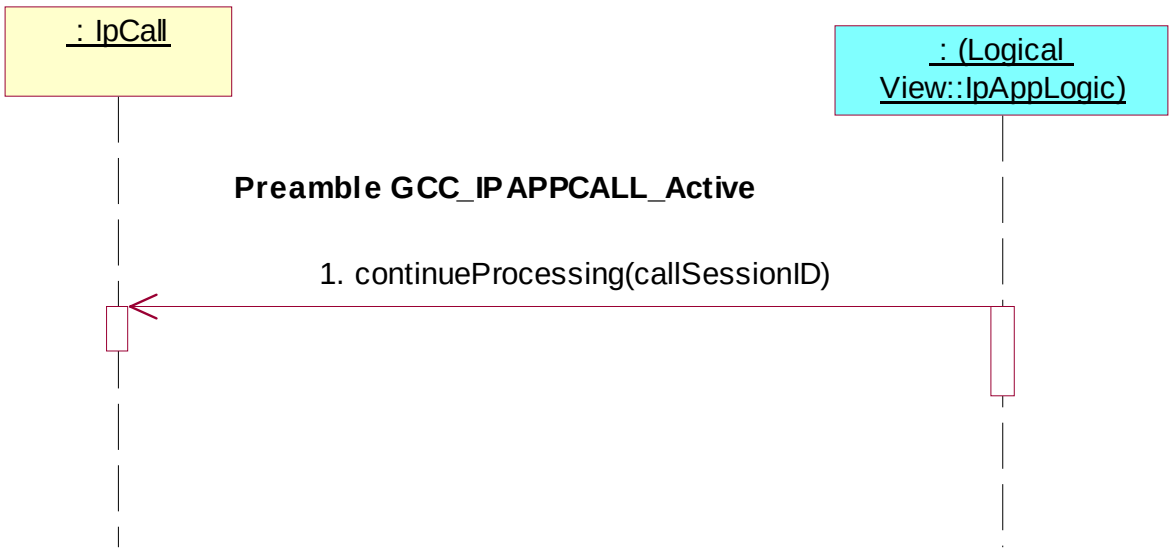
Reference: ES 202 915-4-2 [2], clause 6.3

Precondition: IUT capable of invoking **continueProcessing()**

Preamble: GCC_IPAPPCALL_Active

Test Sequence:

1. Triggered Action: cause IUT to call **continueProcessing()** method on the tester's (SCF's) IpCall interface.
   Parameters: callSessionID



7.2.1.2.3 Active state, Routing to Destination(s) sub-state

Precondition: IUT capable of invoking **createCall()** and **routeReq()**

Preamble GCC_IPAPPCALL_Active_Routing_to_Destination(s)

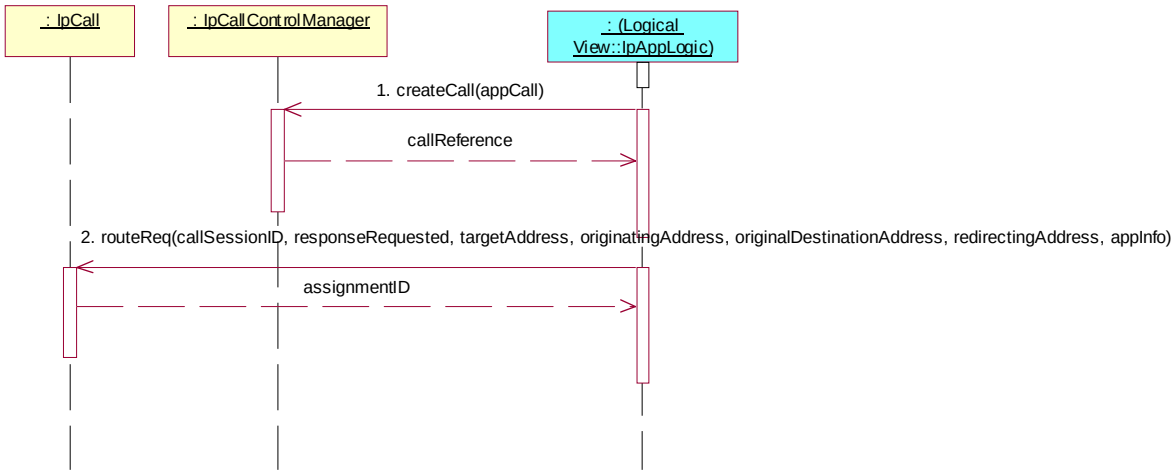
Reference: ES 202 915-4-2 [2], clause 7.2

Pre-preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

1. Triggered Action: cause IUT to call **createCall()** method on the tester's (SCF's) IpCallControlManager interface.
   Parameters: appCall

2. Triggered Action: cause IUT to call **routeReq()** method on the tester's (SCF's) IpCall interface, i.e. connect to call leg to subscriber A.
   Parameters: callSessionID, responseRequested, targetAddress, originatingAddress, originalDestinationAddress, redirectingAddress, appInfo



Test GCC_IPAPPCALL_14

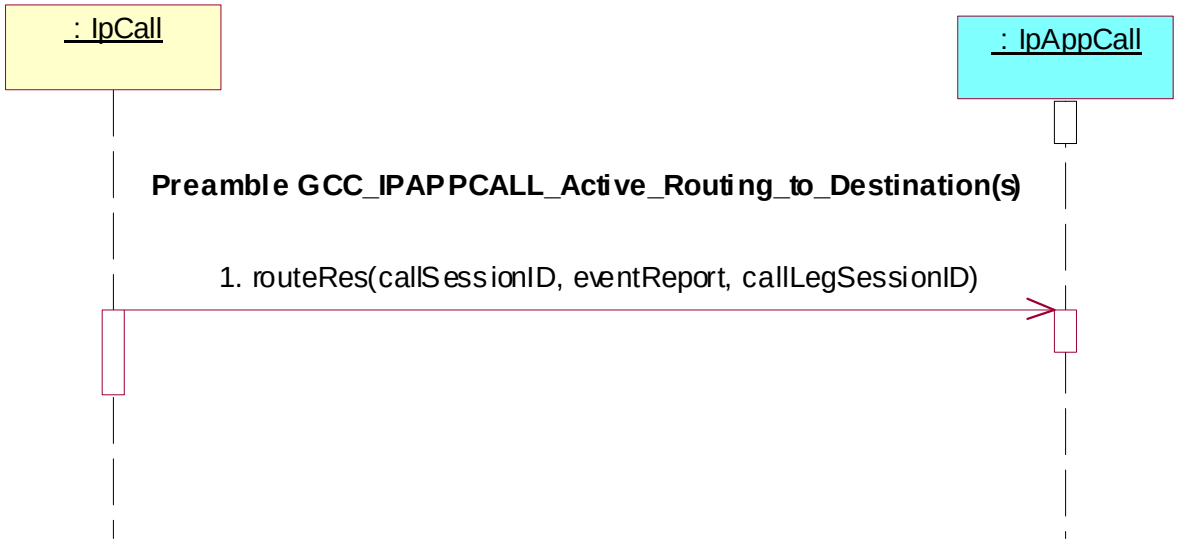
Summary: application initiated call, 1st call leg, successful

Reference: ES 202 915-4-2 [2], clauses 4.3, 6.3 and 6.4

Preamble: GCC_IPAPPCALL_Active_Routing_to_Destination(s)

Test Sequence:

1. Method call **routeRes()**
   Parameters: callSessionId, eventReport, callLegSessionId
   Check: no exception is returned



Test GCC_IPAPPCALL_15

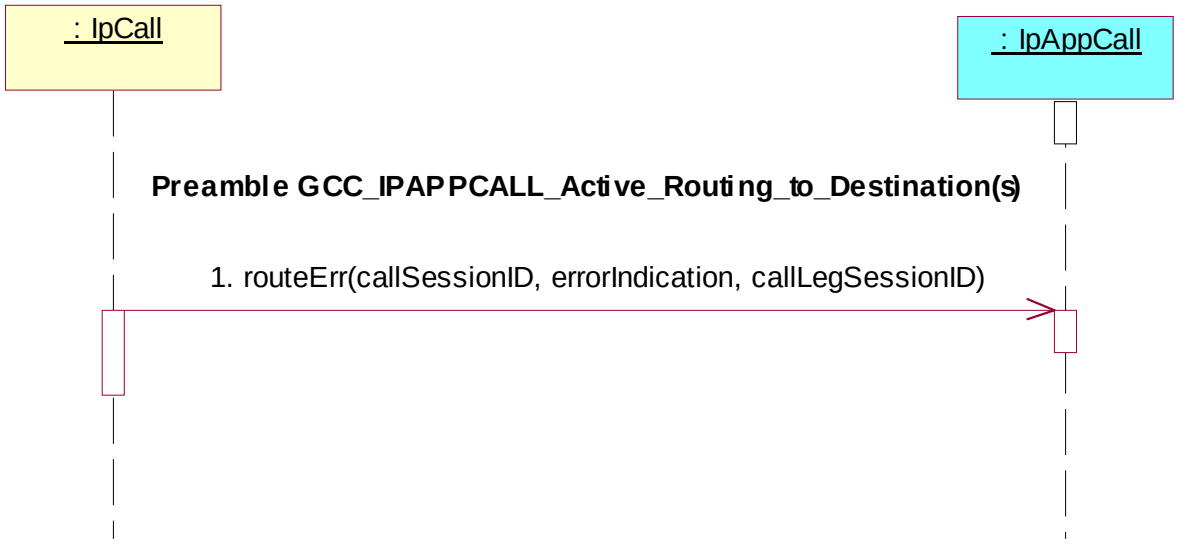
Summary: application initiated call, 1st call leg, unsuccessful

Reference: ES 202 915-4-2 [2], clauses 4.3, 6.3 and 6.4

Preamble: GCC_IPAPPCALL_Active_Routing_to_Destination(s)

Test Sequence:

1. Method call **routeErr()**
   Parameters: callSessionId, errorIndication, callLegSessionId
   Check: no exception is returned



7.2.1.2.4 Active state, 1 Party in Call sub-state

Precondition: IUT capable of invoking **enableCallNotification()** or **createCall()** and **routeReq()**

Preamble GCC_IPAPPCALL_1_Party_in_Call

Reference: ES 202 915-4-2 [2], clause 7.2

Pre-preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

1. Triggered Action: cause IUT to call **enableCallNotification()** method on the tester's (SCF's) IpCallControlManager interface.
   Parameters: appCallControlManager, eventCriteria
   eventCriteria.MonitorMode = P_CALL_MONITOR_MODE_INTERRUPT
   eventCriteria.CallEventName <> P_EVENT_GCCS_ANSWER_FROM_CALL_PARTY

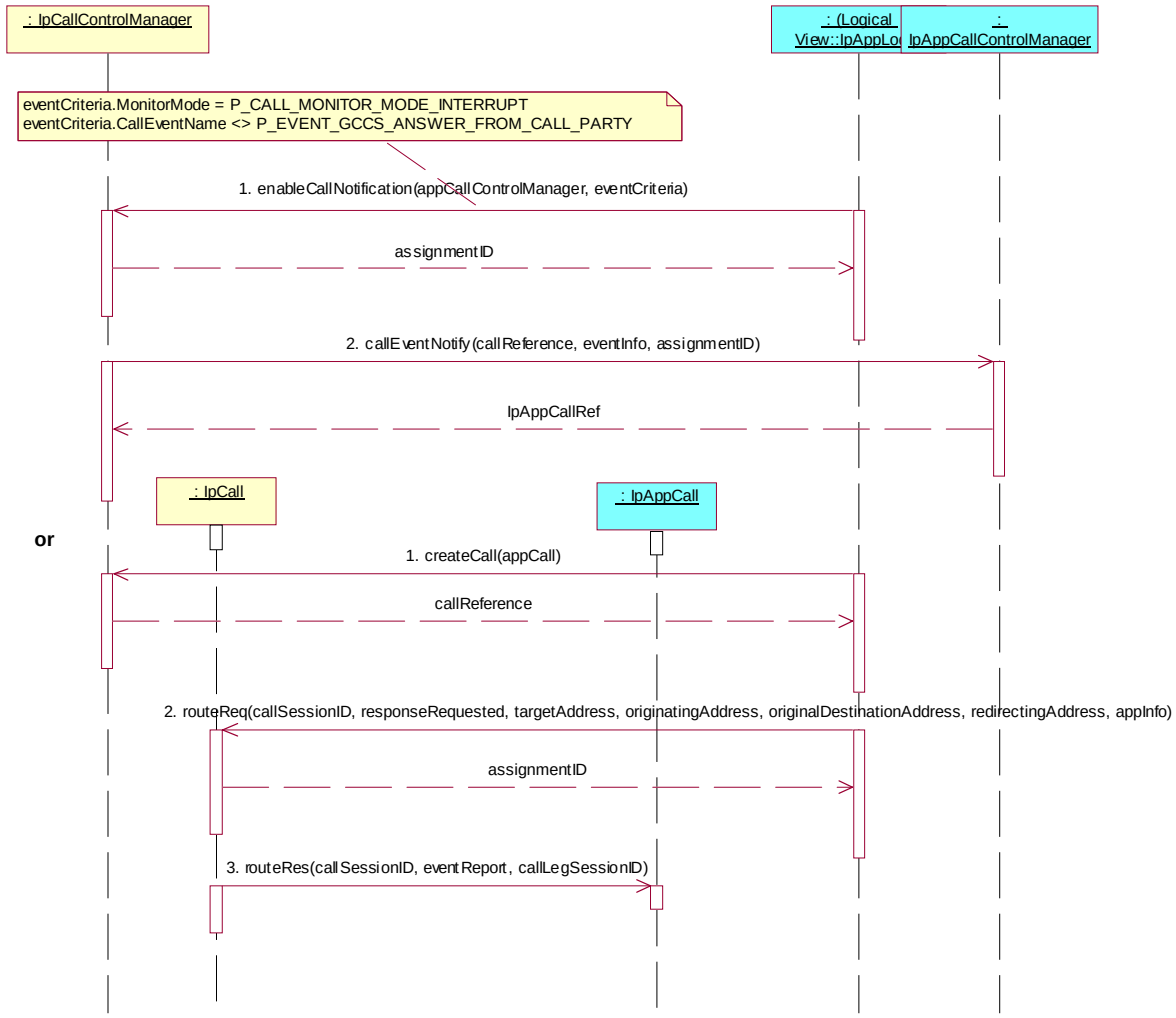
2. Method call **callEventNotify()**
   Parameters: callReference, eventInfo, assignmentID
   Check: valid value of IpAppCallRef is returned

or

1. Triggered Action: cause IUT to call **createCall()** method on the tester's (SCF's) IpCallControlManager interface.
   Parameters: appCall

2. Triggered Action: cause IUT to call **routeReq()** method on the tester's (SCF's) IpCall interface, i.e. connect to call leg to subscriber A.
   Parameters: callSessionID, responseRequested, targetAddress, originatingAddress, originalDestinationAddress, redirectingAddress, appInfo

3. Method call **routeRes()**
   Parameters: callSessionId, eventReport, callLegSessionId
   Check: no exception is returned



Test GCC_IPAPPCALL_16

Summary: connect to 2nd call leg, successful

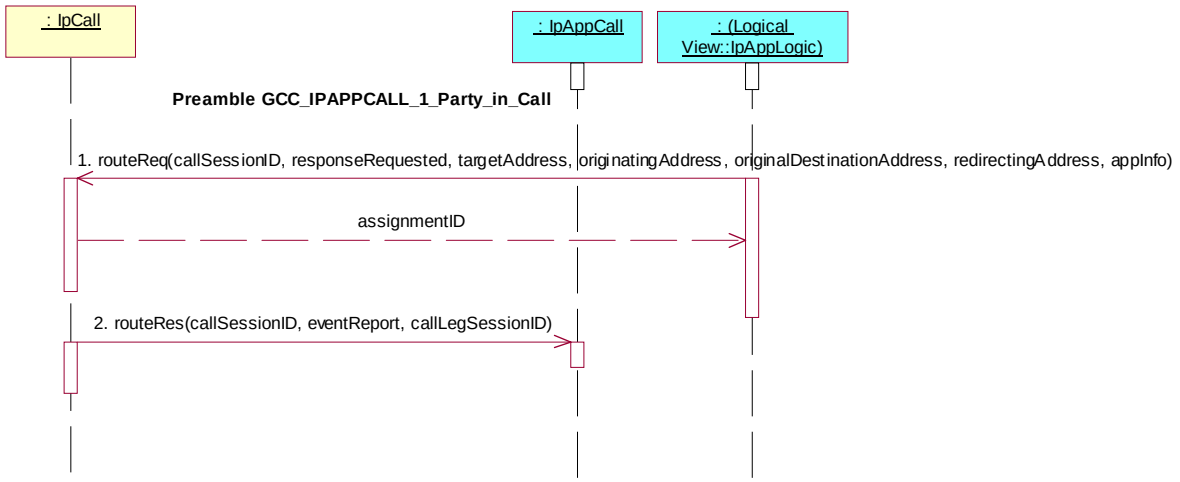
Reference: ES 202 915-4-2 [2], clauses 6.3 and 6.4

Preamble: GCC_IPAPPCALL_1_Party_in_Call

Test Sequence:

1. Triggered Action: cause IUT to call **routeReq()** method on the tester's (SCF's) IpCall interface, i.e. connect to call leg to subscriber B.
   Parameters: callSessionID, responseRequested, targetAddress, originatingAddress, originalDestinationAddress, redirectingAddress, appInfo

2. Method call **routeRes()**
   Parameters: callSessionId, eventReport, callLegSessionId
   Check: no exception is returned



Test GCC_IPAPPCALL_17

Summary: connect to 2nd call leg, unsuccessful

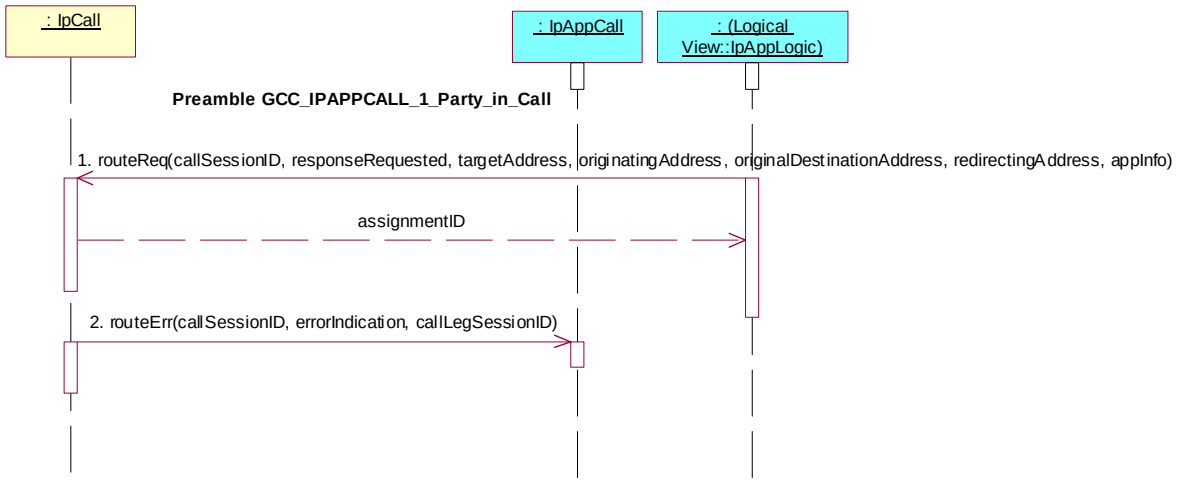
Reference: ES 202 915-4-2 [2], clauses 6.3 and 6.4

Preamble: GCC_IPAPPCALL_1_Party_in_Call

Test Sequence:

1. Triggered Action: cause IUT to call **routeReq()** method on the tester's (SCF's) IpCall interface, i.e. connect to call leg to subscriber B.
   Parameters: callSessionID, responseRequested, targetAddress, originatingAddress, originalDestinationAddress, redirectingAddress, appInfo

2. Method call **routeErr()**
   Parameters: callSessionId, errorIndication, callLegSessionId
   Check: no exception is returned



Test GCC_IPAPPCALL_18

Summary: request information about call

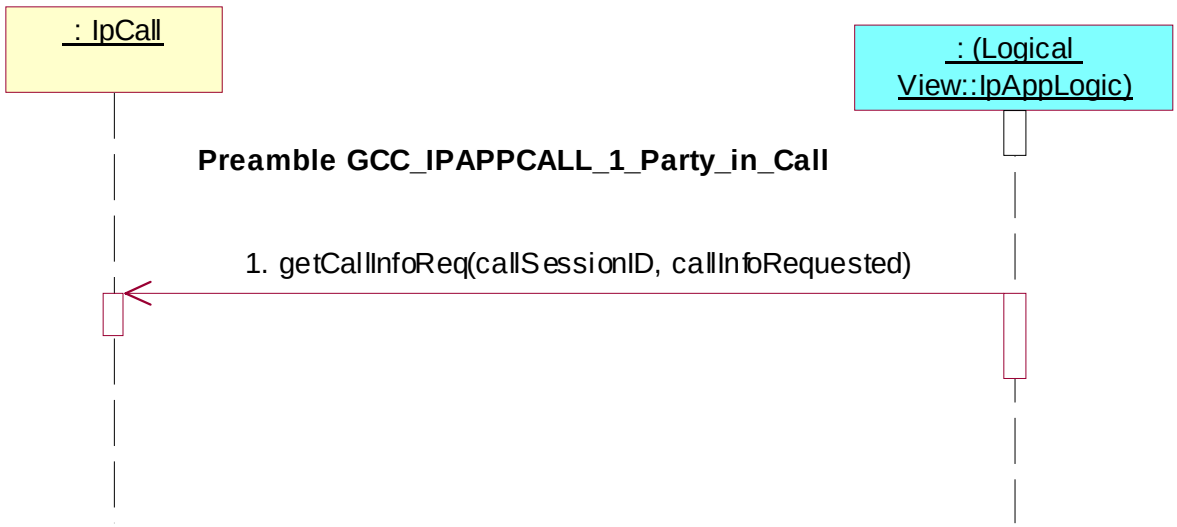
Reference: ES 202 915-4-2 [2], clause 6.3

Precondition: IUT capable of invoking **getCallInfoReq()**

Preamble: GCC_IPAPPCALL_1_Party_in_Call

Test Sequence:

1. Triggered Action: cause IUT to call **getCallInfoReq()** method on the tester's (SCF's) IpCall interface.
   Parameters: callSessionID, callInfoRequested



Test GCC_IPAPPCALL_19

Summary: specify charge plan for call

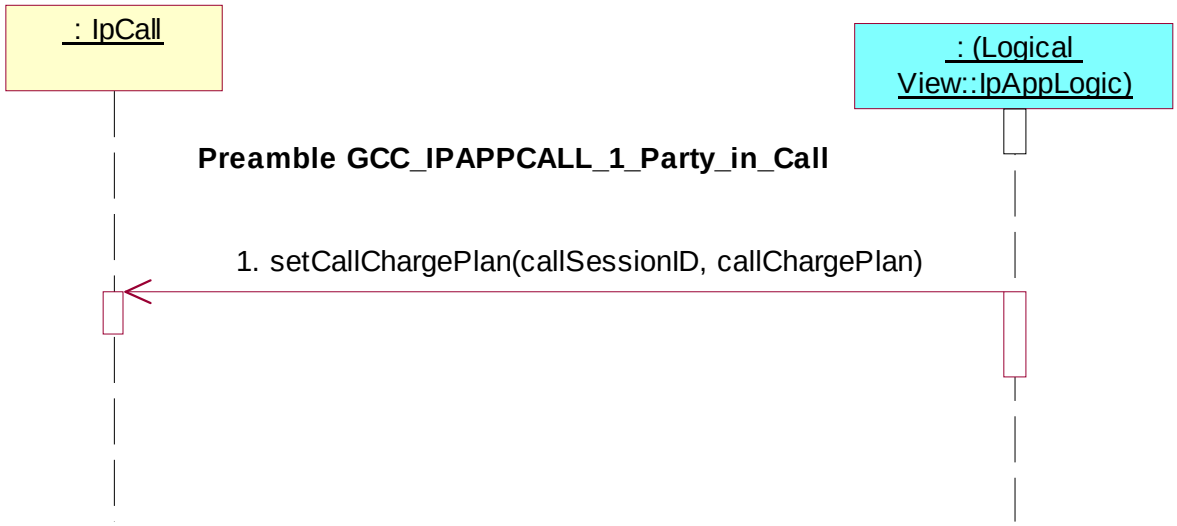
Reference: ES 202 915-4-2 [2], clause 6.3

Precondition: IUT capable of invoking **setCallChargePlan()**

Preamble: GCC_IPAPPCALL_1_Party_in_Call

Test Sequence:

1. Triggered Action: cause IUT to call **setCallChargePlan()** method on the tester's (SCF's) IpCall interface.
   Parameters: callSessionID, callChargePlan



Test GCC_IPAPPCALL_20

Summary: request further digits for call, successful

Reference: ES 202 915-4-2 [2], clauses 6.3 and 6.4

Precondition: Call originated from the network, IUT capable of invoking **getMoreDialledDigitsReq()**

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

The application is permitted to provide its IpAppCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

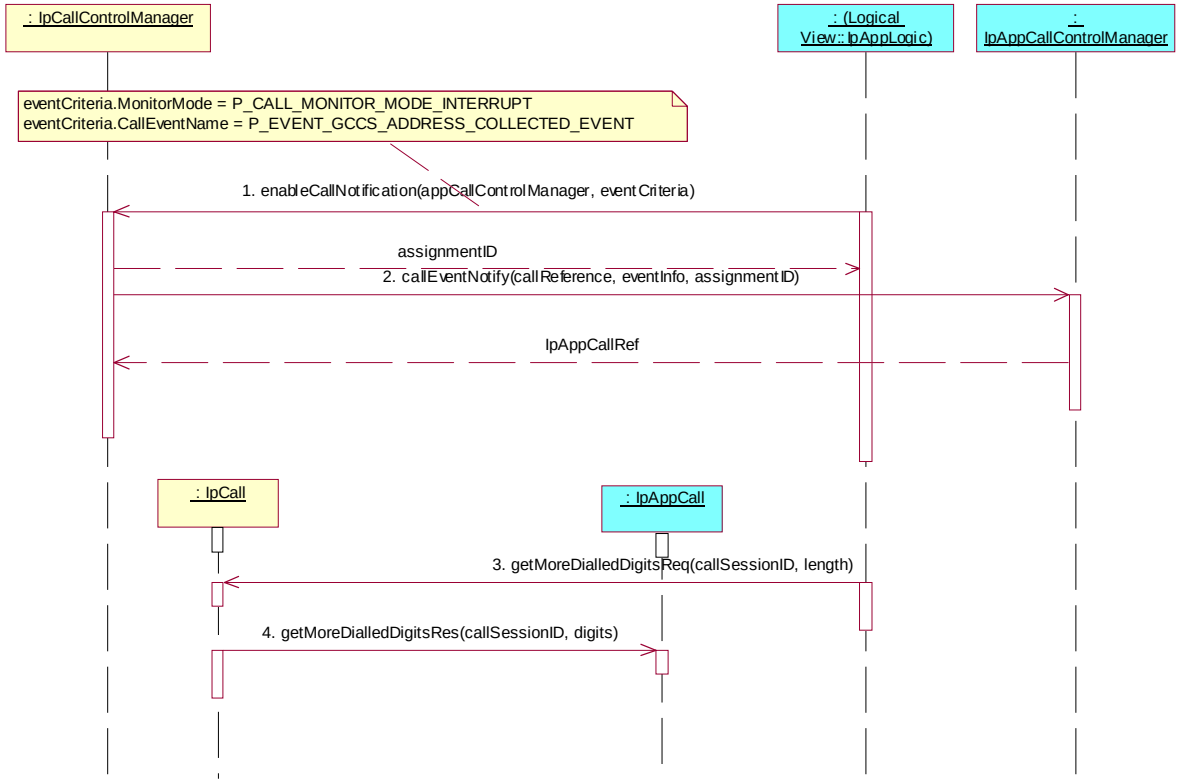
Test Sequence:

1. Triggered Action: cause IUT to call **enableCallNotification()** method on the tester's (SCF's) IpCallControlManager interface.
   Parameters: appCallControlManager, eventCriteria
   eventCriteria.MonitorMode = P_CALL_MONITOR_MODE_INTERRUPT
   eventCriteria.CallEventName = P_EVENT_GCCS_ADDRESS_COLLECTED_EVENT

2. Method call **callEventNotify()**
   Parameters: callReference, eventInfo, assignmentID
   Check: valid value of IpAppCallRef is returned

3. Triggered Action: cause IUT to call **getMoreDialledDigitsReq()** method on the tester's (SCF's) IpCall interface.
   Parameters: callSessionID, length

4. Method call **getMoreDialledDigitsRes()**
   Parameters: callSessionId, digits
   Check: no exception is returned



Test GCC_IPAPPCALL_21

Summary: request further digits for call, unsuccessful

Reference: ES 202 915-4-2 [2], clauses 6.3 and 6.4

Precondition: Call originated from the network, IUT capable of invoking **getMoreDialledDigitsReq()**

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

The application is permitted to provide its IpAppCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

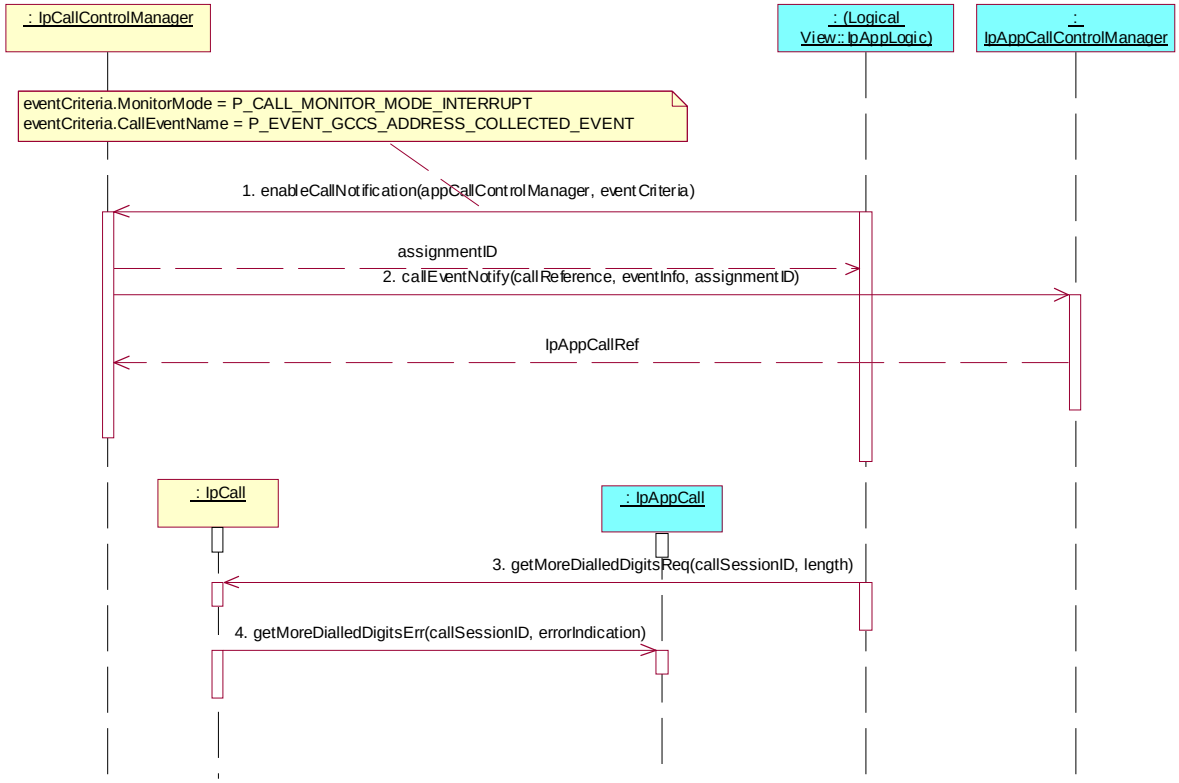
Test Sequence:

1. Triggered Action: cause IUT to call **enableCallNotification()** method on the tester's (SCF's) IpCallControlManager interface.
   Parameters: appCallControlManager, eventCriteria
   eventCriteria.MonitorMode = P_CALL_MONITOR_MODE_INTERRUPT
   eventCriteria.CallEventName = P_EVENT_GCCS_ADDRESS_COLLECTED_EVENT

2. Method call **callEventNotify()**
   Parameters: callReference, eventInfo, assignmentID
   Check: valid value of IpAppCallRef is returned

3. Triggered Action: cause IUT to call **getMoreDialledDigitsReq()** method on the tester's (SCF's) IpCall interface.
   Parameters: callSessionID, length

4. Method call **getMoreDialledDigitsErr()**
   Parameters: callSessionId, errorIndication
   Check: no exception is returned



7.2.1.2.5 Active state, 2 Parties in Call sub-state

Precondition: IUT capable of invoking **enableCallNotification()** or **createCall()** and **routeReq()**

Preamble GCC_IPAPPCALL_2_Parties_in_Call

Reference: ES 202 915-4-2 [2], clause 7.2

Pre-preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

1. Triggered Action: cause IUT to call **enableCallNotification()** method on the tester's (SCF's) IpCallControlManager interface.
   Parameters: appCallControlManager, eventCriteria
   eventCriteria.MonitorMode = P_CALL_MONITOR_MODE_INTERRUPT
   eventCriteria.CallEventName = P_EVENT_GCCS_ANSWER_FROM_CALL_PARTY

2. Method call **callEventNotify()**
   Parameters: callReference, eventInfo, assignmentID
   Check: valid value of IpAppCallRef is returned

or

1. Triggered Action: cause IUT to call **enableCallNotification()** method on the tester's (SCF's) IpCallControlManager interface.
   Parameters: appCallControlManager, eventCriteria
   eventCriteria.MonitorMode = P_CALL_MONITOR_MODE_INTERRUPT
   eventCriteria.CallEventName <> P_EVENT_GCCS_ANSWER_FROM_CALL_PARTY

2. Method call **callEventNotify()**
   Parameters: callReference, eventInfo, assignmentID
   Check: valid value of IpAppCallRef is returned

3. Triggered Action: cause IUT to call **routeReq()** method on the tester's (SCF's) IpCall interface, i.e. connect to call leg to subscriber B.
   Parameters: callSessionID, responseRequested, targetAddress, originatingAddress, originalDestinationAddress, redirectingAddress, appInfo
   responseRequested.CallReportType = P_CALL_REPORT_ANSWER

4. Method call **routeRes()**
   Parameters: callSessionId, eventReport, callLegSessionId
   Check: no exception is returned

or

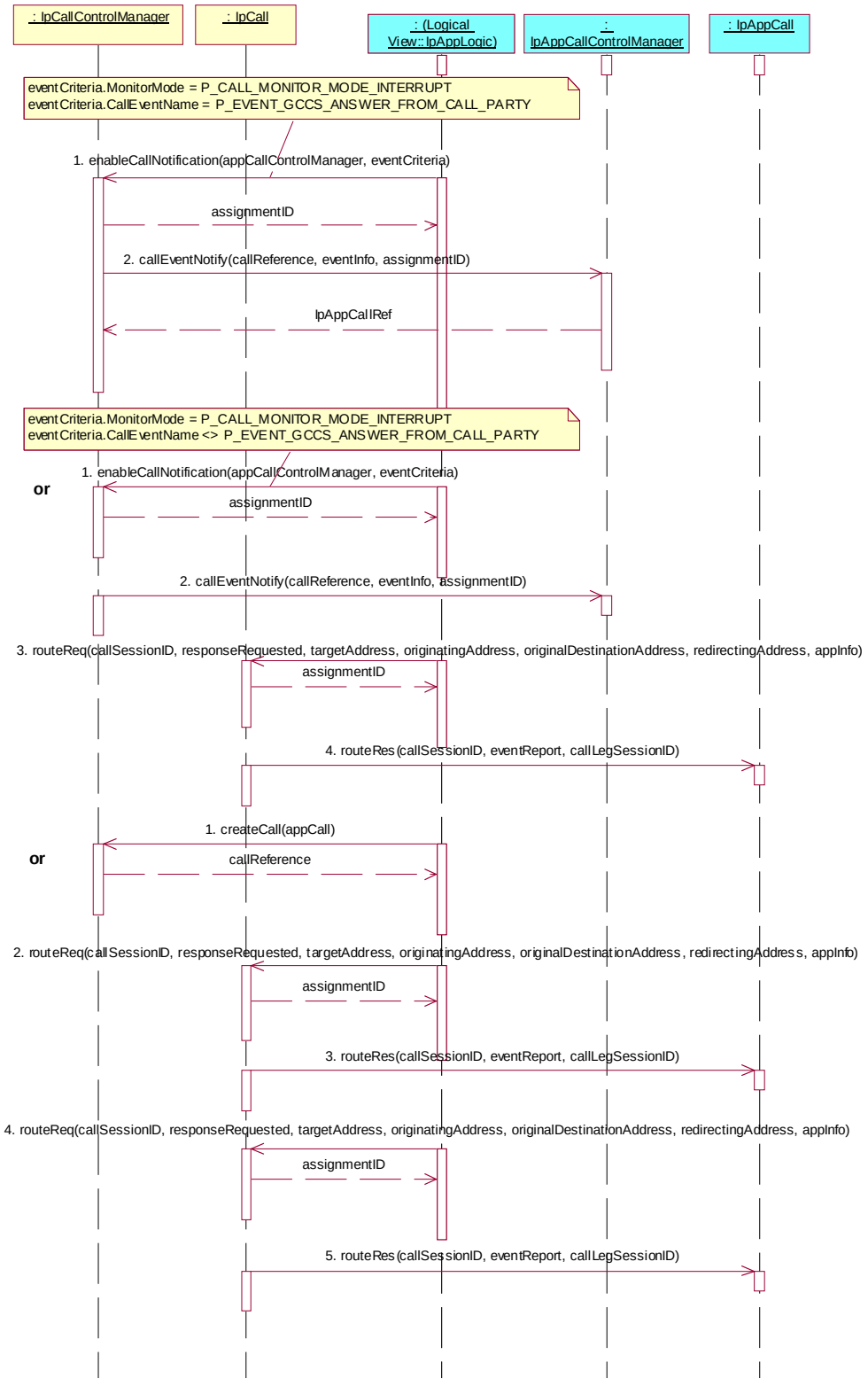
1. Triggered Action: cause IUT to call **createCall()** method on the tester's (SCF's) IpCallControlManager interface.
   Parameters: appCall

2. Triggered Action: cause IUT to call **routeReq()** method on the tester's (SCF's) IpCall interface, i.e. connect to call leg to subscriber A.
   Parameters: callSessionID, responseRequested, targetAddress, originatingAddress, originalDestinationAddress, redirectingAddress, appInfo
   responseRequested.CallReportType = P_CALL_REPORT_ANSWER

3. Method call **routeRes()**
   Parameters: callSessionId, eventReport, callLegSessionId
   Check: no exception is returned

4. Triggered Action: cause IUT to call **routeReq()** method on the tester's (SCF's) IpCall interface, i.e. connect to call leg to subscriber B.
   Parameters: callSessionID, responseRequested, targetAddress, originatingAddress, originalDestinationAddress, redirectingAddress, appInfo
   responseRequested.CallReportType = P_CALL_REPORT_ANSWER

5. Method call **routeRes()**
   Parameters: callSessionId, eventReport, callLegSessionId
   Check: no exception is returned



NOTE: No tests specific to this state have been defined.

7.2.1.2.6 Network Released and Finished state

NOTE: The following test is testing both the Network Released and the Finished state, as there is a direct state transition from the first to the latter state, when no supervision request has been sent by the application or when a sent supervision request is answered by the SCF.

Test GCC_IPAPPCALL_22

Summary: de-assign call that has been released by the network for reasons of fault

Reference: ES 202 915-4-2 [2], clauses 6.3 and 6.4

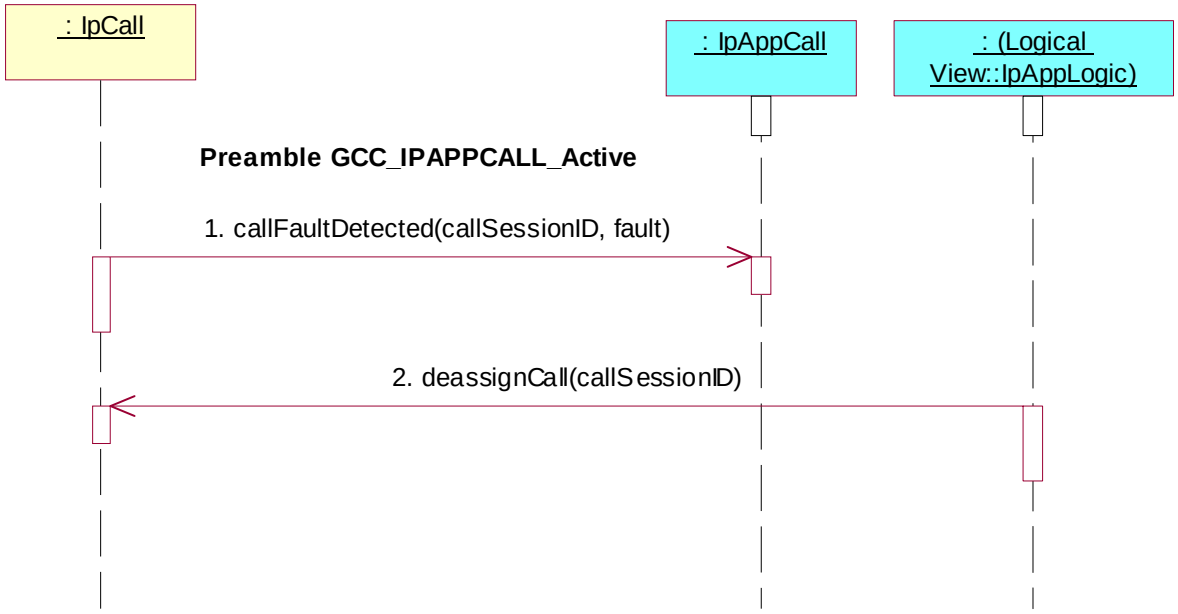
Precondition: IUT capable of invoking **deassignCall()**

Preamble: GCC_IPAPPCALL_Active

Test Sequence:

1. Method call **callFaultDetected()**
   Parameters: callSessionID, fault
   Check: no exception is returned

2. Triggered Action: cause IUT to call **deassignCall()** method on the tester's (SCF's) IpCall interface.
   Parameters: callSessionID



Test GCC_IPAPPCALL_23

Summary: de-assign network released call **after** receipt of supervision result

Reference: ES 202 915-4-2 [2], clauses 6.3 and 6.4

Precondition: IUT capable of invoking **superviseCallReq()** and **deassignCall()**

Preamble: GCC_IPAPPCALL_Active

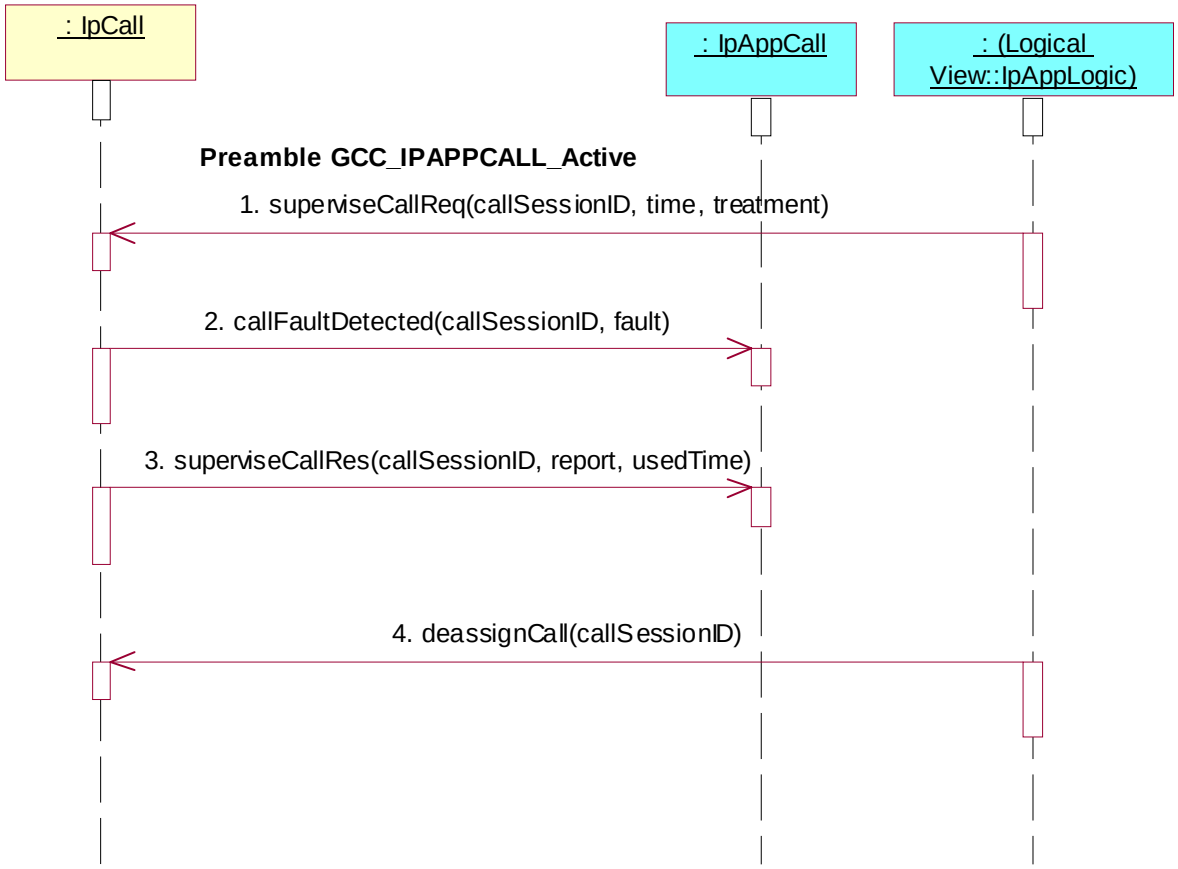
Test Sequence:

1. Triggered Action: cause IUT to call **superviseCallReq()** method on the tester's (SCF's) IpCall interface.
   Parameters: callSessionID, time, treatment

2. Method call **callFaultDetected()**
   Parameters: callSessionID, fault
   Check: no exception is returned

3. Method call **superviseCallRes()**
   Parameters: callSessionID, report, usedTime
   Check: no exception is returned

4. Triggered Action: cause IUT to call **deassignCall()** method on the tester's (SCF's) IpCall interface.
   Parameters: callSessionID



7.2.1.2.7 Application Released state

Test GCC_IPAPPCALL_24

Summary: accept receipt of supervision result **after** release of call by application

Reference: ES 202 915-4-2 [2], clauses 6.3 and 6.4

Precondition: IUT capable of invoking **superviseCallReq()** and **release()**

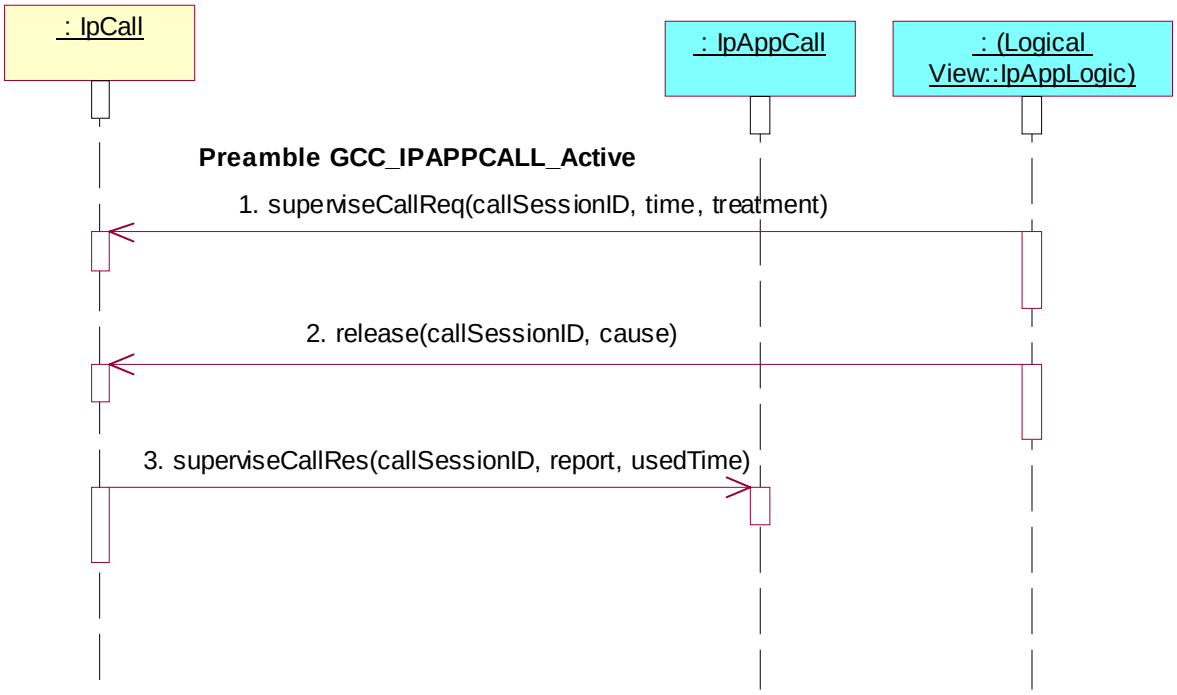
Preamble: GCC_IPAPPCALL_Active

Test Sequence:

1. Triggered Action: cause IUT to call **superviseCallReq()** method on the tester's (SCF's) IpCall interface.
   Parameters: callSessionID, time, treatment

2. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) IpCall interface.
   Parameters: callSessionID, cause

3. Method call **superviseCallRes()**
   Parameters: callSessionID, report, usedTime
   Check: no exception is returned



7.2.2 MultiParty Call Control Service (MPCC)

The TPs in this clause are based on ES 202 915-4-3 [3].

7.2.2.1 IpAppMultiPartyCallControlManager

Test MPCC_IpAppMultiPartyCallControlManager_01

Summary: create call object

Reference: ES 202 915-4-3 [3], clause 6.1

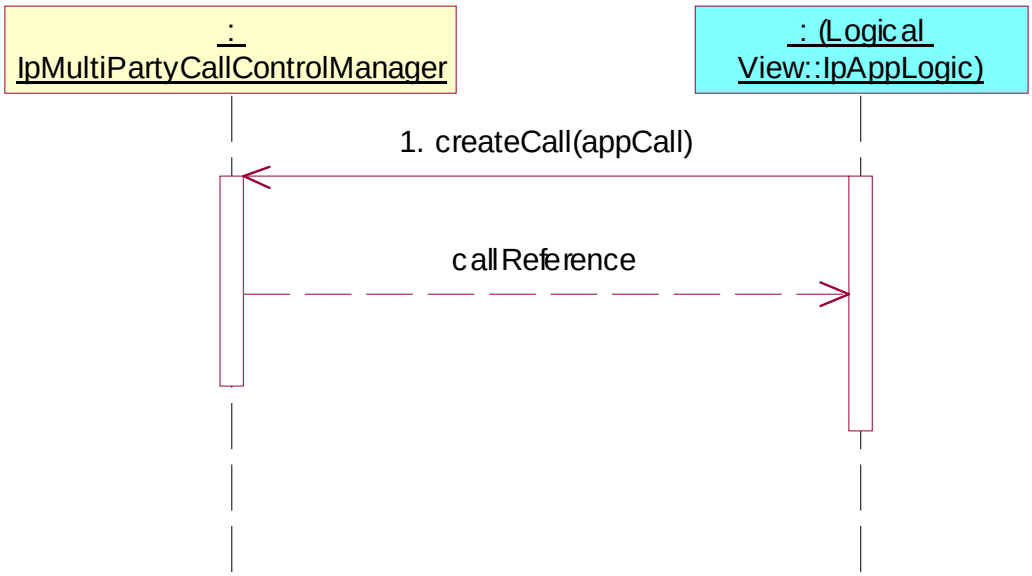
Precondition: IUT capable of invoking **createCall()**

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

The application is permitted to provide its IpAppMultiPartyCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **createCall()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCall



Test MPCC_IpAppMultiPartyCallControlManager_02

Summary: create call object and accept abort

Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.2

Precondition: IUT capable of invoking **createCall()**; **callAborted()** implemented

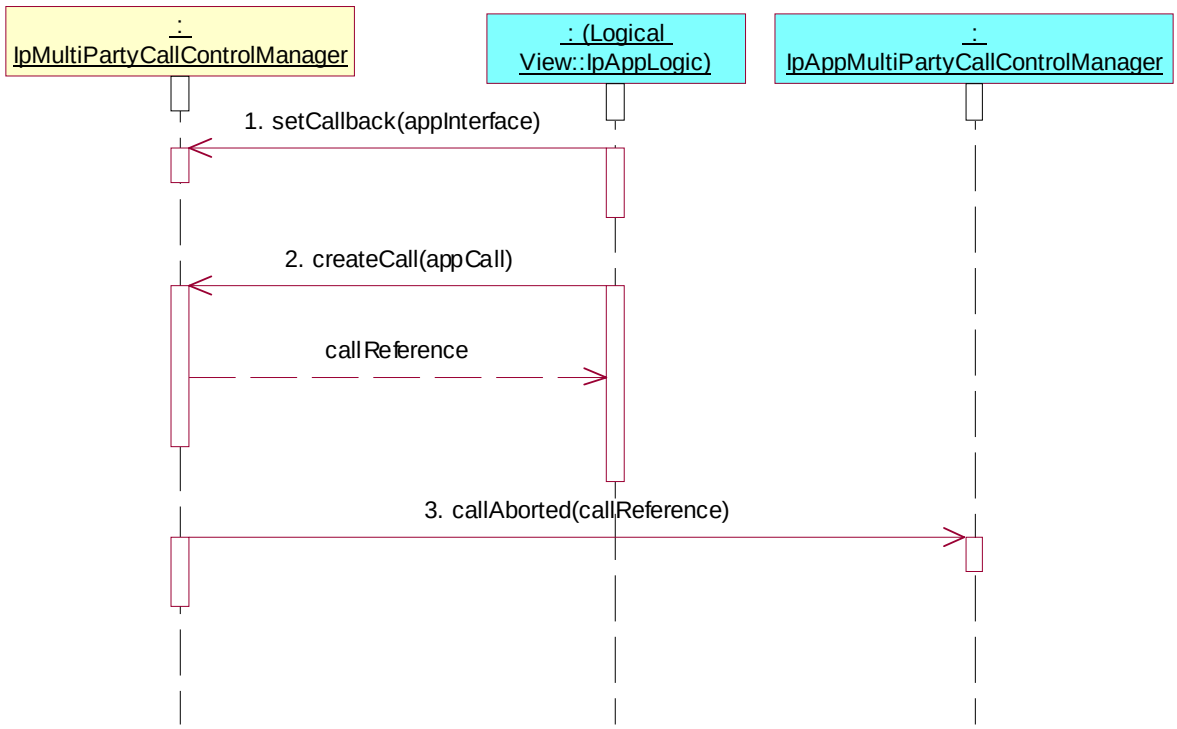
Preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiPartyCallControlManager interface through selecting that service and signing the required service agreement.

Test Sequence:

1. Triggered Action: cause IUT to call **setCallBack()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: valid, non NULL, value of appInterface

2. Triggered Action: cause IUT to call **createCall()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCall

3. Method call **callAborted()**
   Parameters: callReference
   Check: no exception is returned



Test MPCC_IpAppMultiPartyCallControlManager_03

Summary: enable and accept call notifications

Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.2

Precondition: IUT capable of invoking **createNotification()**, **reportNotification()** implemented

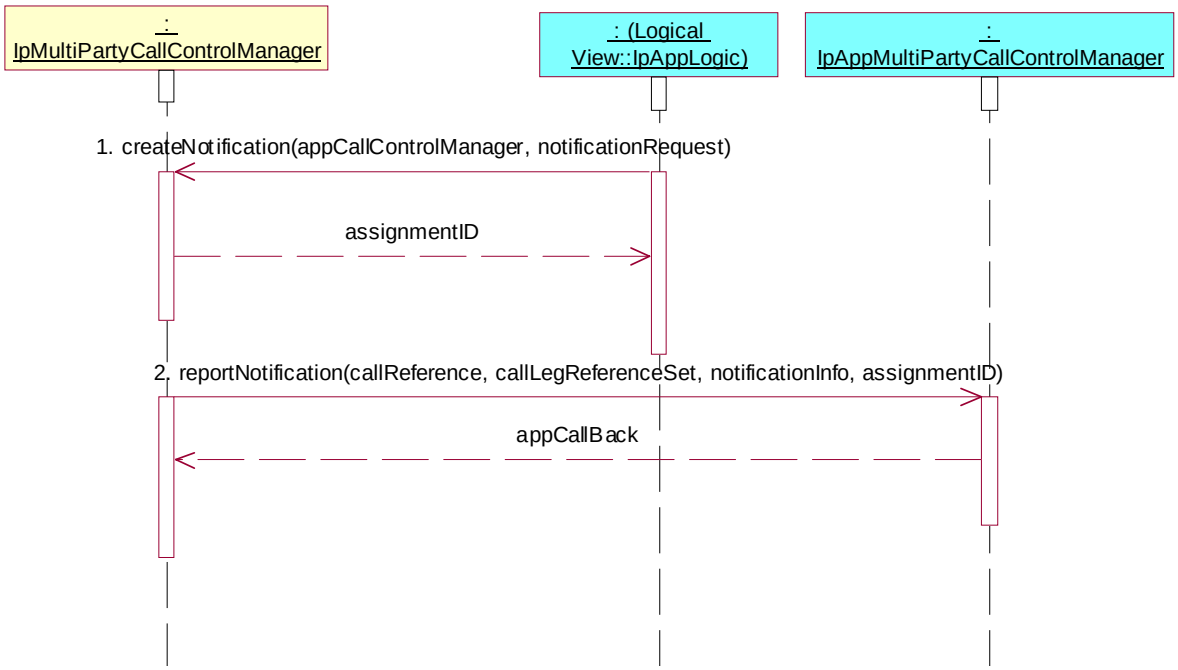
Preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiPartyCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiPartyCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   Check: valid value of TpAppMultiPartyCallBack is returned



Test MPCC_IpAppMultiPartyCallControlManager_04

Summary: interrupt and continue call notifications

Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.2

Precondition: IUT capable of invoking **createNotification()**, **managerInterrupted(), managerContinued()** and **reportNotification()** implemented

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

The application is permitted to provide its IpAppMultiPartyCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest

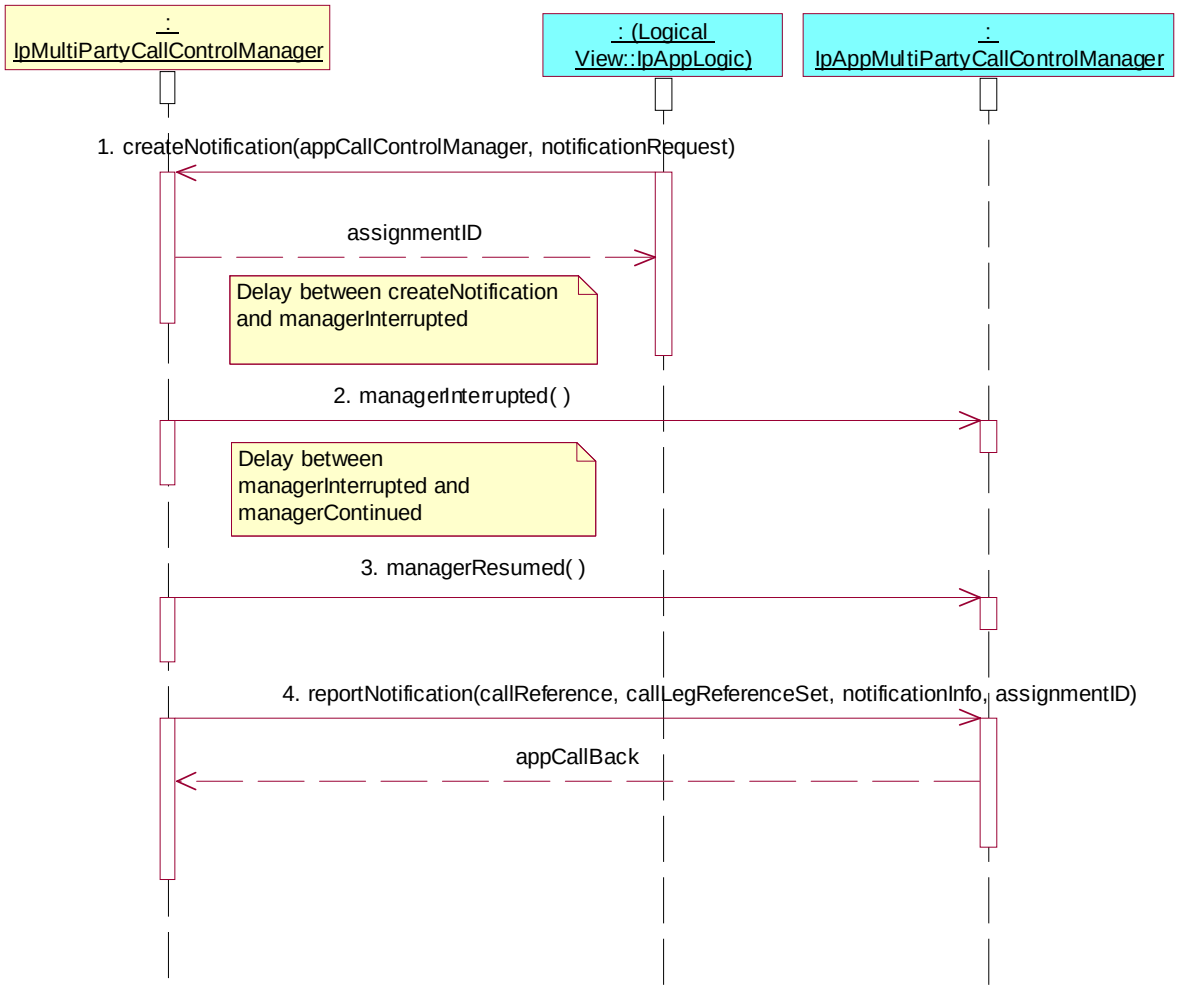
Delay between createNotification and managerInterrupted

2. Method call **managerInterrupted()**
   Parameters: none
   Check: no exception is returned

Delay between managerInterrupted and managerContinued

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

4. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   Check: valid value of TpAppMultiPartyCallBack is returned



Test MPCC_IpAppMultiPartyCallControlManager_05

Summary: enable and change call notifications

Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.2

Precondition: IUT capable of invoking **createNotification()** and **changeNotification()**, **reportNotification()** implemented

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

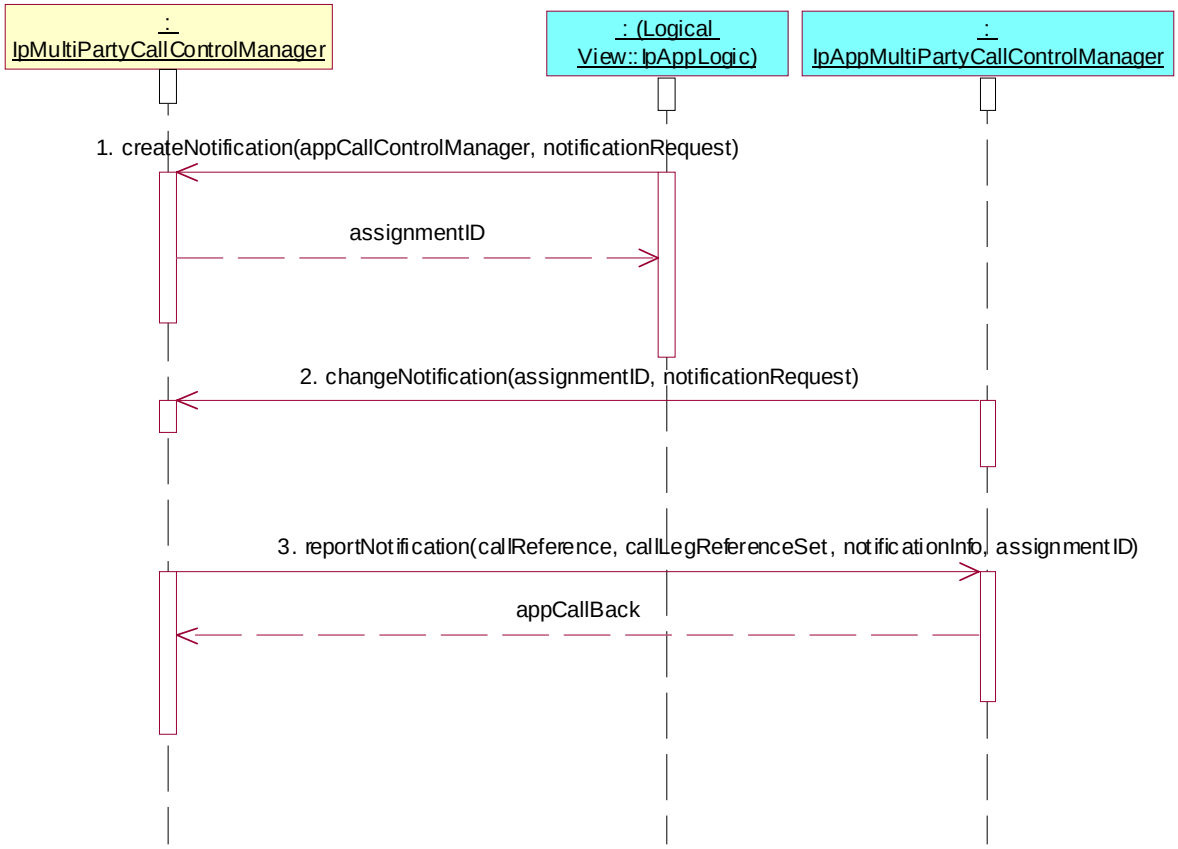
The application is permitted to provide its IpAppMultiPartyCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest

2. Triggered Action: cause IUT to call **changeNotification()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: assignmentID, notificationRequest

3. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   Check: valid value of TpAppMultiPartyCallBack is returned



Test MPCC_IpAppMultiPartyCallControlManager_06

Summary: enable and destroy call notifications

Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.2

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

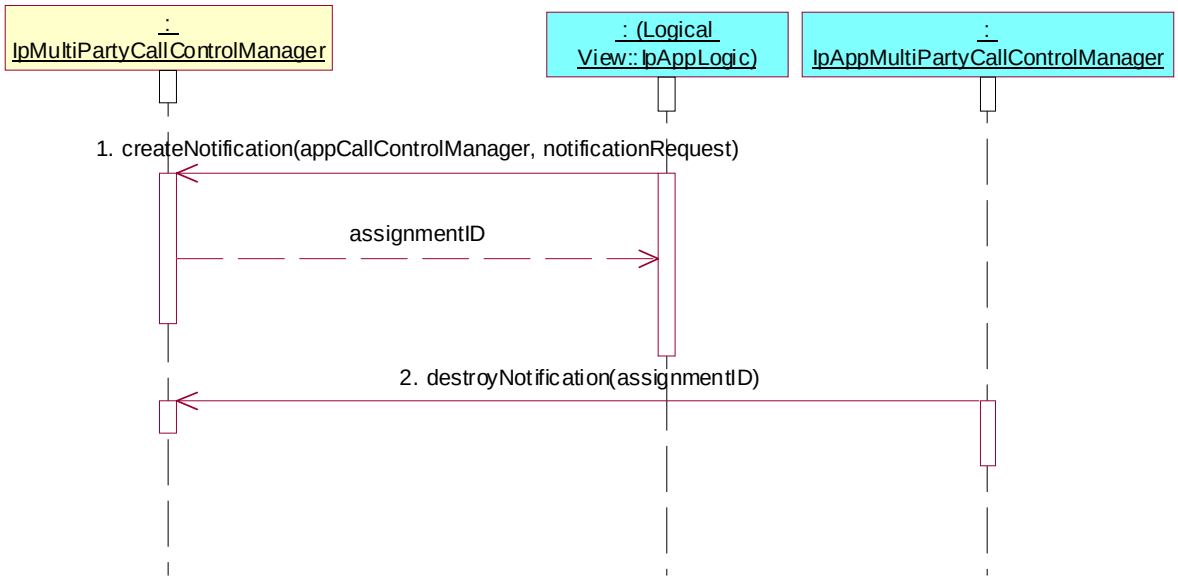
Preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiPartyCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiPartyCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest

2. Triggered Action: cause IUT to call **destroyNotification()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: assignmentID



Test MPCC_IpAppMultiPartyCallControlManager_07

Summary: enable call notifications and get criteria

Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.2

Precondition: IUT capable of invoking **createNotification()** and **getNextNotification()**

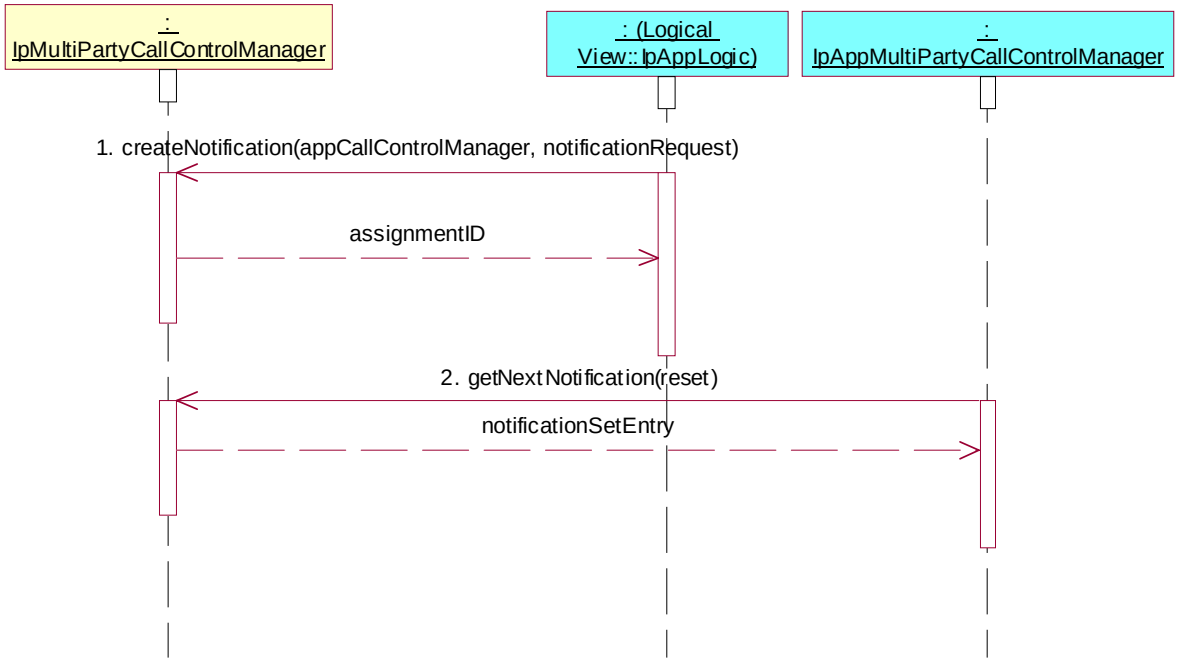
Preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiPartyCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiPartyCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest

2. Triggered Action: cause IUT to call **getNextNotification()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: reset



Test MPCC_IpAppMultiPartyCallControlManager_08

Summary: enable load control and accept overload notifications

Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.2

Precondition: IUT capable of invoking **setCallLoadControl()**, **callOverloadEncountered()** and **callOverloadCeased()** implemented

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

The application is permitted to provide its IpAppMultiPartyCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest

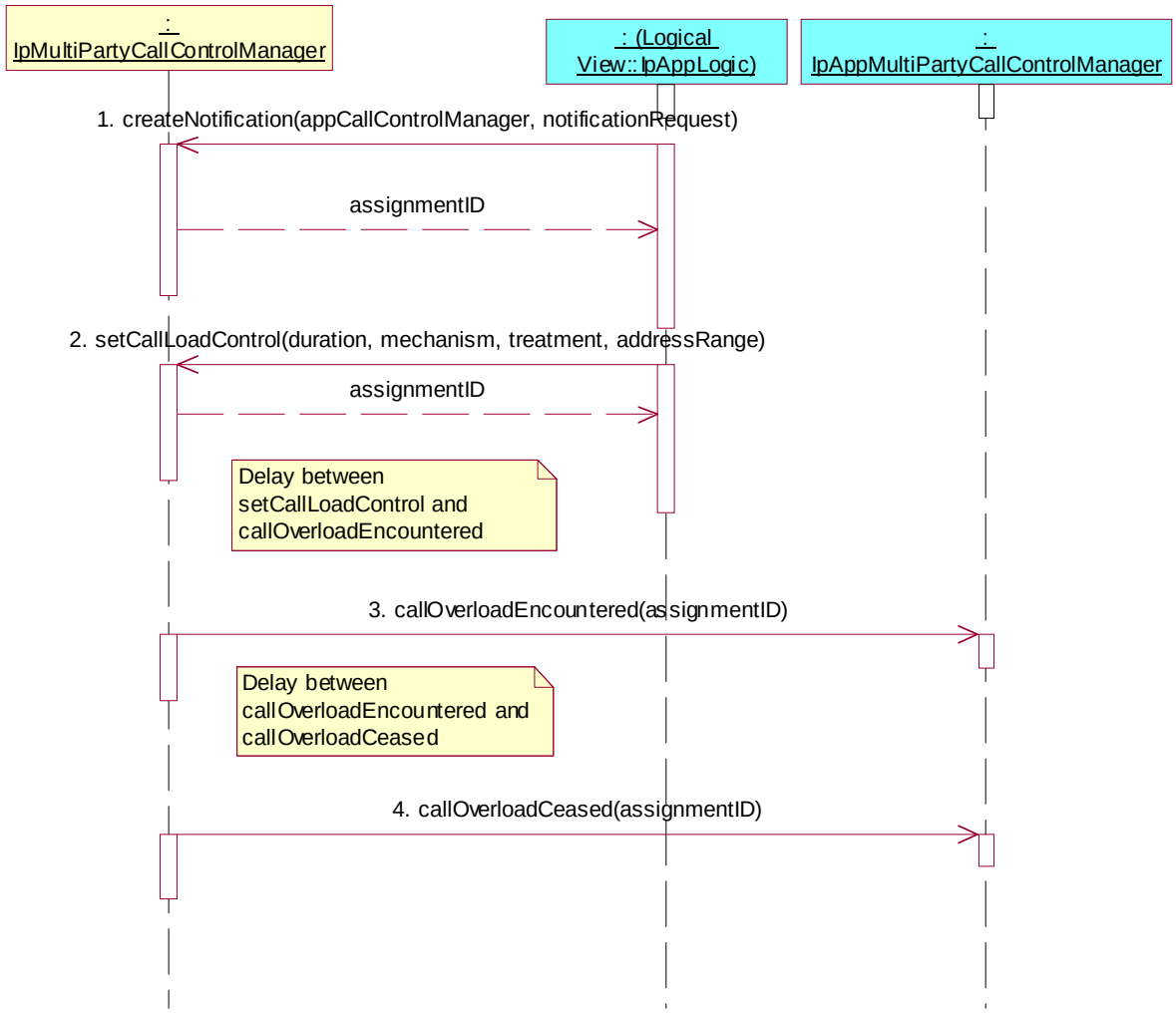
2. Triggered Action: cause IUT to call **setCallLoadControl()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: duration, mechanism, treatment, addressRange

Delay between setCallLoadControl and callOverloadEncountered

3. Method call **callOverloadEncountered()**
   Parameters: assignmentID
   Check: no exception is returned

Delay between callOverloadEncountered and callOverloadCeased

4. Method call **callOverloadCeased()**
   Parameters: assignmentID
   Check: no exception is returned



Test MPCC_IpAppMultiPartyCallControlManager_09

Summary: enable and accept call notifications

Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.2

Precondition: IUT capable of invoking **enableNotifications()**, **reportNotification()** implemented

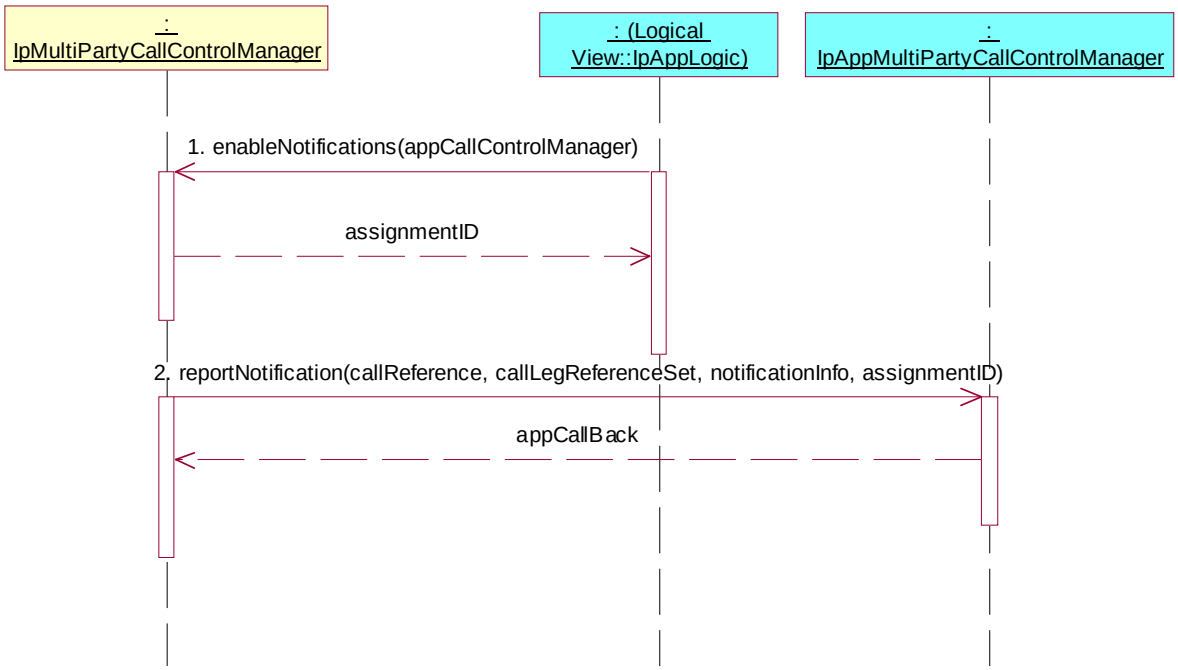
Preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiPartyCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiPartyCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **enableNotifications()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCallControlManager

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   Check: valid value of TpAppMultiPartyCallBack is returned



Test MPCC_IpAppMultiPartyCallControlManager_10

Summary: enable and disable call notifications

Reference: ES 202 915-4-3 [3], clause 6.1

Precondition: IUT capable of invoking **enableNotifications()** and **disableNotifications()**

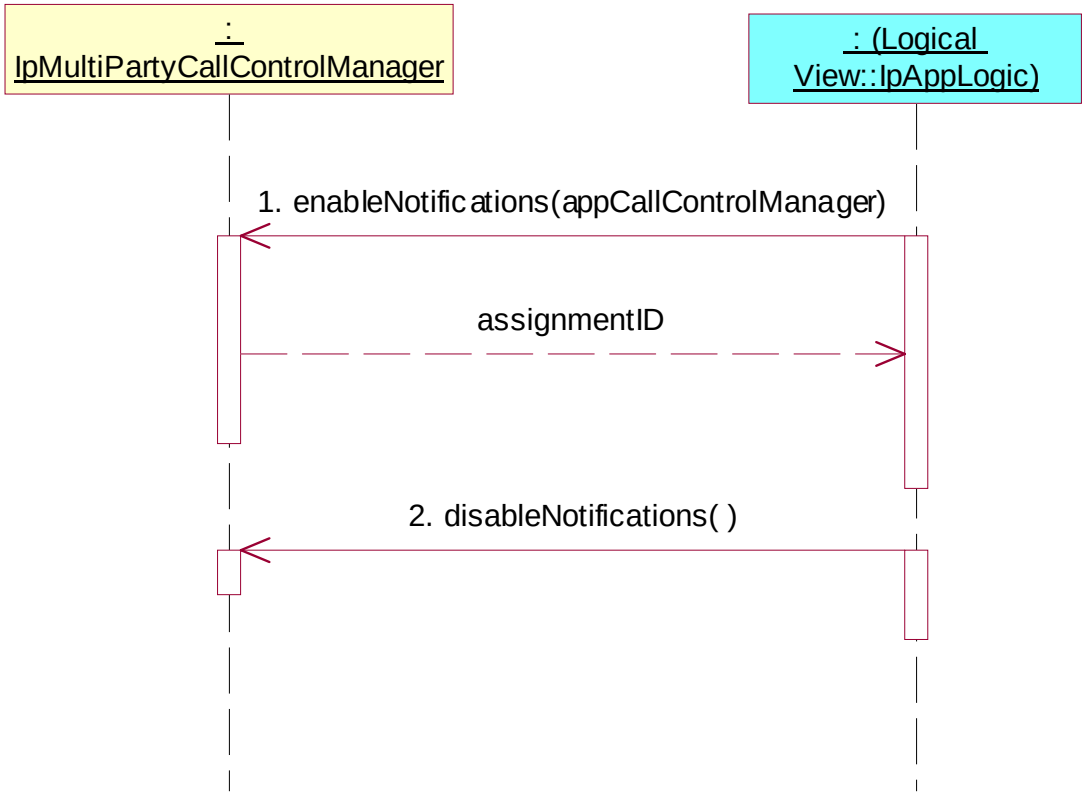
Preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiPartyCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiPartyCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **enableNotifications()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCallControlManager

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



7.2.2.2 IpAppMultiPartyCall

Applications need not be capable of performing each of the sequences below, even if they support the methods indicated below.

Reference: ES 202 915-4-3 [3], clause 7.2

7.2.2.2.1 Idle state

Precondition: IUT capable of invoking **createCall()**

Preamble MPCC_IpAppMultiPartyCall_Idle

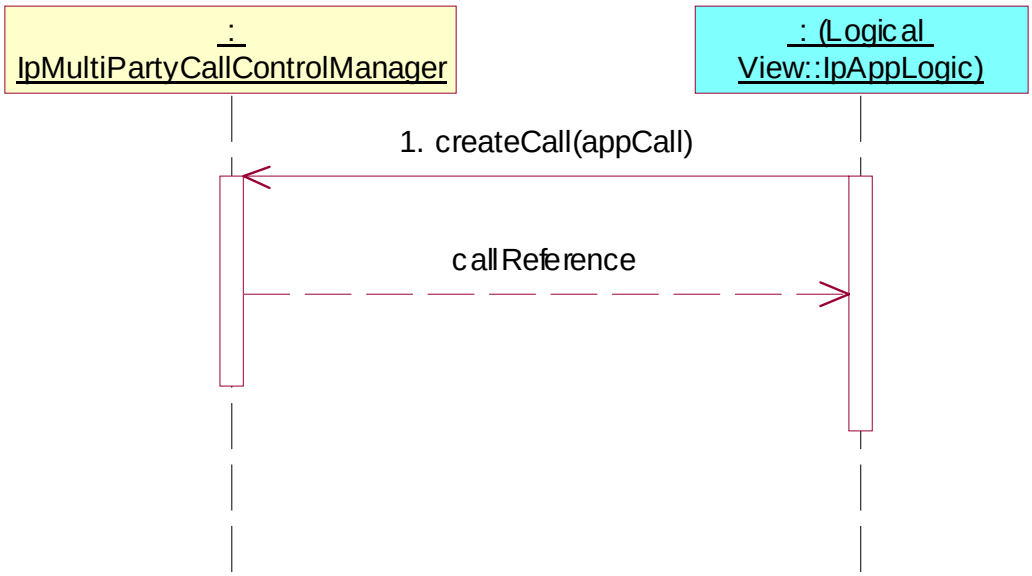
Reference: ES 202 915-4-3 [3], clause 7.2.1

Pre-preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiPartyCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiPartyCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Preamble Sequence:

1. Triggered Action: cause IUT to call **createCall()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCall



Test MPCC_IpAppMultiPartyCall_01

Summary: create call leg

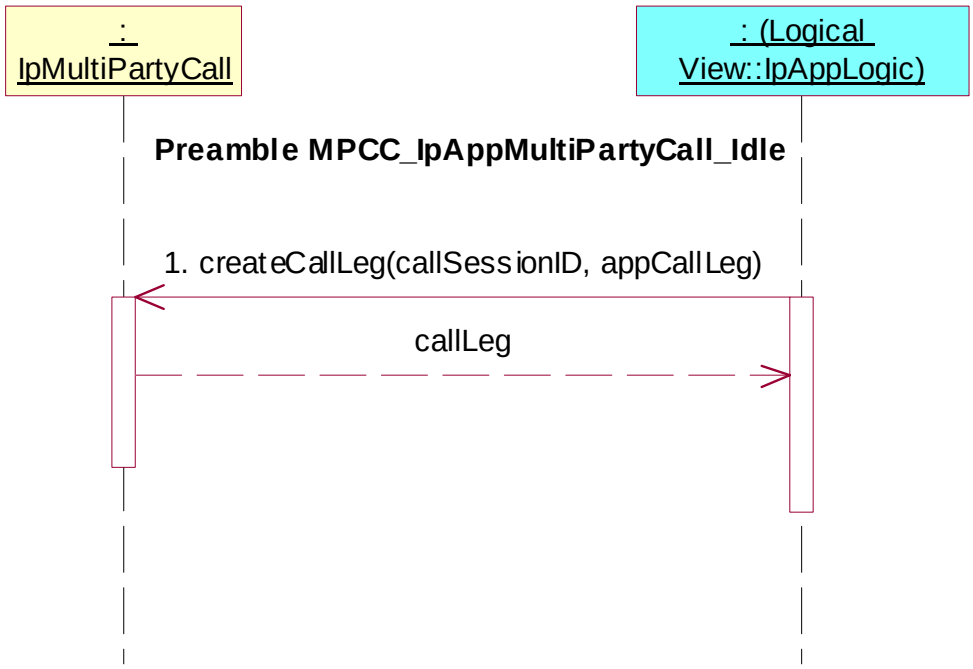
Reference: ES 202 915-4-3 [3], clause 7.2.1

Precondition: IUT capable of invoking **createCallLeg()**

Preamble: **MPCC_IpAppMultiPartyCall_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **createCallLeg()** method on the tester's (SCF's) IpMultiPartyCall interface.
   Parameters: callSessionID, appCallLeg



Test MPCC_IpAppMultiPartyCall_02

Summary: create and route call leg, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.2.1

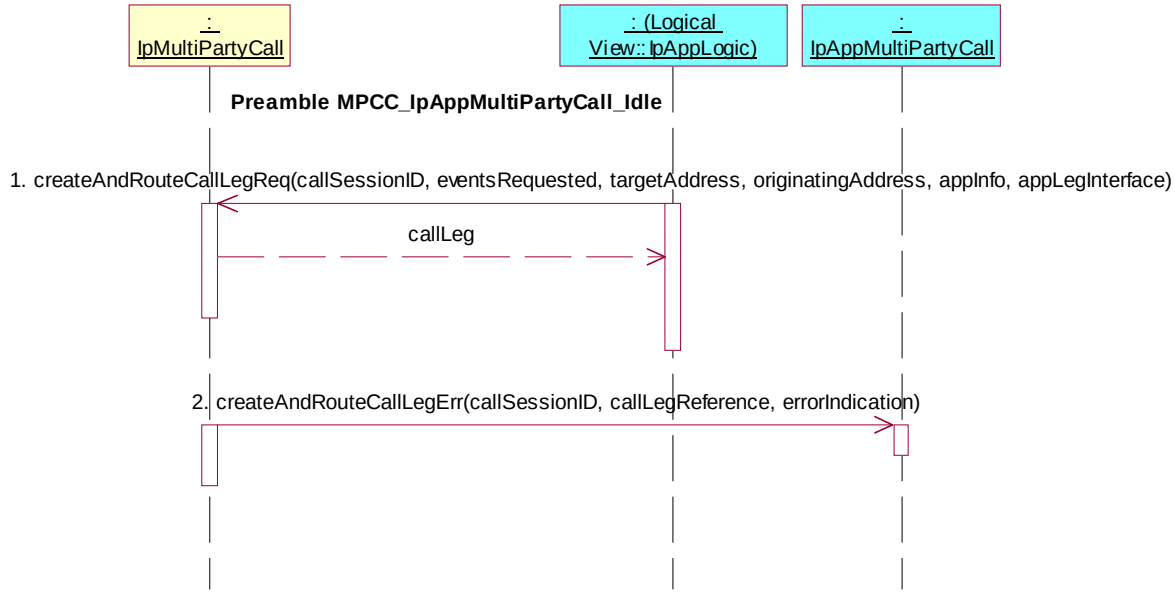
Precondition: IUT capable of invoking **createAndRouteCallLegReq()**

Preamble: **MPCC_IpAppMultiPartyCall_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **createAndRouteCallLegReq()** method on the tester's (SCF's) IpMultiPartyCall interface.
   Parameters: callSessionID, eventsRequested, targetAddress, originatingAddress, appInfo, appLegInterface

2. Method call **createAndRouteCallLegErr()**
   Parameters: callSessionID, appCallLegReference, errorIndication
   Check: no exception is returned



Test MPCC_IpAppMultiPartyCall_03

Summary: supervise call

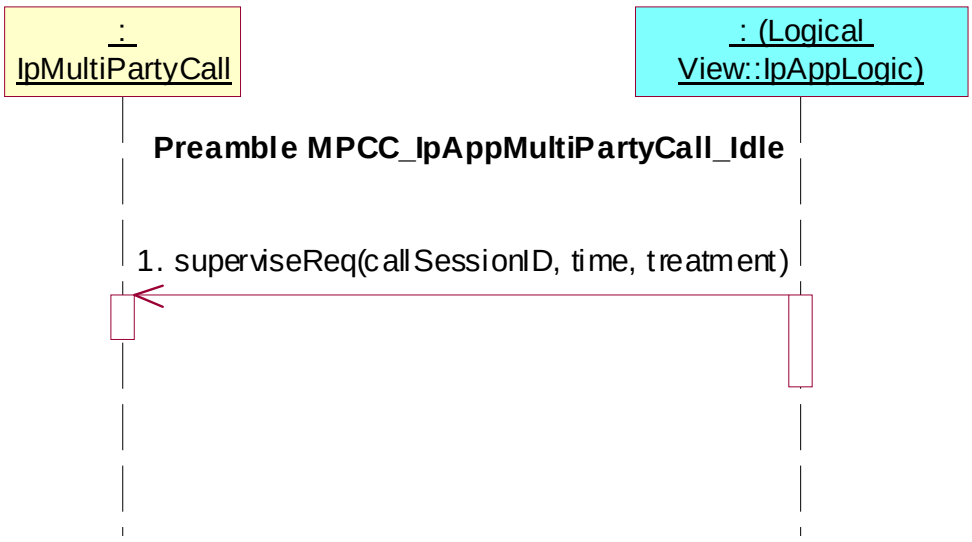
Reference: ES 202 915-4-3 [3], clause 7.2.1

Precondition: IUT capable of invoking **superviseReq()**

Preamble: **MPCC_IpAppMultiPartyCall_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) IpMultiPartyCall interface.
   Parameters: callSessionID, time, treatment



Test MPCC_IpAppMultiPartyCall_04

Summary: request call information

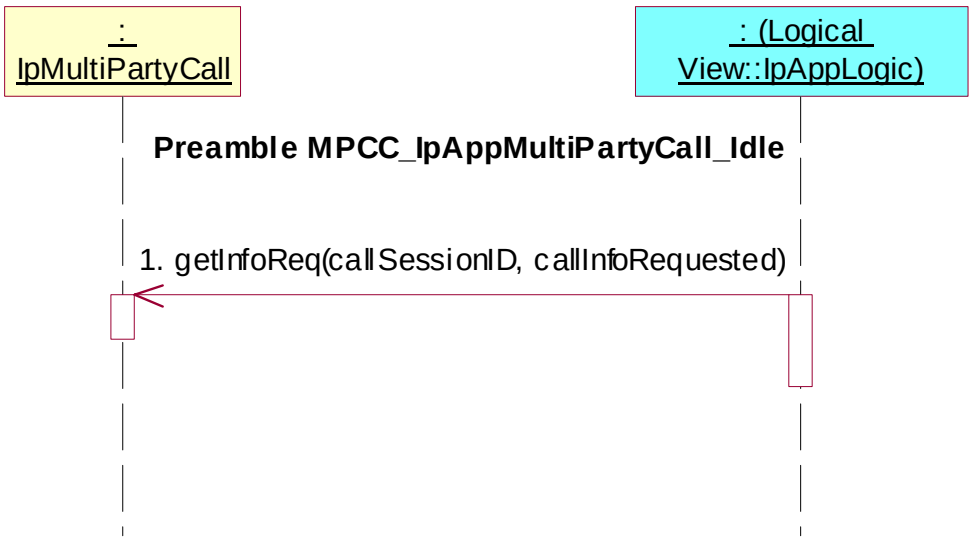
Reference: ES 202 915-4-3 [3], clause 7.2.1

Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **MPCC_IpAppMultiPartyCall_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) IpMultiPartyCall interface.
   Parameters: callSessionID, callInfoRequested



Test MPCC_IpAppMultiPartyCall_05

Summary: request call information, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.2.1

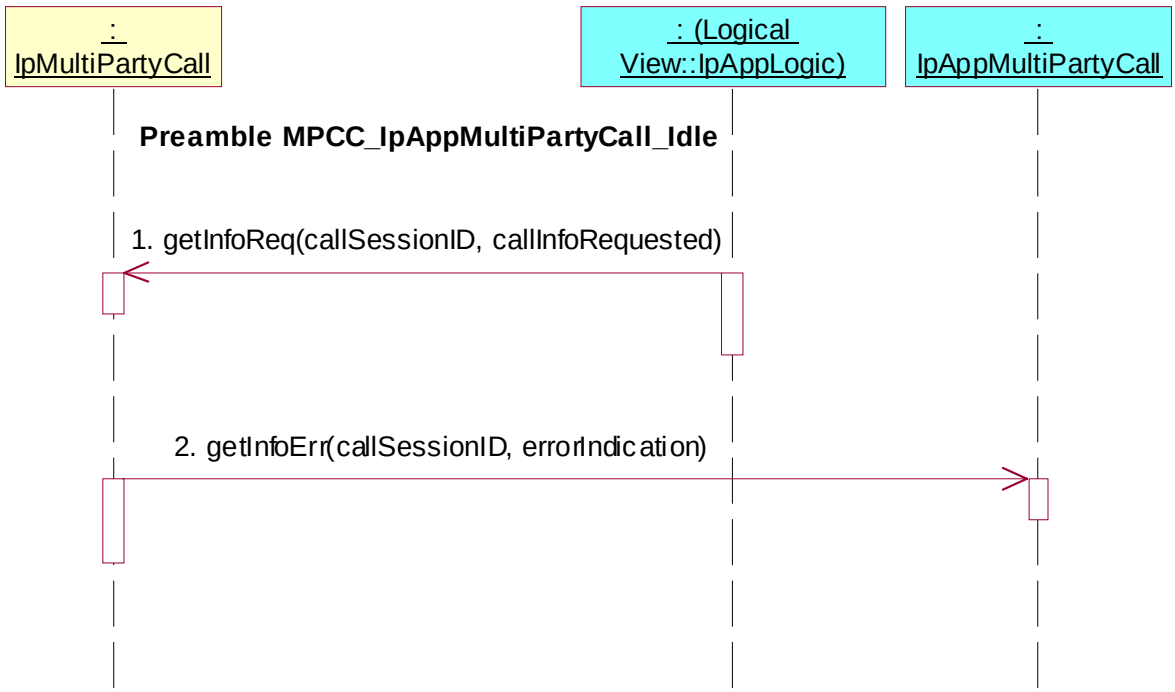
Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **MPCC_IpAppMultiPartyCall_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) IpMultiPartyCall interface.
   Parameters: callSessionID, callInfoRequested

2. Method call **getInfoErr()**
   Parameters: callSessionID, errorIndication
   Check: no exception is returned



Test MPCC_IpAppMultiPartyCall_06

Summary: set charge plan for the call

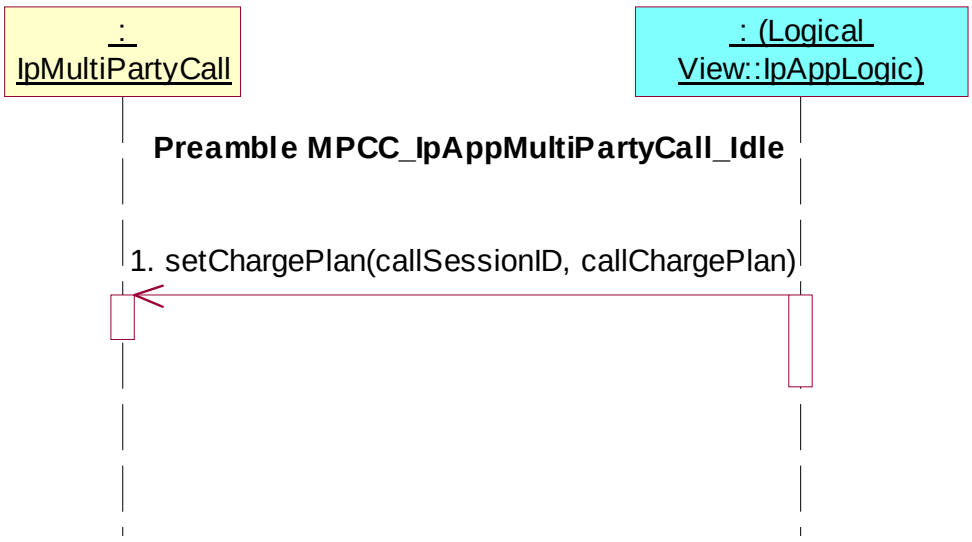
Reference: ES 202 915-4-3 [3], clause 7.2.1

Precondition: IUT capable of invoking **setChargePlan()**

Preamble: **MPCC_IpAppMultiPartyCall_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **setChargePlan()** method on the tester's (SCF's) IpMultiPartyCall interface.
   Parameters: callSessionID, callChargePlan



Test MPCC_IpAppMultiPartyCall_07

Summary: allow advice of charge information

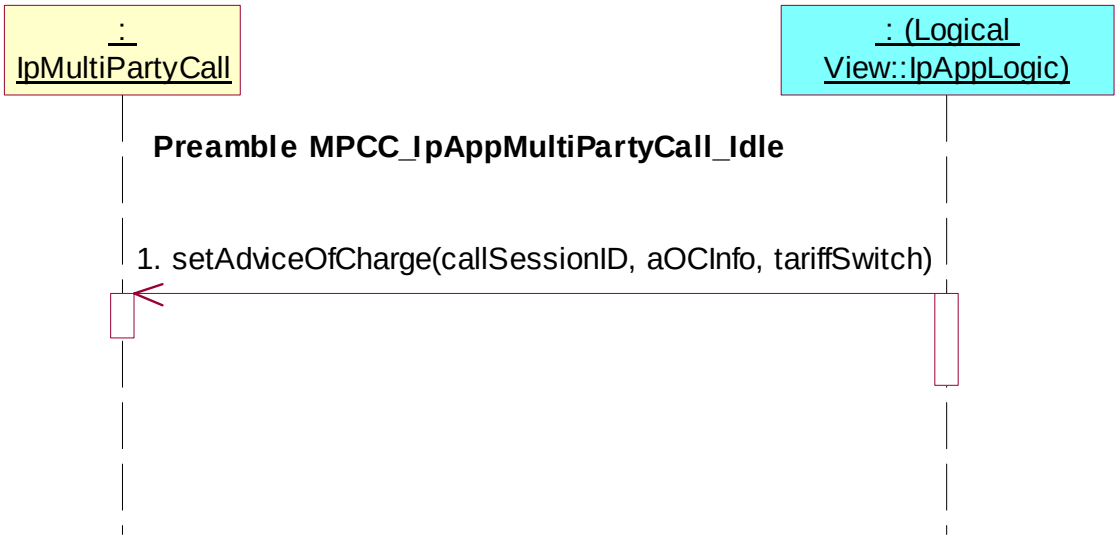
Reference: ES 202 915-4-3 [3], clause 7.2.1

Precondition: IUT capable of invoking **setAdviceOfCharge()**

Preamble: **MPCC_IpAppMultiPartyCall_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **setAdviceOfCharge()** method on the tester's (SCF's) IpMultiPartyCall interface.
   Parameters: callSessionID, aOCInfo, tariffSwitch



7.2.2.2.2 Active state

Precondition: IUT capable of invoking **createCall()** and **createCallLeg()**

or IUT capable of invoking **createCall()** and **createAndRouteCallLegReq()**

or IUT capable of invoking **createNotification()**

or IUT capable of invoking **enableNotifications()**

Preamble MPCC_IpAppMultiPartyCall_Active

Reference: ES 202 915-4-3 [3], clause 7.2.2

Pre-preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiPartyCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiPartyCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Preamble Sequence:

1. Triggered Action: cause IUT to call **createCall()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCall

2. Triggered Action: cause IUT to call **createCallLeg()** method on the tester's (SCF's) IpMultiPartyCall interface.
   Parameters: callSessionID, appCallLeg

or

1. Triggered Action: cause IUT to call **createCall()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCall

2. Triggered Action: cause IUT to call **createAndRouteCallLegReq()** method on the tester's (SCF's) IpMultiPartyCall interface.
   Parameters: callSessionID, eventsRequested, targetAddress, originatingAddress, appInfo, appLegInterface

or

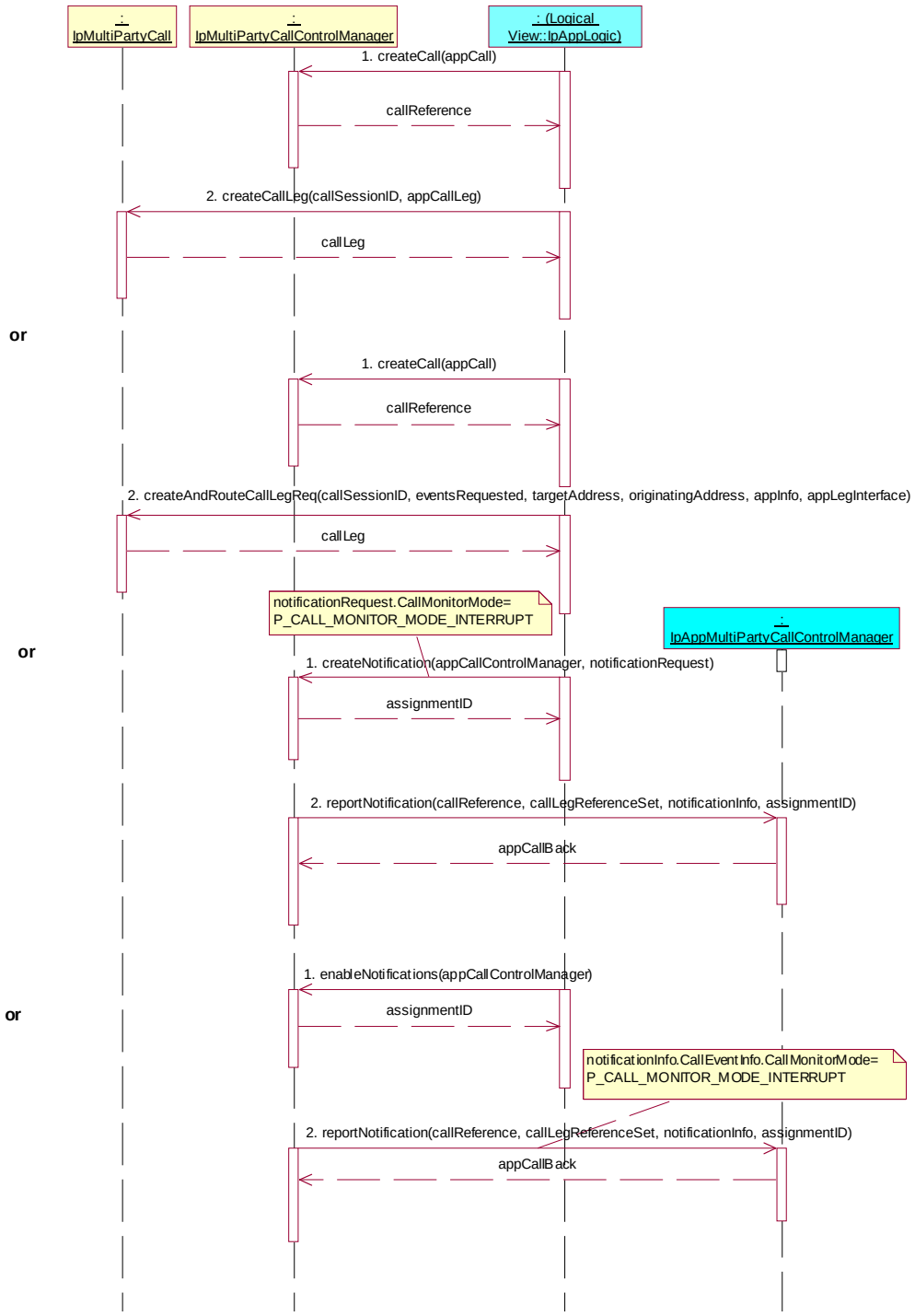
1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest
   notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   Check: valid value of TpAppMultiPartyCallBack is returned

or

1. Triggered Action: cause IUT to call **enableNotifications()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCallControlManager

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   notificationInfo.CallEventInfo.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   Check: valid value of TpAppMultiPartyCallBack is returned


: IpMultiPartyCall
: IpMultiPartyCallControlManager
: (Logical View::IpAppLogic)
1. createCall(appCall)
callReference
2. createCallLeg(callSessionID, appCallLeg)
callLeg
or
1. createCall(appCall)
callReference
2. createAndRouteCallLegReq(callSessionID, eventsRequested, targetAddress, originatingAddress, appInfo, appLegInterface)
callLeg
notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
: IpAppMultiPartyCallControlManager
or
1. createNotification(appCallControlManager, notificationRequest)
assignmentID
2. reportNotification(callReference, callLegReferenceSet, notificationInfo, assignmentID)
appCallBack
1. enableNotifications(appCallControlManager)
or
assignmentID
notificationInfo.CallEventInfo.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
2. reportNotification(callReference, callLegReferenceSet, notificationInfo, assignmentID)
appCallBack

Test MPCC_IpAppMultiPartyCall_08

Summary: create call leg

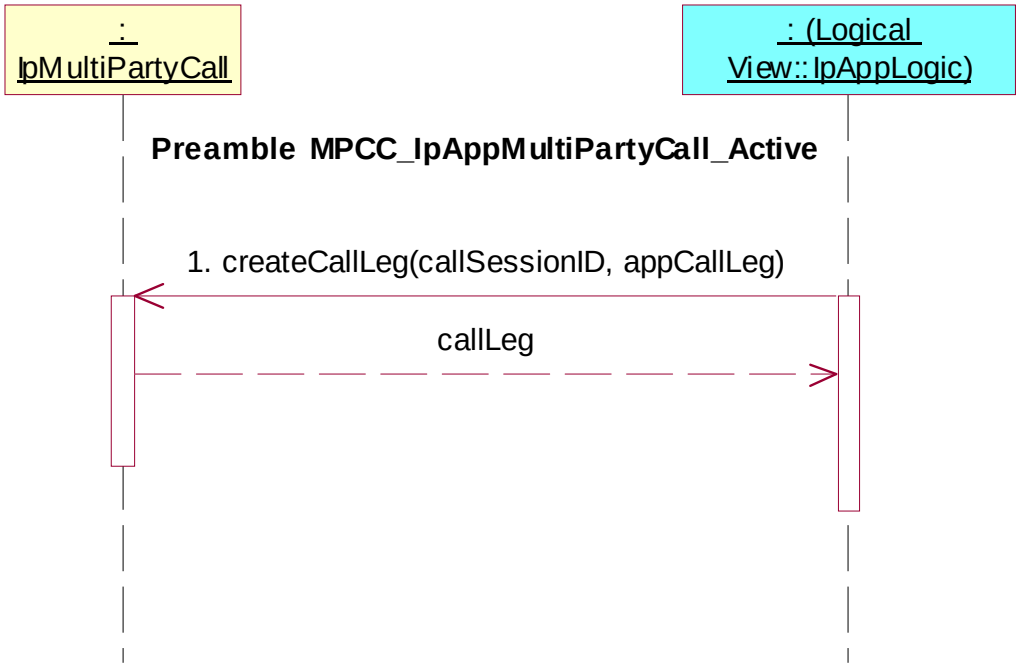
Reference: ES 202 915-4-3 [3], clause 7.2.2

Precondition: IUT capable of invoking **createCallLeg()**

Preamble: **MPCC_IpAppMultiPartyCall_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **createCallLeg()** method on the tester's (SCF's) IpMultiPartyCall interface.
   Parameters: callSessionID, appCallLeg



Test MPCC_IpAppMultiPartyCall_09

Summary: create and route call leg, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.2.2

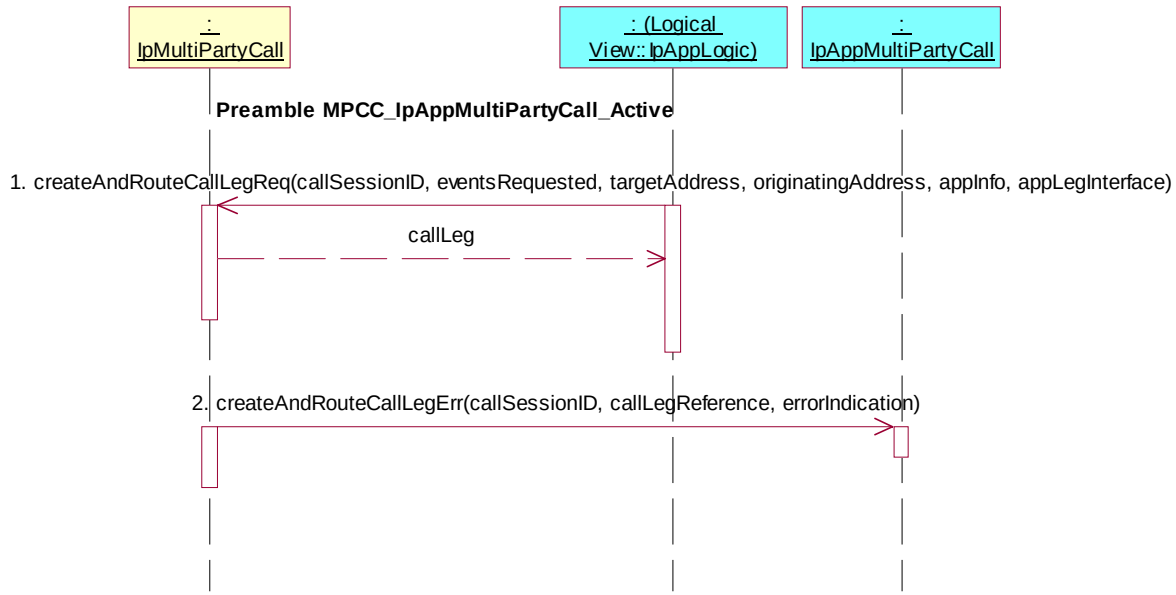
Precondition: IUT capable of invoking **createAndRouteCallLegReq()**

Preamble: **MPCC_IpAppMultiPartyCall_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **createAndRouteCallLegReq()** method on the tester's (SCF's) IpMultiPartyCall interface.
   Parameters: callSessionID, eventsRequested, targetAddress, originatingAddress, appInfo, appLegInterface

2. Method call **createAndRouteCallLegErr()**
   Parameters: callSessionID, appCallLegReference, errorIndication
   Check: no exception is returned



Test MPCC_IpAppMultiPartyCall_10

Summary: supervise call, successful

Reference: ES 202 915-4-3 [3], clause 7.2.2

Precondition: IUT capable of invoking **superviseReq()**

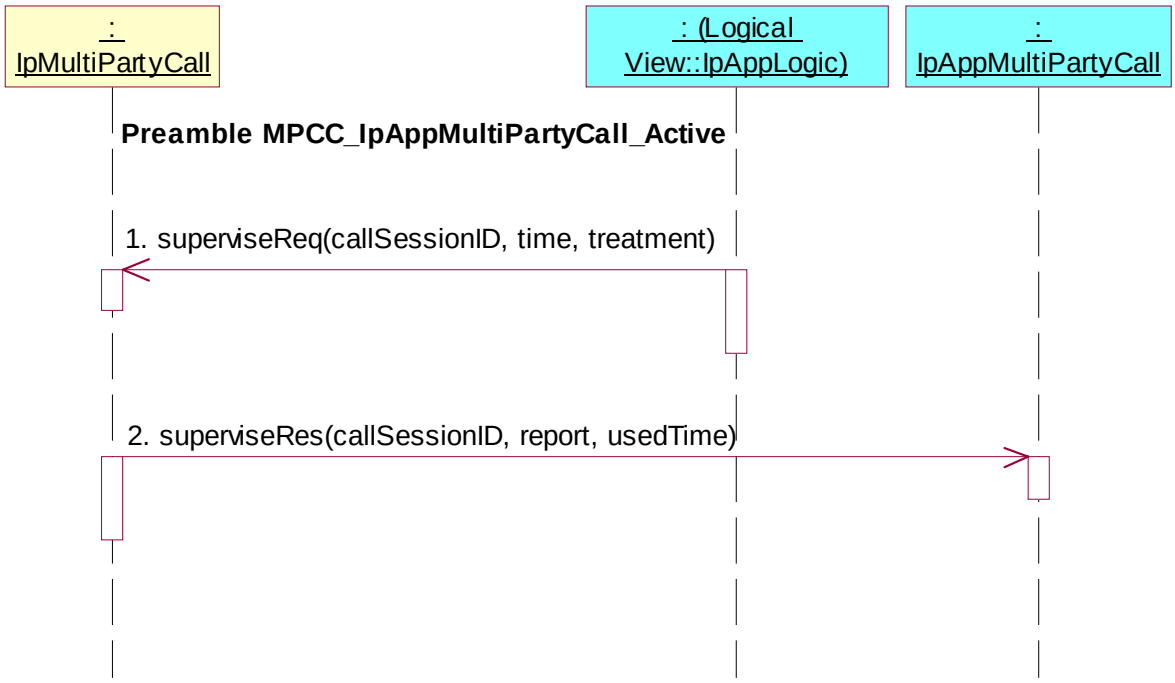
Preamble: **MPCC_IpAppMultiPartyCall_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) IpMultiPartyCall interface.
   Parameters: callSessionID, time, treatment

NOTE: Between these two method sequences, the IUT may need to be triggered to complete establishment of the call to both parties, in order to justify the Tester's calling of a superviseRes() method.

2. Method call **superviseRes()**
   Parameters: callSessionID, report, usedTime
   Check: no exception is returned



Test MPCC_IpAppMultiPartyCall_11

Summary: supervise call, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.2.2

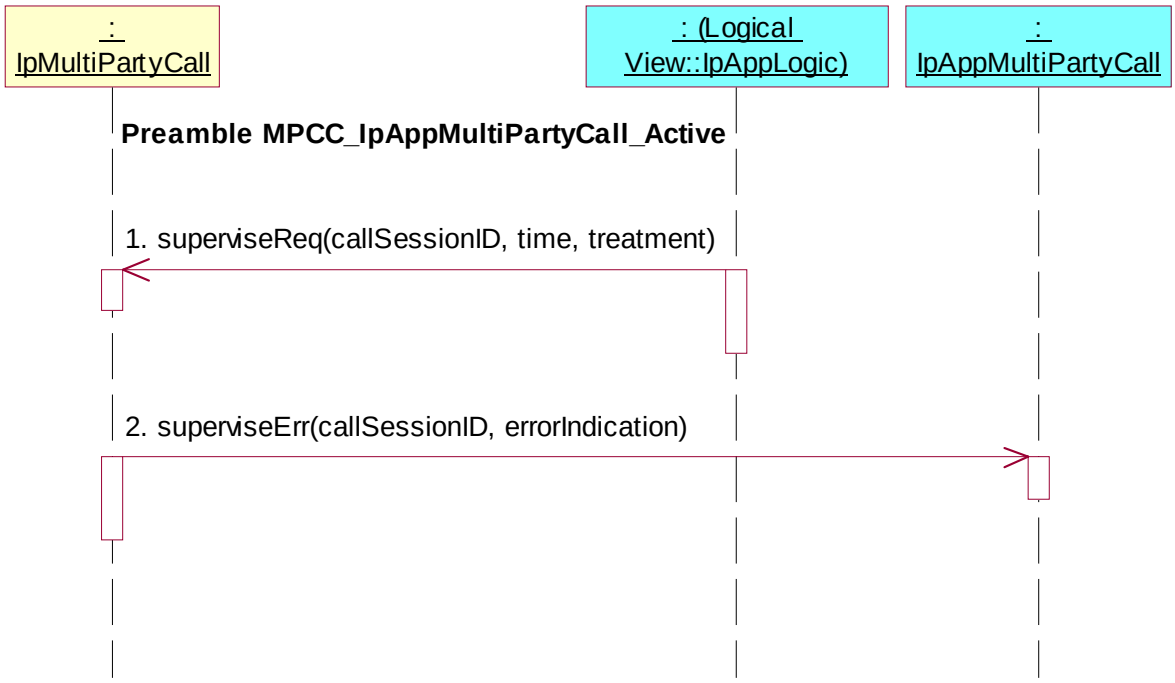
Precondition: IUT capable of invoking **superviseReq()**

Preamble: **MPCC_IpAppMultiPartyCall_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) IpMultiPartyCall interface.
   Parameters: callSessionID, time, treatment

2. Method call **superviseErr()**
   Parameters: callSessionID, errorIndication
   Check: no exception is returned



Test MPCC_IpAppMultiPartyCall_12

Summary: request call leg information

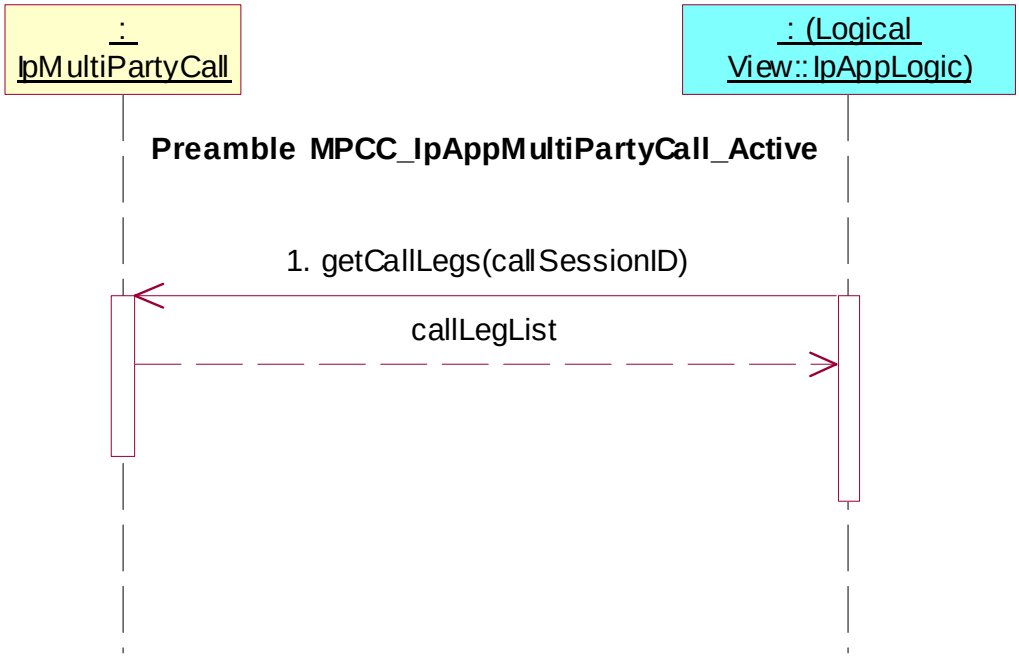
Reference: ES 202 915-4-3 [3], clause 7.2.2

Precondition: IUT capable of invoking **getCallLegs()**

Preamble: **MPCC_IpAppMultiPartyCall_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **getCallLegs()** method on the tester's (SCF's) IpMultiPartyCall interface.
   Parameters: callSessionID



Test MPCC_IpAppMultiPartyCall_13

Summary: release call

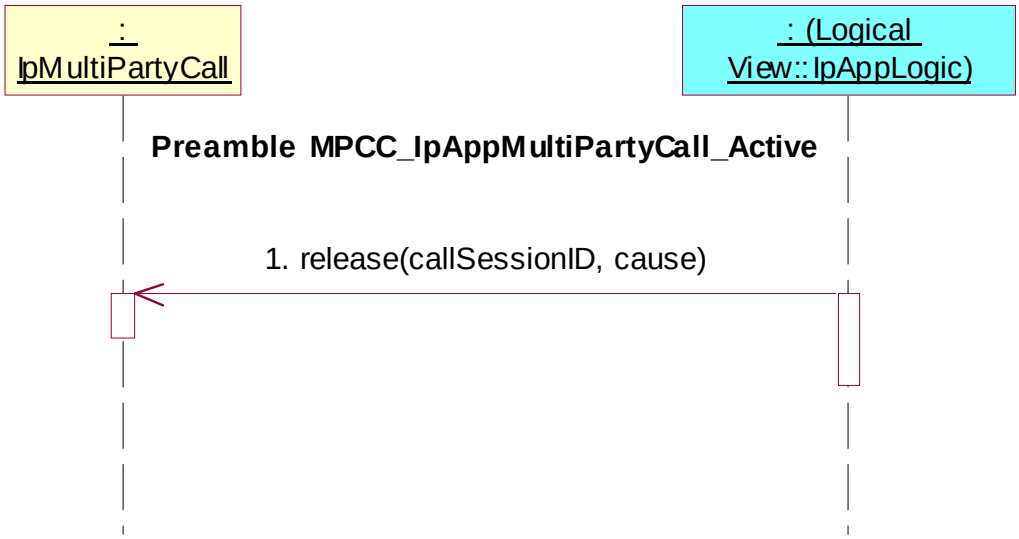
Reference: ES 202 915-4-3 [3], clause 7.2.2

Precondition: IUT capable of invoking **release()**

Preamble: **MPCC_IpAppMultiPartyCall_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) IpMultiPartyCall interface.
   Parameters: callSessionID, cause



Test MPCC_IpAppMultiPartyCall_14

Summary: deassign call

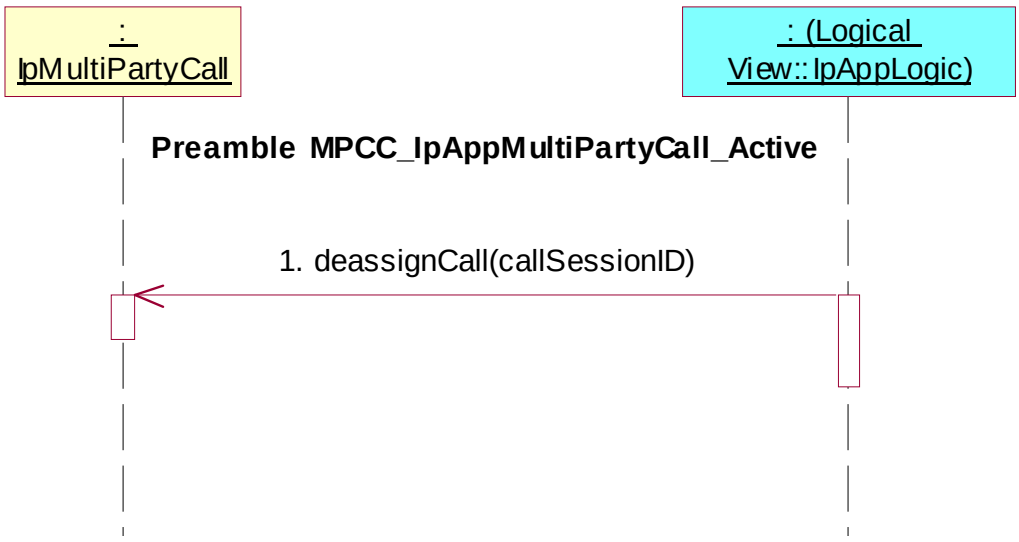
Reference: ES 202 915-4-3 [3], clause 7.2.2

Precondition: IUT capable of invoking **deassignCall()**

Preamble: **MPCC_IpAppMultiPartyCall_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **deassignCall()** method on the tester's (SCF's) IpMultiPartyCall interface.
   Parameters: callSessionID



Test MPCC_IpAppMultiPartyCall_15

Summary: indication of termination of call

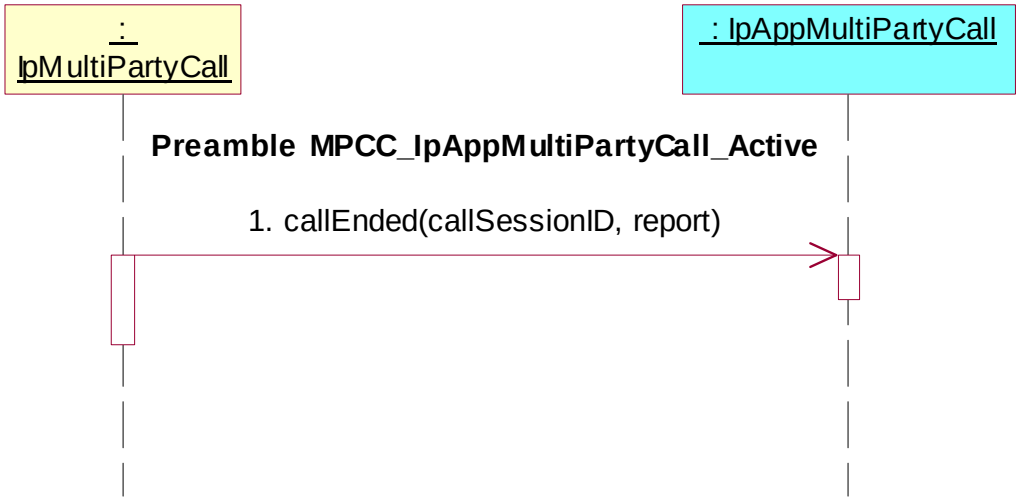
Reference: ES 202 915-4-3 [3], clause 7.2.2

Precondition: **callEnded()** implemented

Preamble: **MPCC_IpAppMultiPartyCall_Active**

Test Sequence:

1. Method call **callEnded()**
   Parameters: callSessionID, report
   Check: no exception is returned



7.2.2.2.3 Released state

Precondition: IUT capable of invoking **createCall()** and **createCallLeg()**

or IUT capable of invoking **createCall()** and **createAndRouteCallLegReq()**

or IUT capable of invoking **createNotification()**

and IUT capable of invoking **release()**

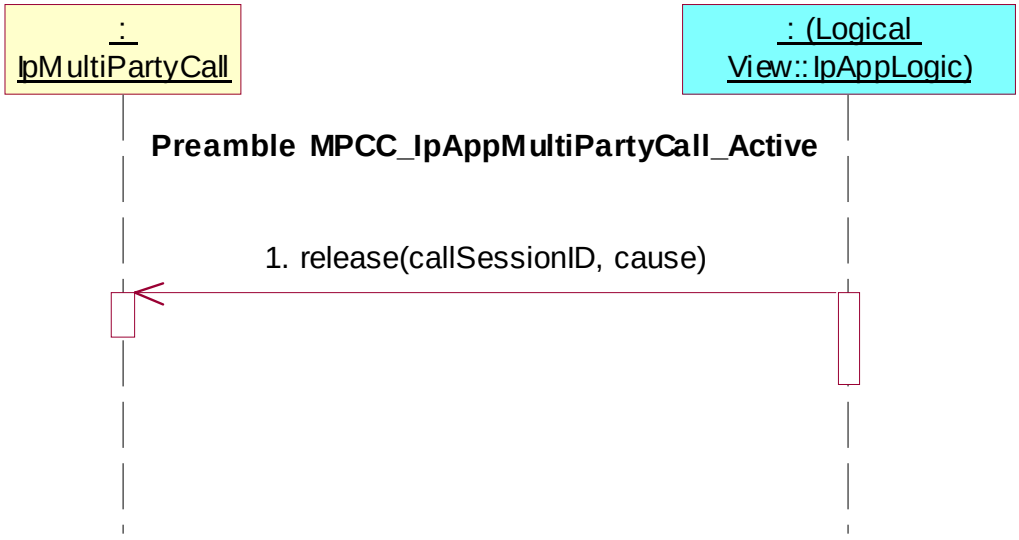
Preamble MPCC_IpAppMultiPartyCall_Released

Reference: ES 202 915-4-3 [3], clause 7.2.3

Pre-preamble: **MPCC_IpAppMultiPartyCall_Active**

Preamble Sequence:

1. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) IpMultiPartyCall interface.
   Parameters: callSessionID, cause



Test MPCC_IpAppMultiPartyCall_16

Summary: request call leg information

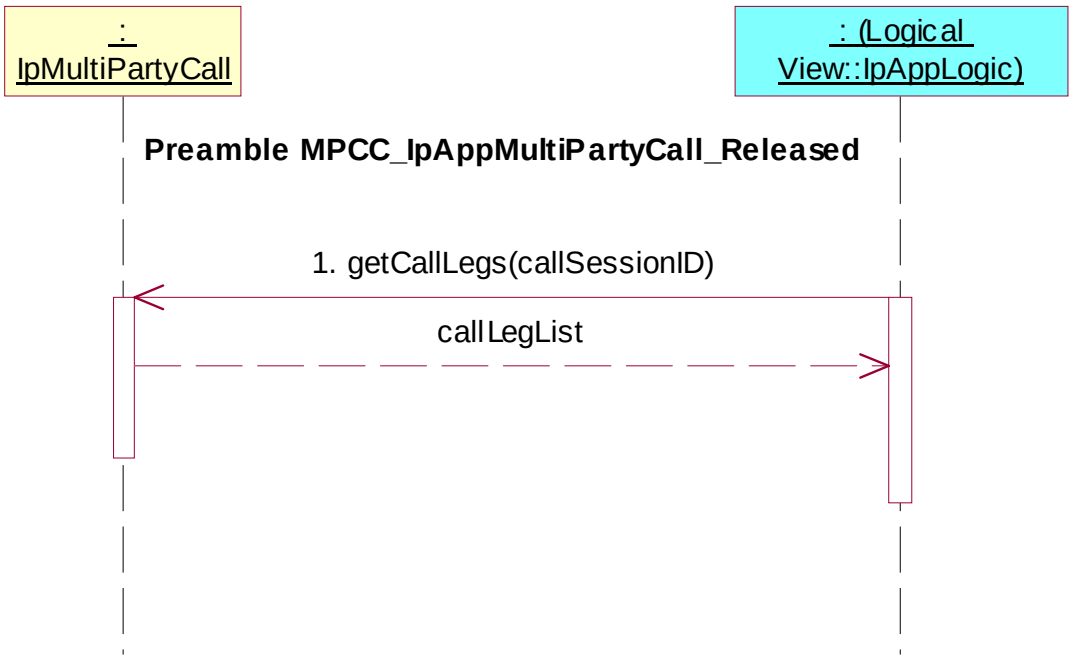
Reference: ES 202 915-4-3 [3], clause 7.2.3

Precondition: IUT capable of invoking **getCallLegs()**

Preamble: **MPCC_IpAppMultiPartyCall_Released**

Test Sequence:

1. Triggered Action: cause IUT to call **getCallLegs()** method on the tester's (SCF's) IpMultiPartyCall interface.
   Parameters: callSessionID



Test MPCC_IpAppMultiPartyCall_17

Summary: indication of termination of call

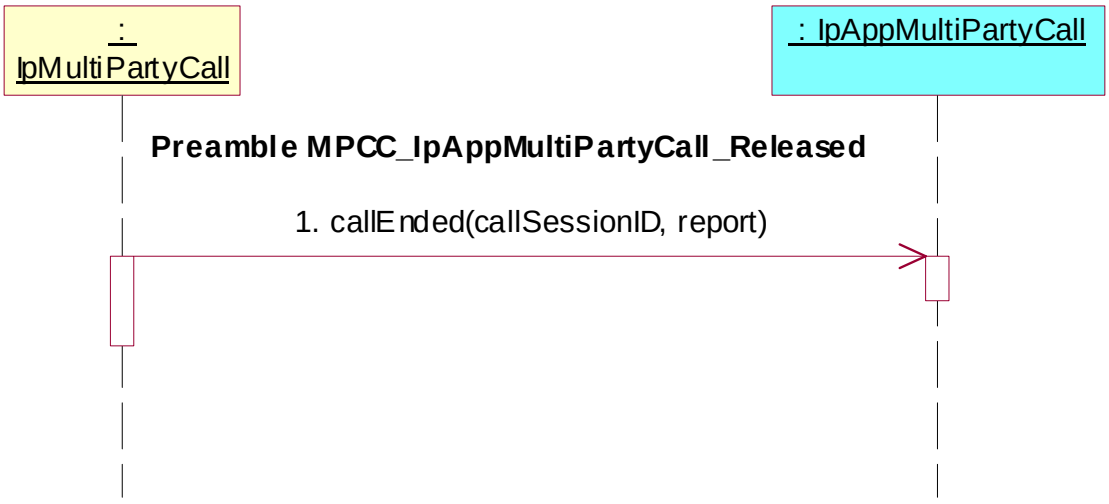
Reference: ES 202 915-4-3 [3], clause 7.2.3

Precondition: **callEnded()** implemented

Preamble: **MPCC_IpAppMultiPartyCall_Released**

Test Sequence:

1. Method call **callEnded()**
   Parameters: callSessionID, report
   Check: no exception is returned



7.2.2.3 IpAppCallLeg

Applications need not be capable of performing each of the sequences below, even if they support the methods indicated below.

Reference: ES 202 915-4-3 [3], clause 7.3

7.2.2.3.1 Originating Leg

Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **createNotification()** or **enableNotifications()**

7.2.2.3.1.1 Initiating state

Preamble MPCC_IpAppCallLeg_Initiating

Reference: ES 202 915-4-3 [3], clause 7.3.1.1

Pre-preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiPartyCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiPartyCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Preamble Sequence:

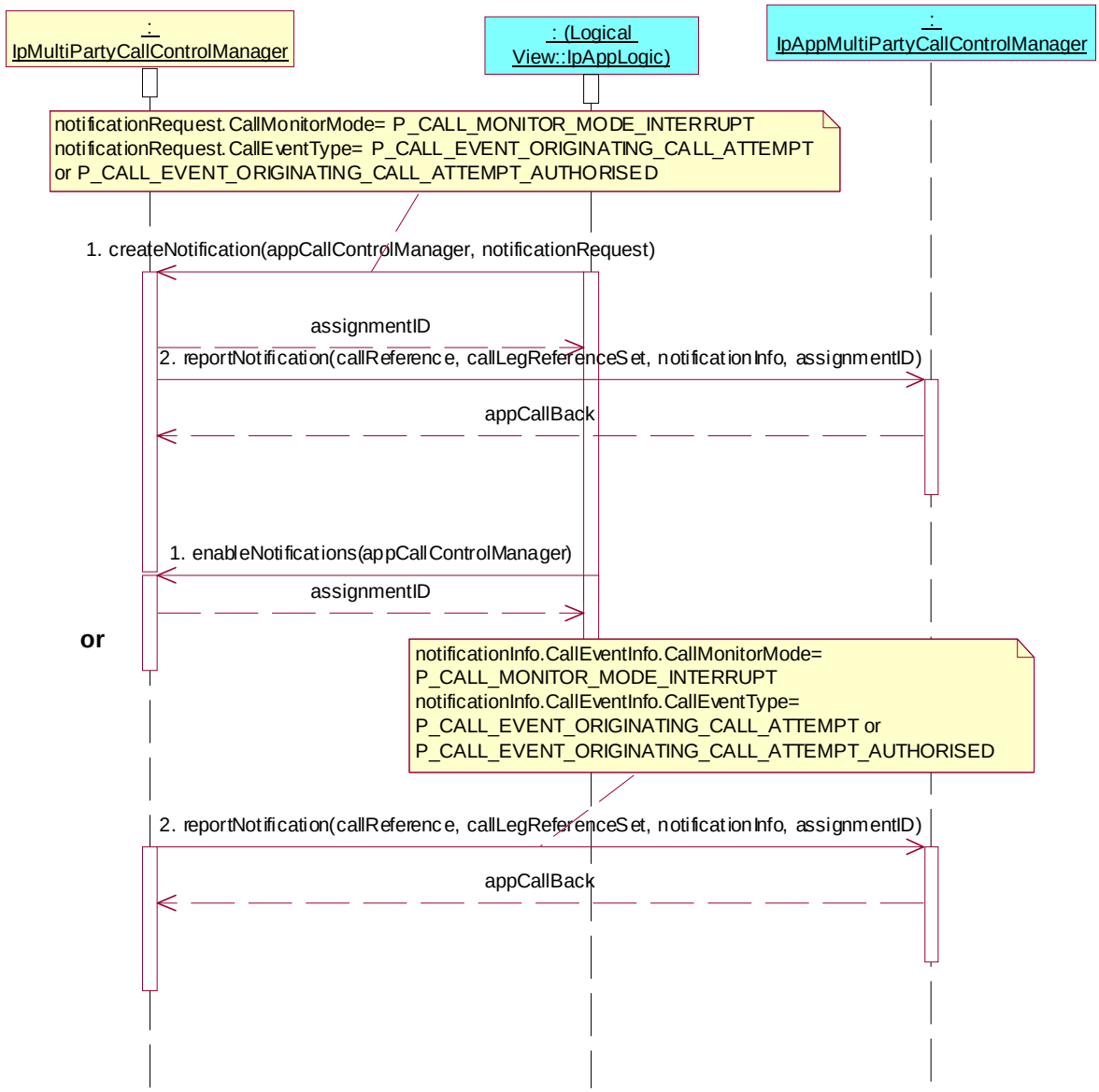
1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest
   notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   notificationRequest.CallEventType= P_CALL_EVENT_ORIGINATING_CALL_ATTEMPT or P_CALL_EVENT_ORIGINATING_CALL_ATTEMPT_AUTHORISED

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   Check: valid value of TpAppMultiPartyCallBack is returned

or

1. Triggered Action: cause IUT to call **enableNotifications()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCallControlManager

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   notificationInfo.CallEventInfo.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   notificationInfo.CallEventInfo.CallEventType=
   P_CALL_EVENT_ORIGINATING_CALL_ATTEMPT or
   P_CALL_EVENT_ORIGINATING_CALL_ATTEMPT_AUTHORISED
   Check: valid value of TpAppMultiPartyCallBack is returned



Test MPCC_IpAppCallLeg_01

Summary: request reference of call related to call leg

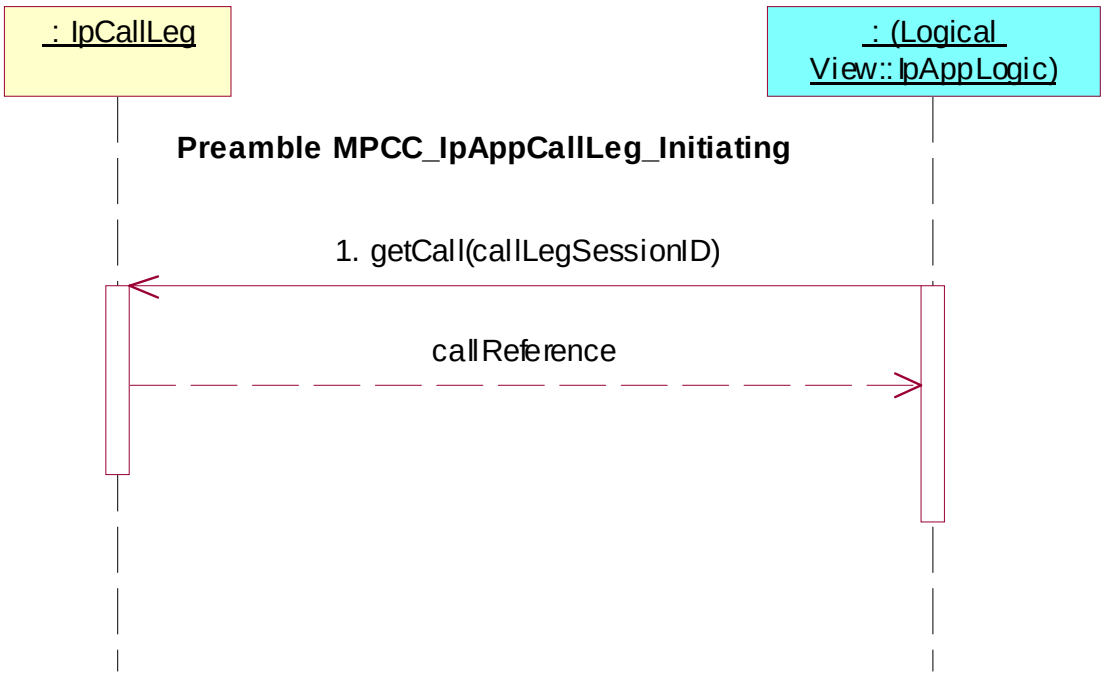
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **getCall()**

Preamble: **MPCC_IpAppCallLeg_Initiating**

Test Sequence:

1. Triggered Action: cause IUT to call **getCall()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID



Test MPCC_IpAppCallLeg_02

Summary: continue processing of call leg

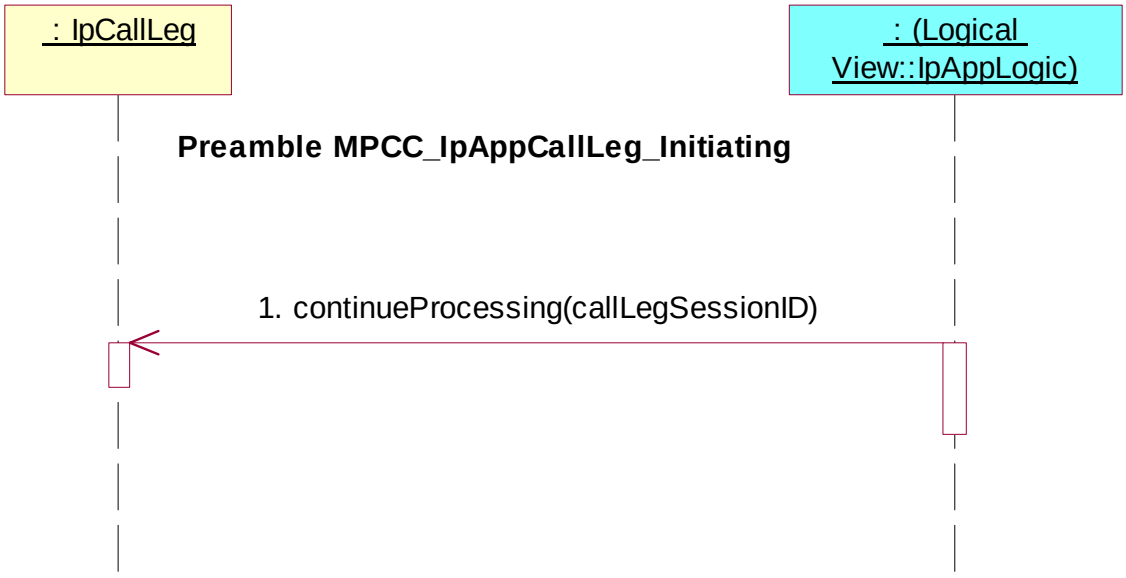
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **continueProcessing()**

Preamble: **MPCC_IpAppCallLeg_Initiating**

Test Sequence:

1. Triggered Action: cause IUT to call **continueProcessing()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID



Test MPCC_IpAppCallLeg_03

Summary: release call leg

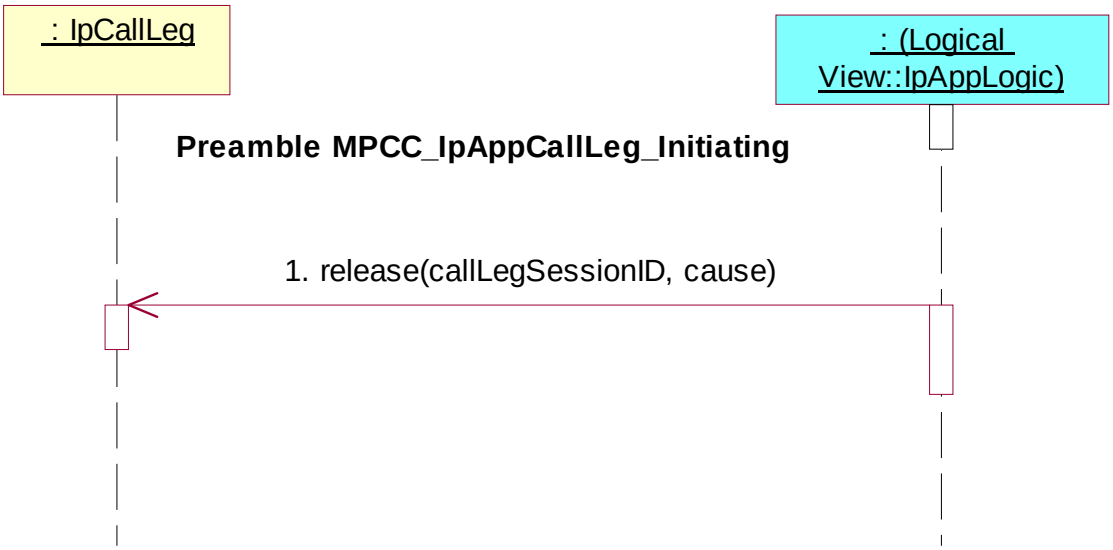
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **release()**

Preamble: **MPCC_IpAppCallLeg_Initiating**

Test Sequence:

1. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID, cause



Test MPCC_IpAppCallLeg_04

Summary: de-assign call leg

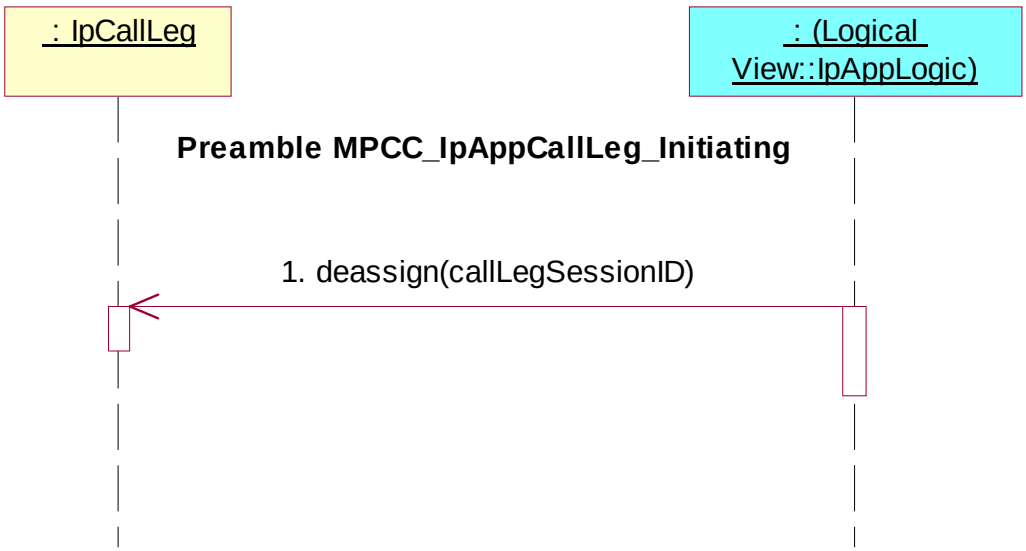
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **deassign()**

Preamble: **MPCC_IpAppCallLeg_Initiating**

Test Sequence:

1. Triggered Action: cause IUT to call **deassign()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID



Test MPCC_IpAppCallLeg_05

Summary: change or clear event criteria

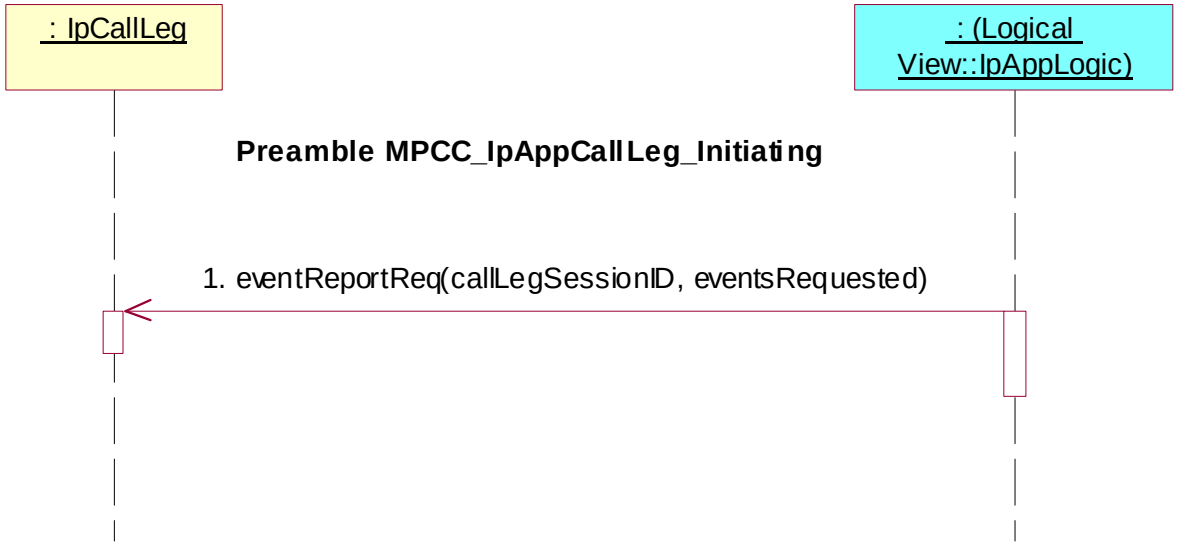
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **eventReportReq()**

Preamble: **MPCC_IpAppCallLeg_Initiating**

Test Sequence:

1. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID, eventsRequested



Test MPCC_IpAppCallLeg_06

Summary: change or clear event criteria, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.1

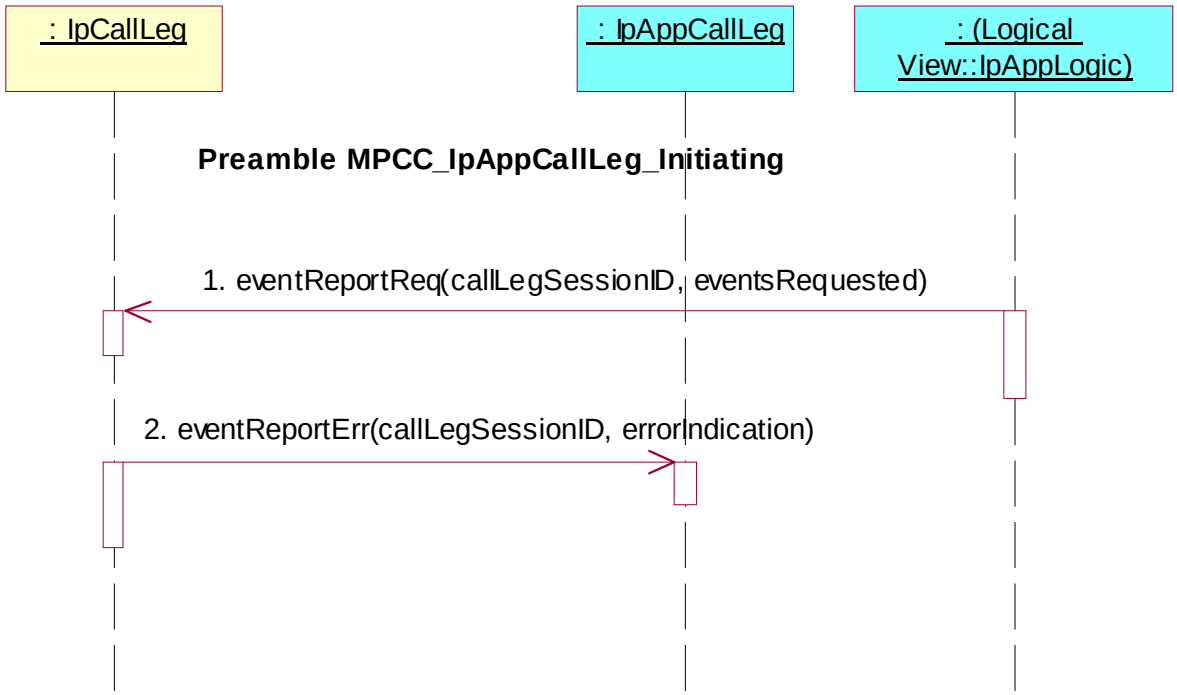
Precondition: IUT capable of invoking **eventReportReq()**

Preamble: **MPCC_IpAppCallLeg_Initiating**

Test Sequence:

1. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID, eventsRequested

2. Method call **eventReportErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MPCC_IpAppCallLeg_07

Summary: get information about call leg

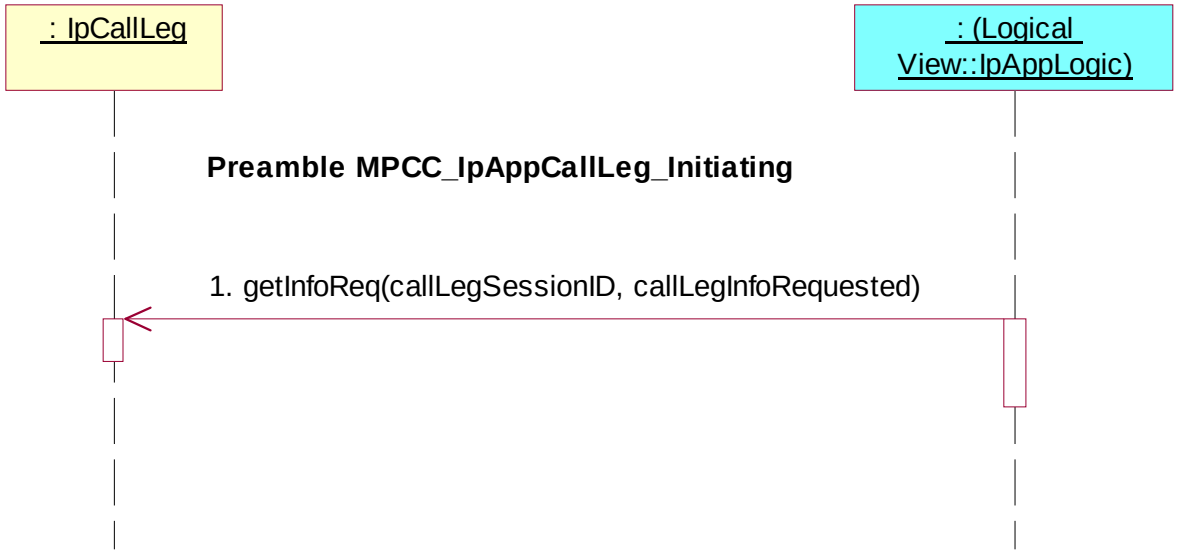
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **MPCC_IpAppCallLeg_Initiating**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID, callLegInfoRequested



Test MPCC_IpAppCallLeg_08

Summary: get information about call leg, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.1

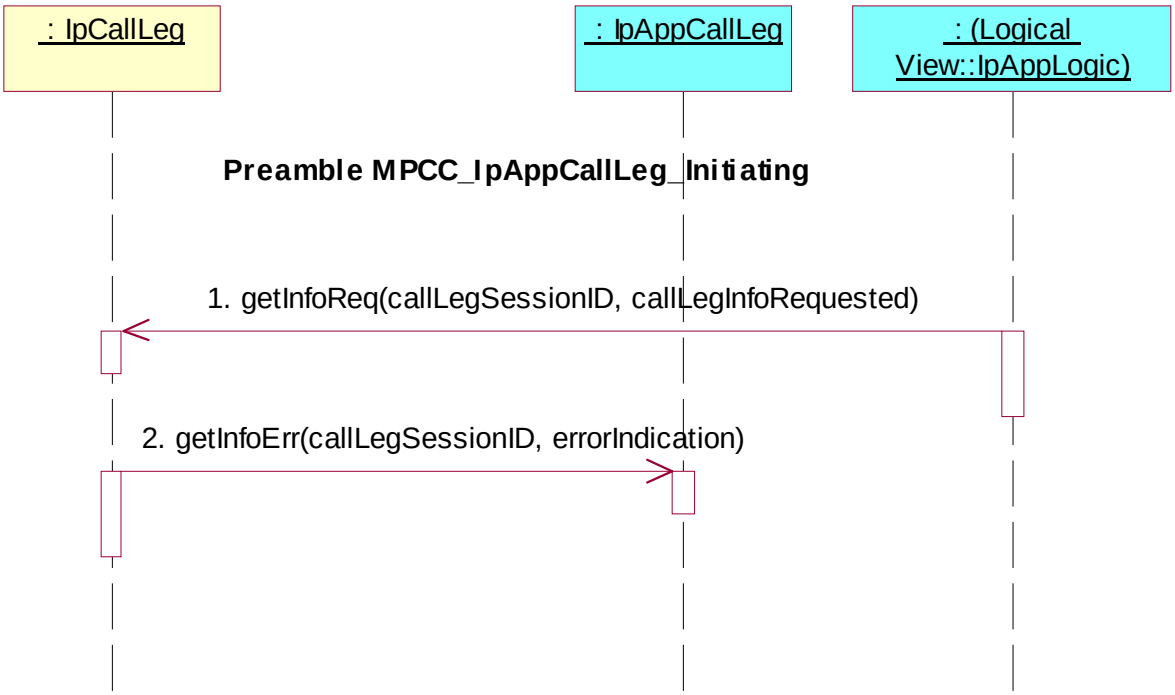
Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **MPCC_IpAppCallLeg_Initiating**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID, callLegInfoRequested

2. Method call **getInfoErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MPCC_IpAppCallLeg_09

Summary: set charge plan for call leg

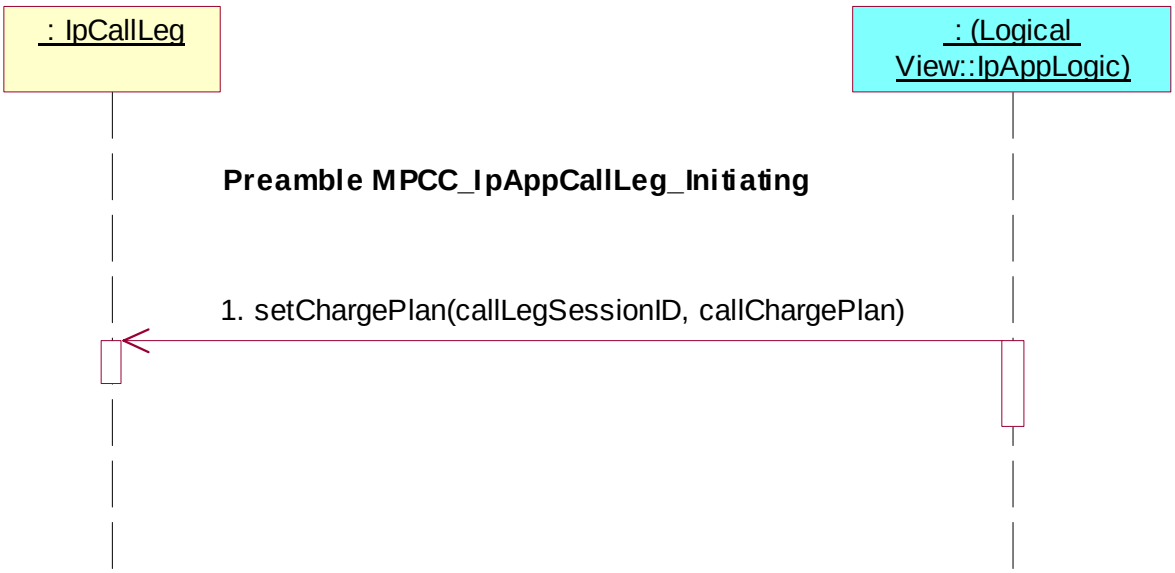
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **setChargePlan()**

Preamble: **MPCC_IpAppCallLeg_Initiating**

Test Sequence:

1. Triggered Action: cause IUT to call **setChargePlan()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID, callChargePlan



Test MPCC_IpAppCallLeg_10

Summary: allow advice of charge information

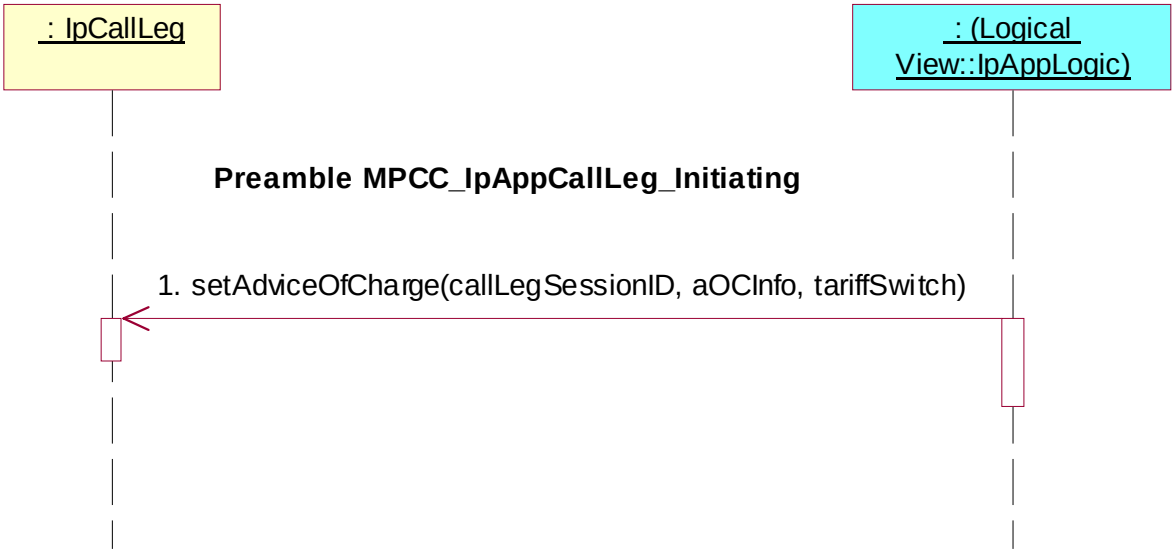
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **setAdviceOfCharge()**

Preamble: **MPCC_IpAppCallLeg_Initiating**

Test Sequence:

1. Triggered Action: cause IUT to call **setAdviceOfCharge()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID, aOCInfo, tariffSwitch



Test MPCC_IpAppCallLeg_11

Summary: supervise call leg

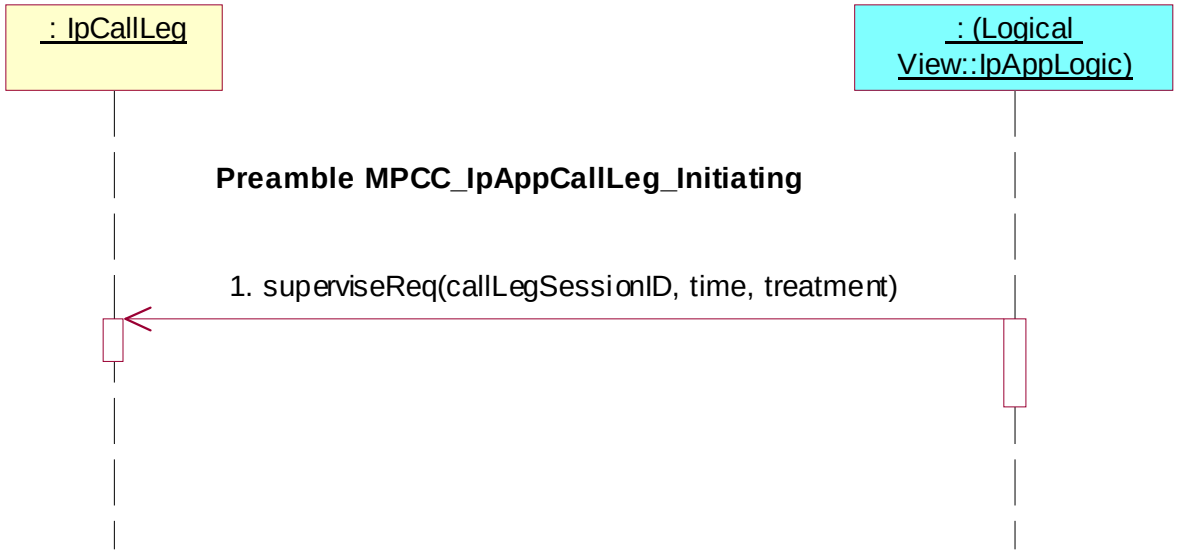
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **superviseReq()**

Preamble: **MPCC_IpAppCallLeg_Initiating**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID, time, treatment



Test MPCC_IpAppCallLeg_12

Summary: supervise call leg, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.1

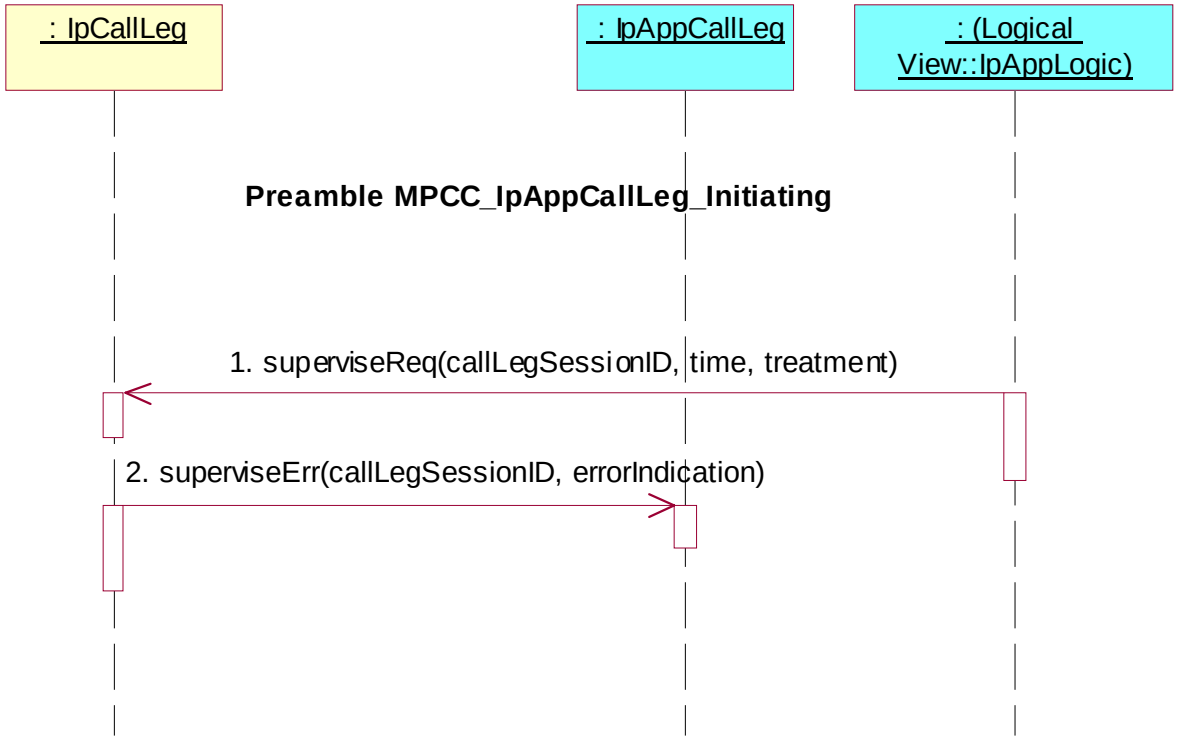
Precondition: IUT capable of invoking **superviseReq()**

Preamble: **MPCC_IpAppCallLeg_Initiating**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID, time, treatment

2. Method call **superviseErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



7.2.2.3.1.2 Analysing state

Preamble MPCC_IpAppCallLeg_Analysing

Reference: ES 202 915-4-3 [3], clause 7.3.1.2

Pre-preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiPartyCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiPartyCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Preamble Sequence:

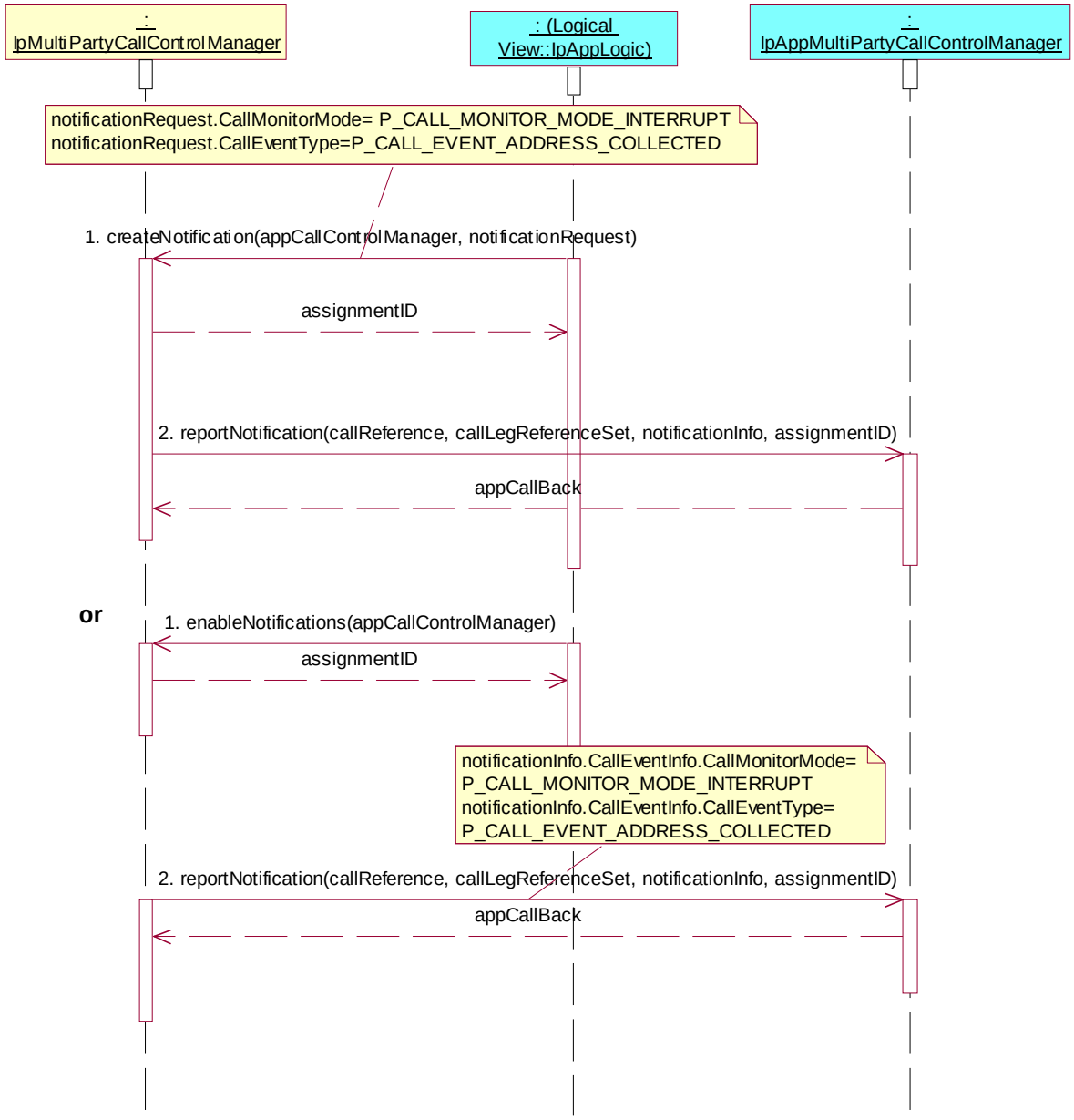
1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest
   notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   notificationRequest.CallEventType= P_CALL_EVENT_ADDRESS_COLLECTED

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   Check: valid value of TpAppMultiPartyCallBack is returned

or

1. Triggered Action: cause IUT to call **enableNotifications()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCallControlManager

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   notificationInfo.CallEventInfo.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   notificationInfo.CallEventInfo.CallEventType= P_CALL_EVENT_ADDRESS_COLLECTED
   Check: valid value of TpAppMultiPartyCallBack is returned



Test MPCC_IpAppCallLeg_13

Summary: attach media, successful

Reference: ES 202 915-4-3 [3], clause 7.3.1

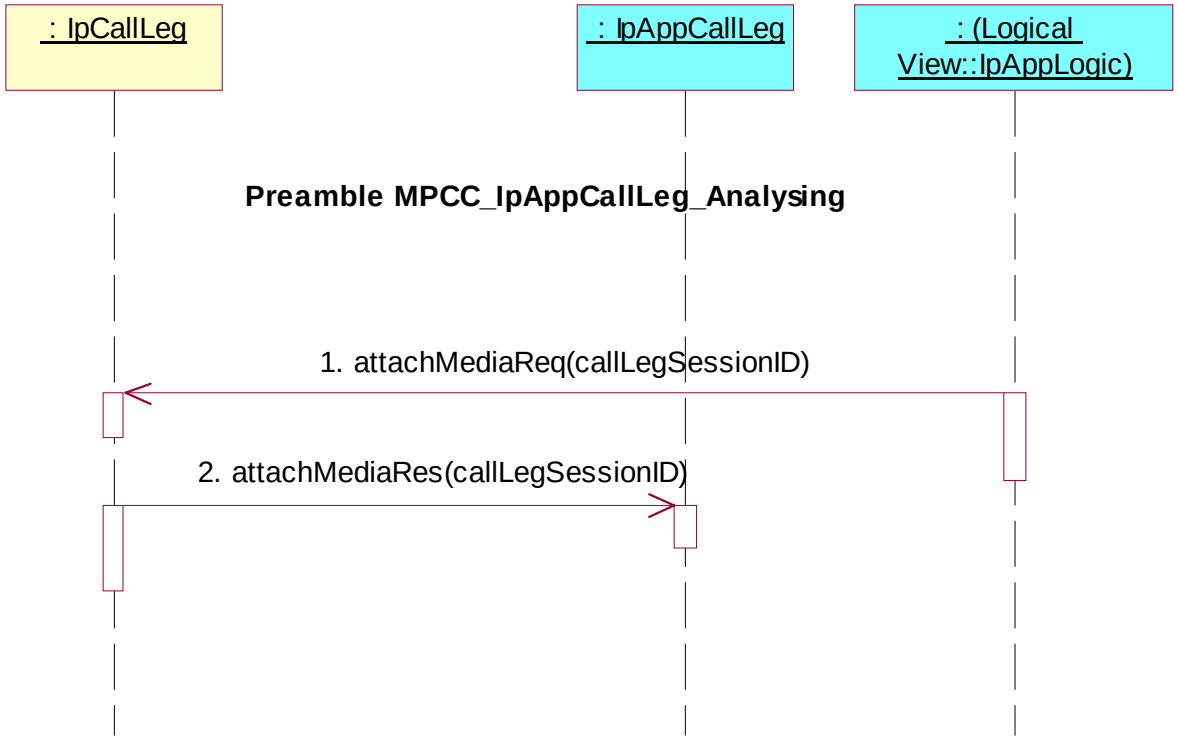
Precondition: IUT capable of invoking **attachMediaReq()**

Preamble: **MPCC_IpAppCallLeg_Analysing**

Test Sequence:

1. Triggered Action: cause IUT to call **attachMediaReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID

2. Method call **attachMediaRes()**
   Parameters: callLegSessionID
   Check: no exception is returned



Test MPCC_IpAppCallLeg_14

Summary: attach media, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.1

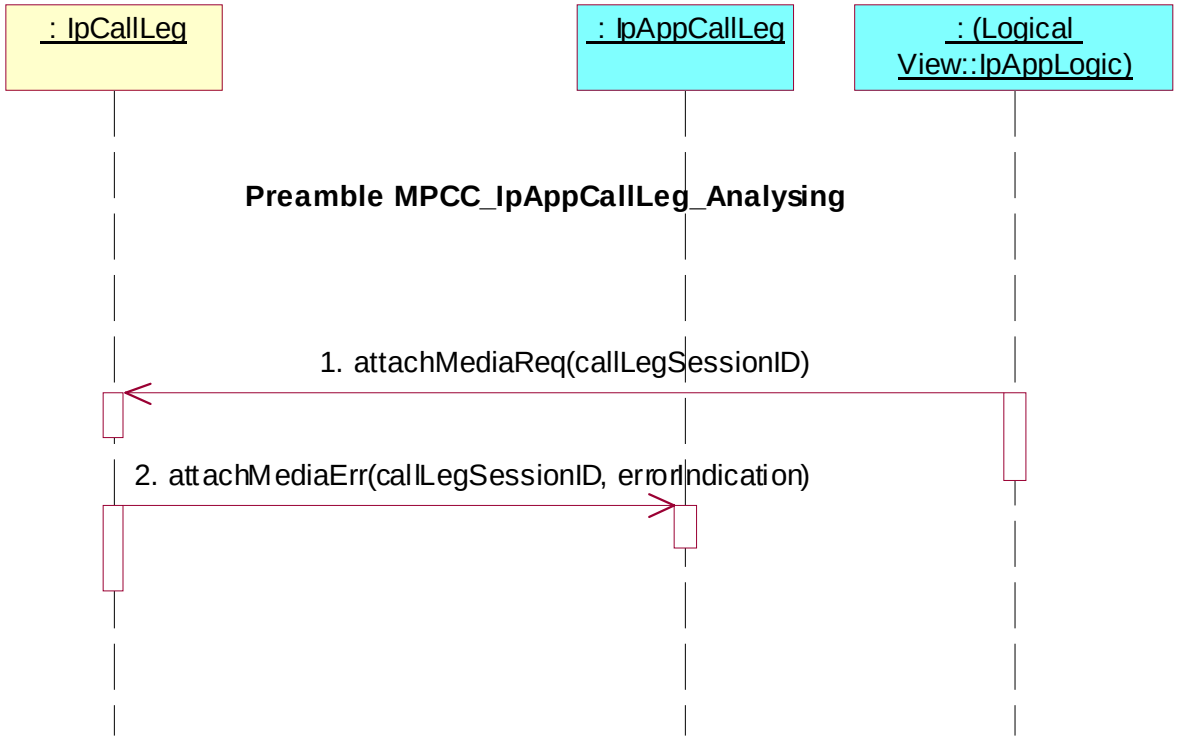
Precondition: IUT capable of invoking **attachMediaReq()**

Preamble: **MPCC_IpAppCallLeg_Analysing**

Test Sequence:

1. Triggered Action: cause IUT to call **attachMediaReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID

2. Method call **attachMediaErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MPCC_IpAppCallLeg_15

Summary: detach media, successful

Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **detachMediaReq()**

Preamble: **MPCC_IpAppCallLeg_Analysing**

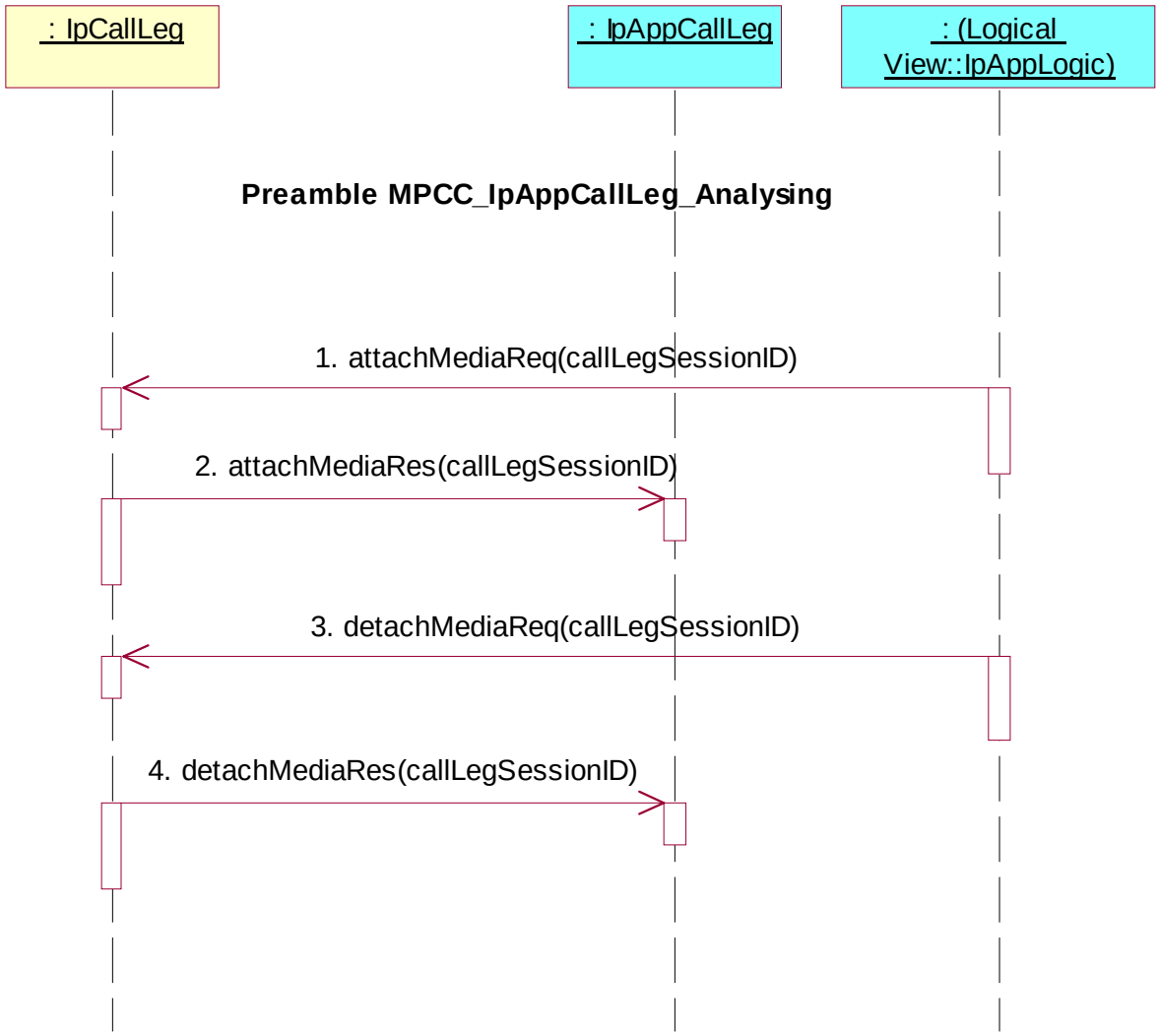
Test Sequence:

1. Triggered Action: cause IUT to call **attachMediaReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID

2. Method call **attachMediaRes()**
   Parameters: callLegSessionID
   Check: no exception is returned

3. Triggered Action: cause IUT to call **detachMediaReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID

4. Method call **detachMediaRes()**
   Parameters: callLegSessionID
   Check: no exception is returned



Test MPCC_IpAppCallLeg_16

Summary: detach media, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **detachMediaReq()**

Preamble: **MPCC_IpAppCallLeg_Analysing**

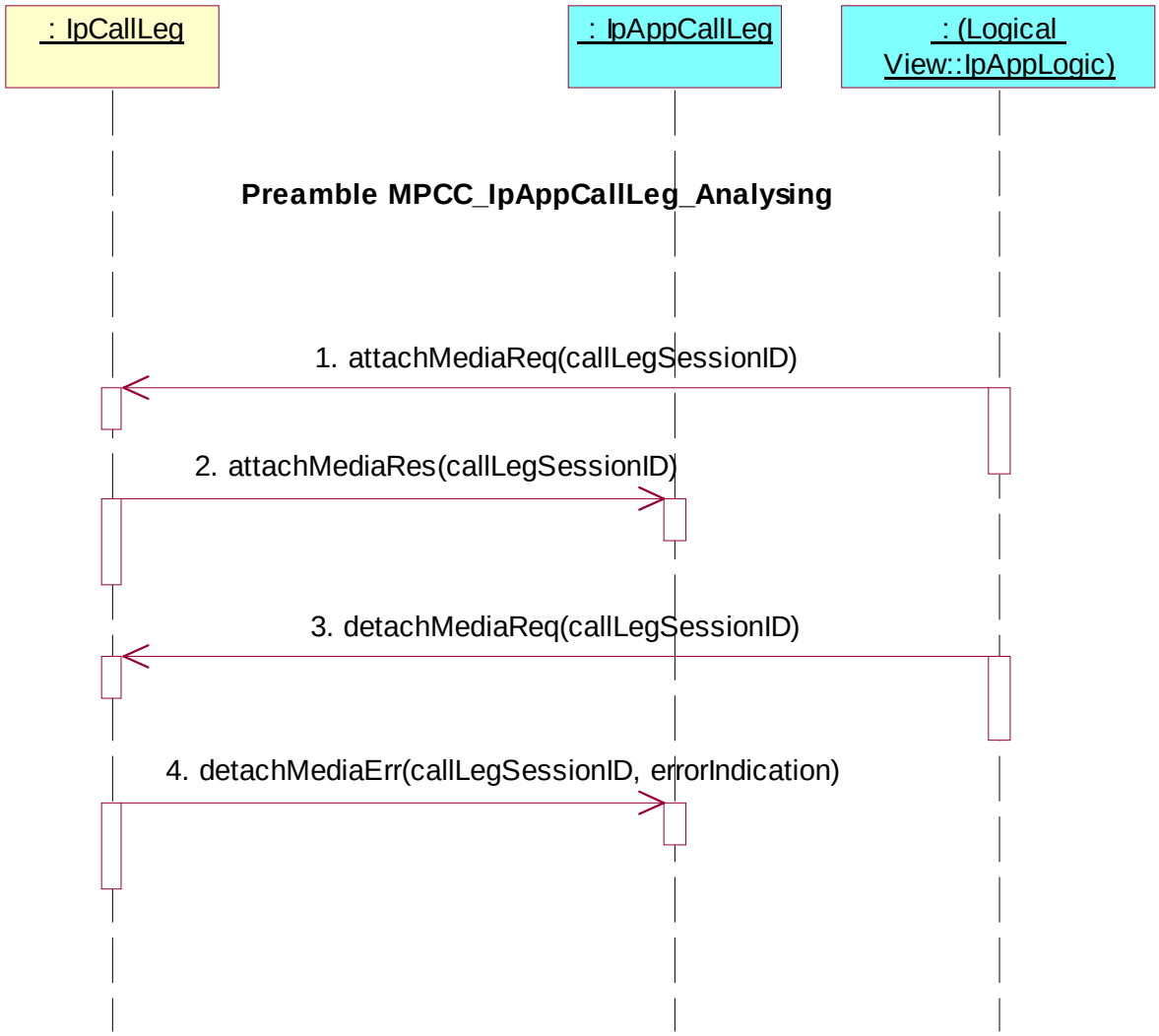
Test Sequence:

1. Triggered Action: cause IUT to call **attachMediaReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID

2. Method call **attachMediaRes()**
   Parameters: callLegSessionID
   Check: no exception is returned

3. Triggered Action: cause IUT to call **detachMediaReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID

4. Method call **detachMediaErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MPCC_IpAppCallLeg_17

Summary: request reference of call related to call leg

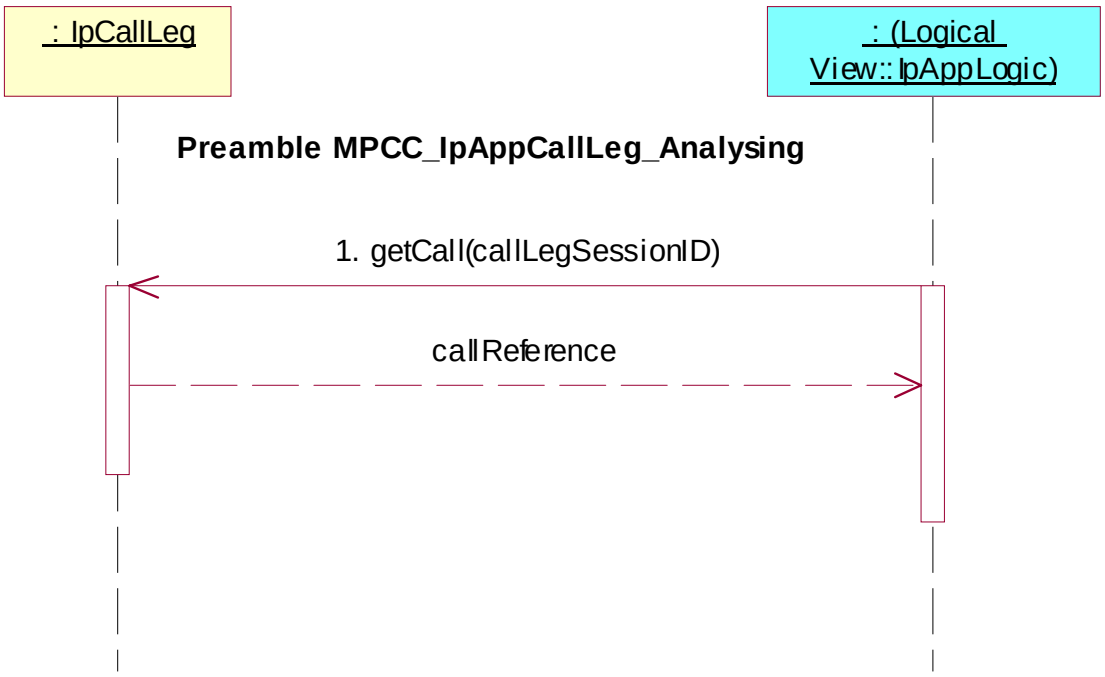
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **getCall()**

Preamble: **MPCC_IpAppCallLeg_Analysing**

Test Sequence:

1. Triggered Action: cause IUT to call **getCall()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID



Test MPCC_IpAppCallLeg_18

Summary: continue processing of call leg

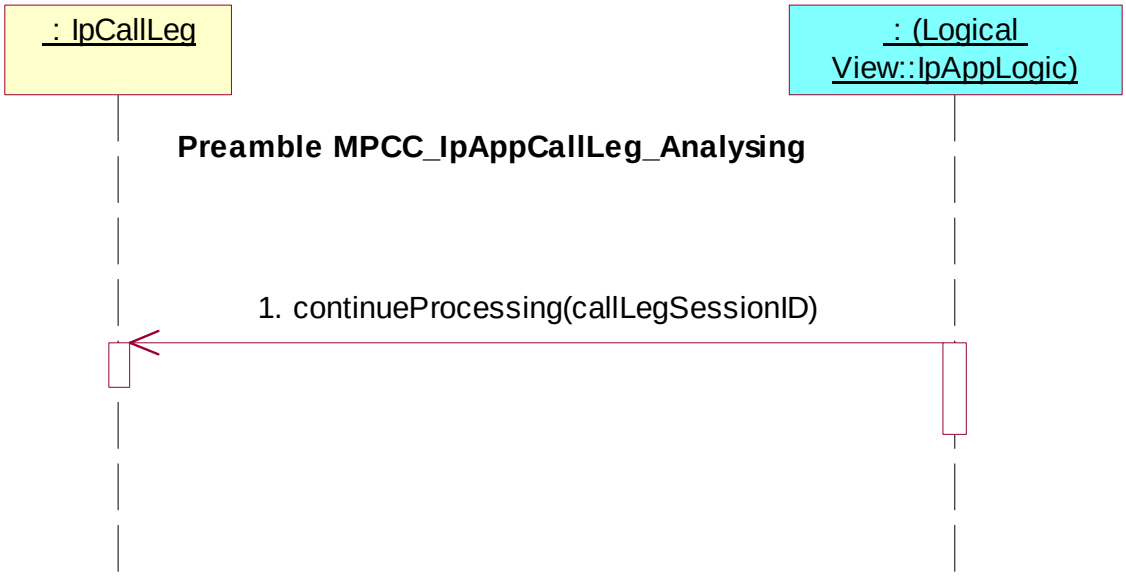
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **continueProcessing()**

Preamble: **MPCC_IpAppCallLeg_Analysing**

Test Sequence:

1. Triggered Action: cause IUT to call **continueProcessing()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID



Test MPCC_IpAppCallLeg_19

Summary: release call leg

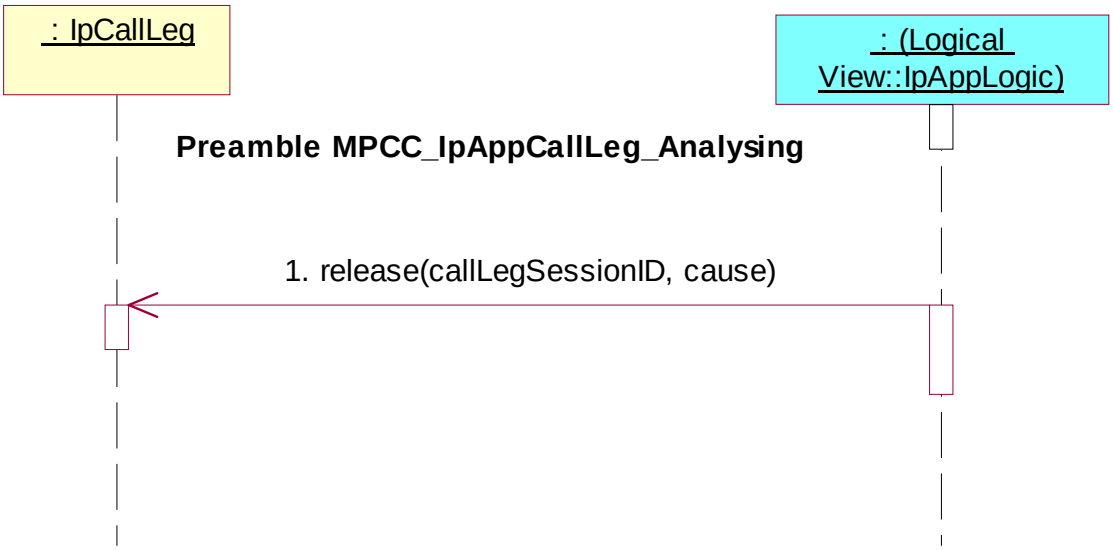
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **release()**

Preamble: **MPCC_IpAppCallLeg_Analysing**

Test Sequence:

1. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID, cause



Test MPCC_IpAppCallLeg_20

Summary: de-assign call leg

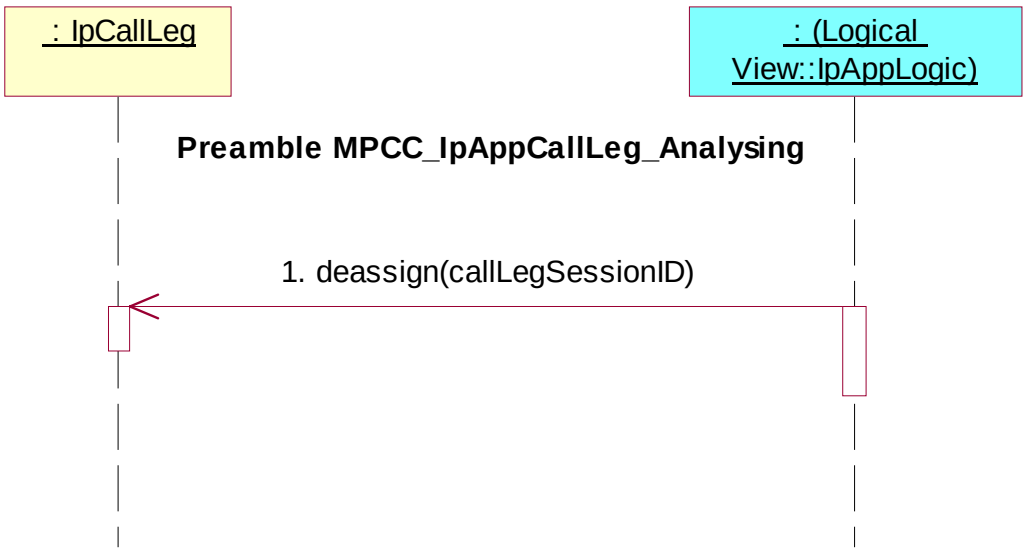
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **deassign()**

Preamble: **MPCC_IpAppCallLeg_Analysing**

Test Sequence:

1. Triggered Action: cause IUT to call **deassign()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID



Test MPCC_IpAppCallLeg_21

Summary: change or clear event criteria

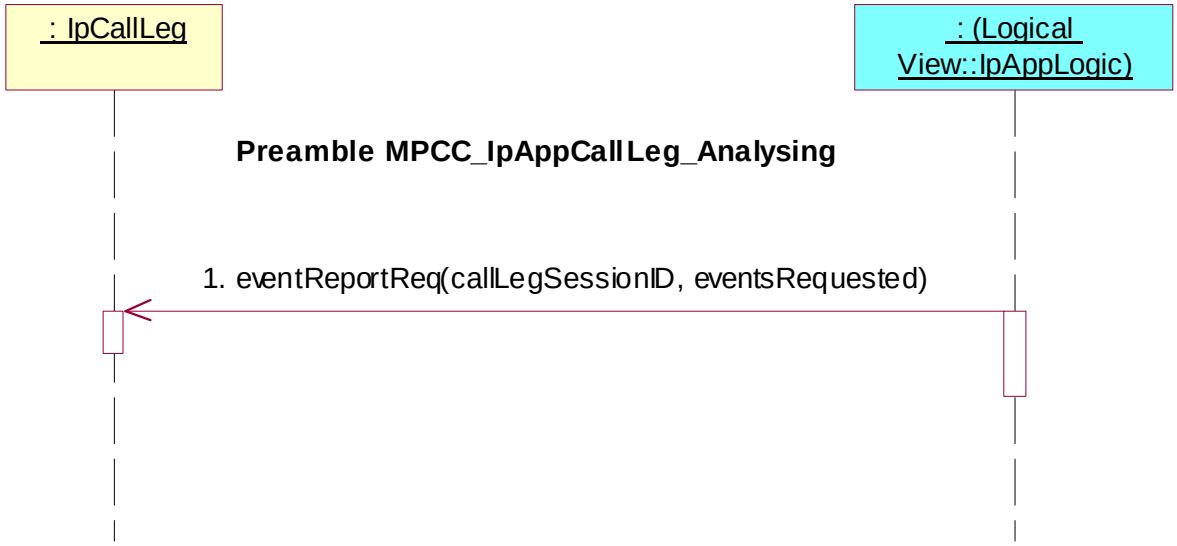
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **eventReportReq()**

Preamble: **MPCC_IpAppCallLeg_Analysing**

Test Sequence:

1. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID, eventsRequested



Test MPCC_IpAppCallLeg_22

Summary: change or clear event criteria, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.1

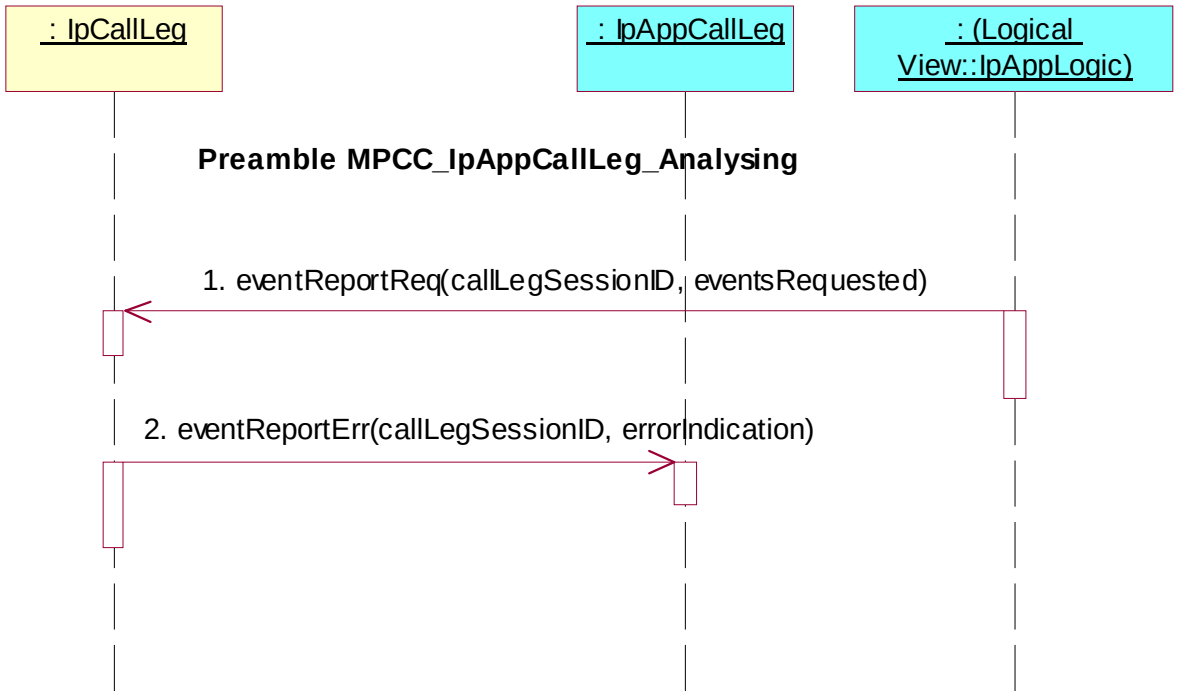
Precondition: IUT capable of invoking **eventReportReq()**

Preamble: **MPCC_IpAppCallLeg_Analysing**

Test Sequence:

1. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID, eventsRequested

2. Method call **eventReportErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MPCC_IpAppCallLeg_23

Summary: get information about call leg

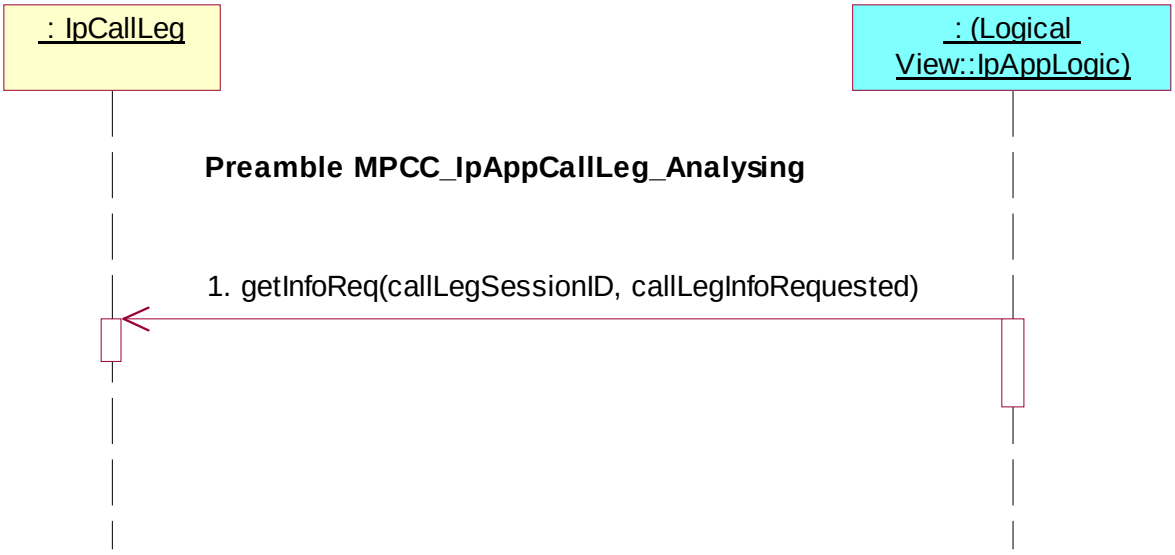
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **MPCC_IpAppCallLeg_Analysing**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID, callLegInfoRequested



Test MPCC_IpAppCallLeg_24

Summary: get information about call leg, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.1

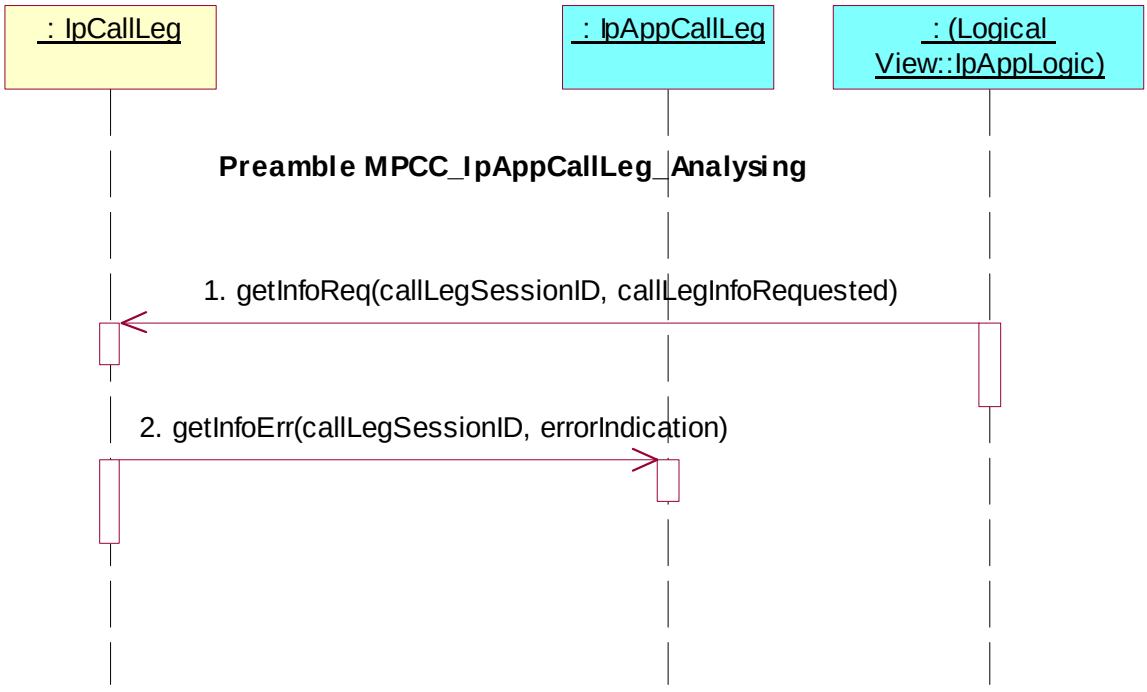
Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **MPCC_IpAppCallLeg_Analysing**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID, callLegInfoRequested

2. Method call **getInfoErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MPCC_IpAppCallLeg_25

Summary: set charge plan for call leg

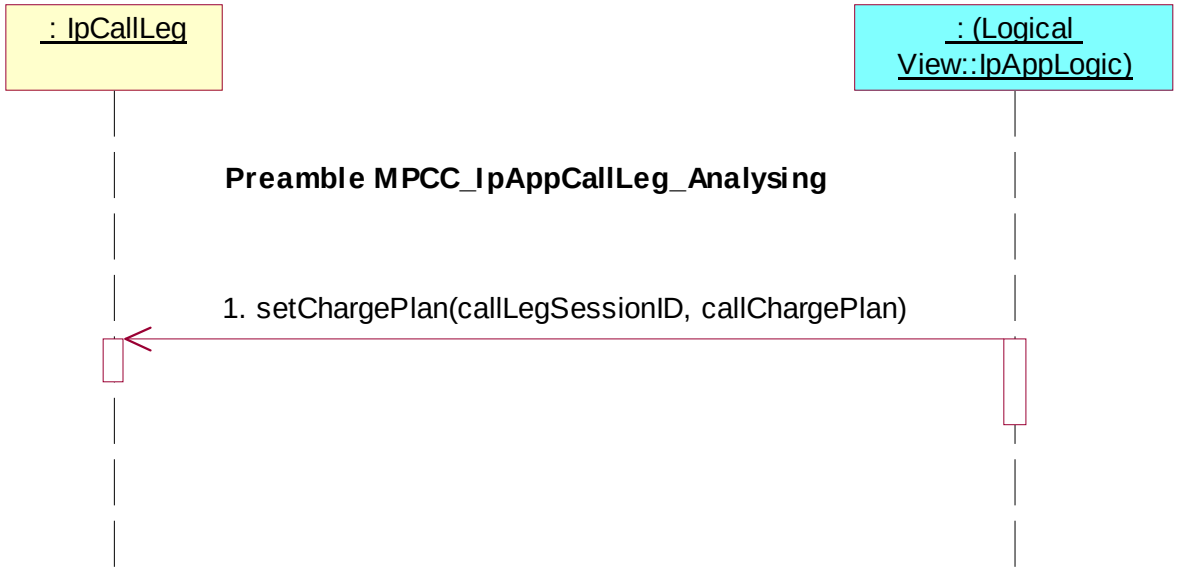
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **setChargePlan()**

Preamble: **MPCC_IpAppCallLeg_Analysing**

Test Sequence:

1. Triggered Action: cause IUT to call **setChargePlan()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID, callChargePlan



Test MPCC_IpAppCallLeg_26

Summary: allow advice of charge information

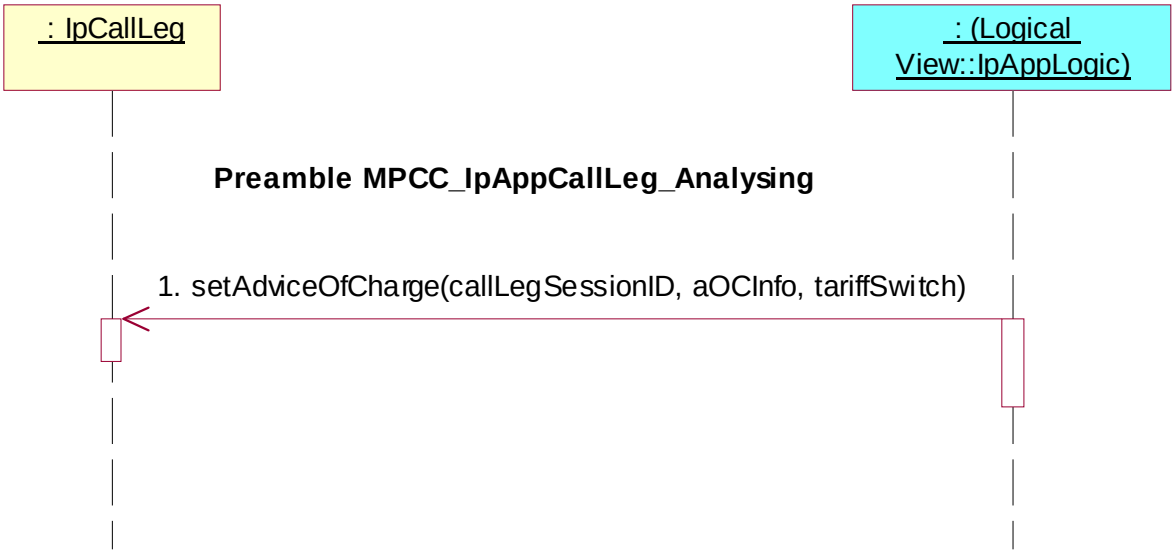
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **setAdviceOfCharge()**

Preamble: **MPCC_IpAppCallLeg_Analysing**

Test Sequence:

1. Triggered Action: cause IUT to call **setAdviceOfCharge()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID, aOCInfo, tariffSwitch



Test MPCC_IpAppCallLeg_27

Summary: supervise call leg

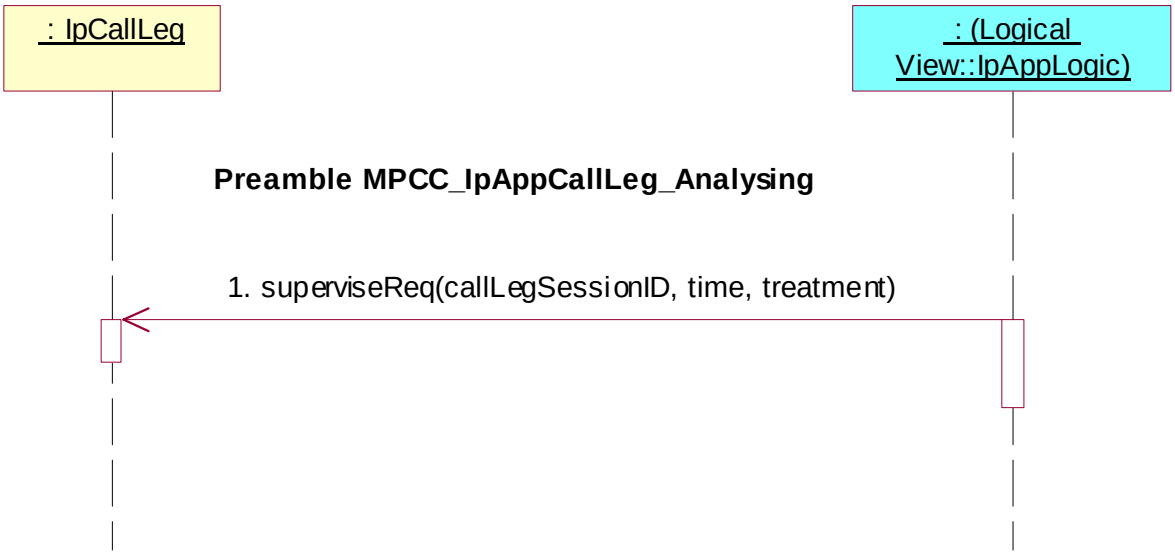
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **superviseReq()**

Preamble: **MPCC_IpAppCallLeg_Analysing**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID, time, treatment



Test MPCC_IpAppCallLeg_28

Summary: supervise call leg, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.1

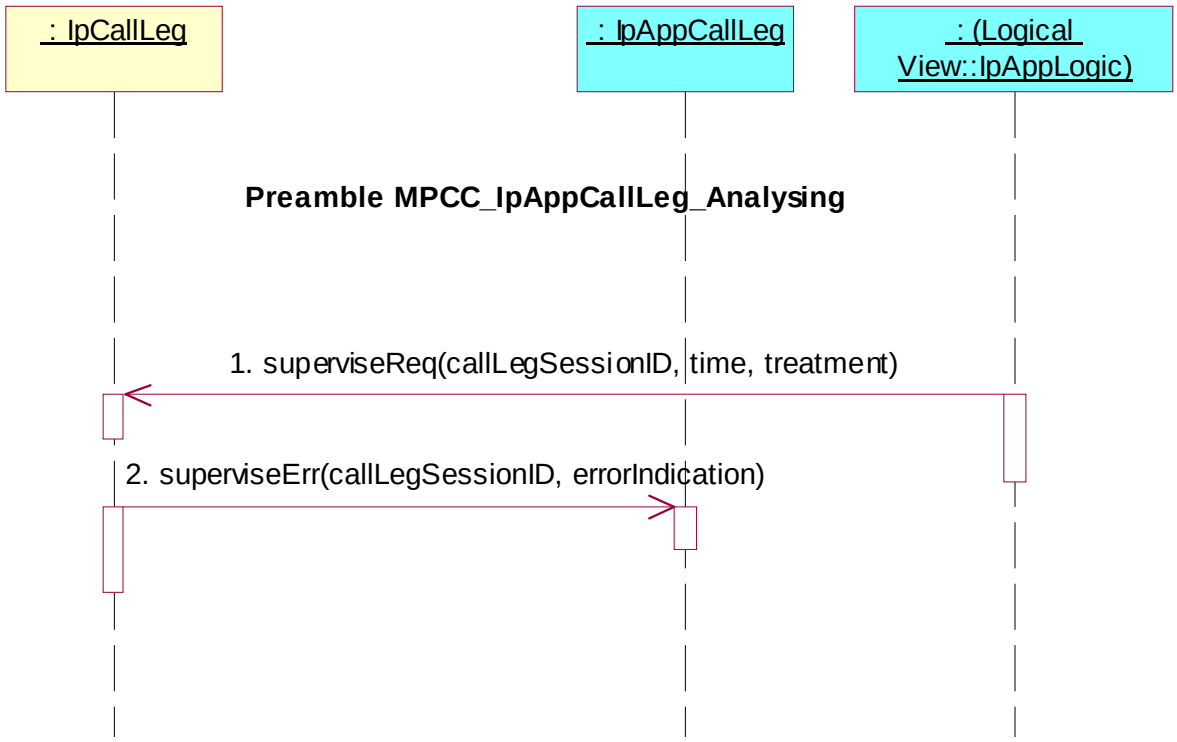
Precondition: IUT capable of invoking **superviseReq()**

Preamble: **MPCC_IpAppCallLeg_Analysing**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID, time, treatment

2. Method call **superviseErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



7.2.2.3.1.3 Active state

Preamble MPCC_IpAppCallLeg_Active

Reference: ES 202 915-4-3 [3], clause 7.3.1.3

Pre-preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiPartyCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiPartyCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Preamble Sequence:

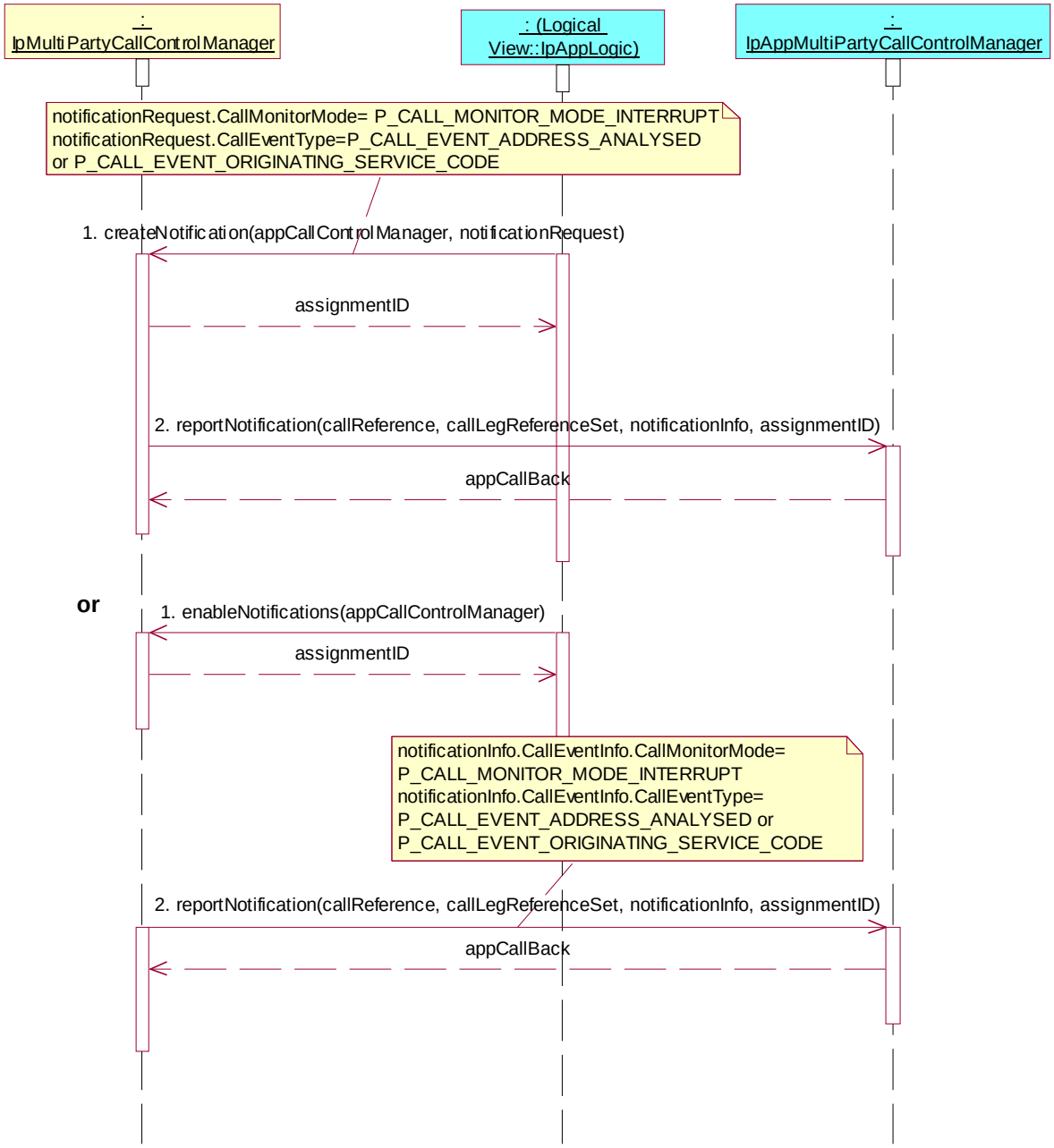
1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest
   notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   notificationRequest.CallEventType= P_CALL_EVENT_ADDRESS_ANALYSED or P_CALL_EVENT_ORIGINATING_SERVICE_CODE

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   Check: valid value of TpAppMultiPartyCallBack is returned

or

1. Triggered Action: cause IUT to call **enableNotifications()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCallControlManager

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   notificationInfo.CallEventInfo.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   notificationInfo.CallEventInfo.CallEventType= P_CALL_EVENT_ADDRESS_ANALYSED or P_CALL_EVENT_ORIGINATING_SERVICE_CODE
   Check: valid value of TpAppMultiPartyCallBack is returned


: IpMultiPartyCallControlManager
: (Logical View::IpAppLogic)
: IpAppMultiPartyCallControlManager
notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
notificationRequest.CallEventType=P_CALL_EVENT_ADDRESS_ANALYSED
or P_CALL_EVENT_ORIGINATING_SERVICE_CODE
1. createNotification(appCallControlManager, notificationRequest)
assignmentID
2. reportNotification(callReference, callLegReferenceSet, notificationInfo, assignmentID)
appCallBack
or
1. enableNotifications(appCallControlManager)
assignmentID
notificationInfo.CallEventInfo.CallMonitorMode=
P_CALL_MONITOR_MODE_INTERRUPT
notificationInfo.CallEventInfo.CallEventType=
P_CALL_EVENT_ADDRESS_ANALYSED or
P_CALL_EVENT_ORIGINATING_SERVICE_CODE
2. reportNotification(callReference, callLegReferenceSet, notificationInfo, assignmentID)
appCallBack

Test MPCC_IpAppCallLeg_29

Summary: attach media, successful

Reference: ES 202 915-4-3 [3], clause 7.3.1

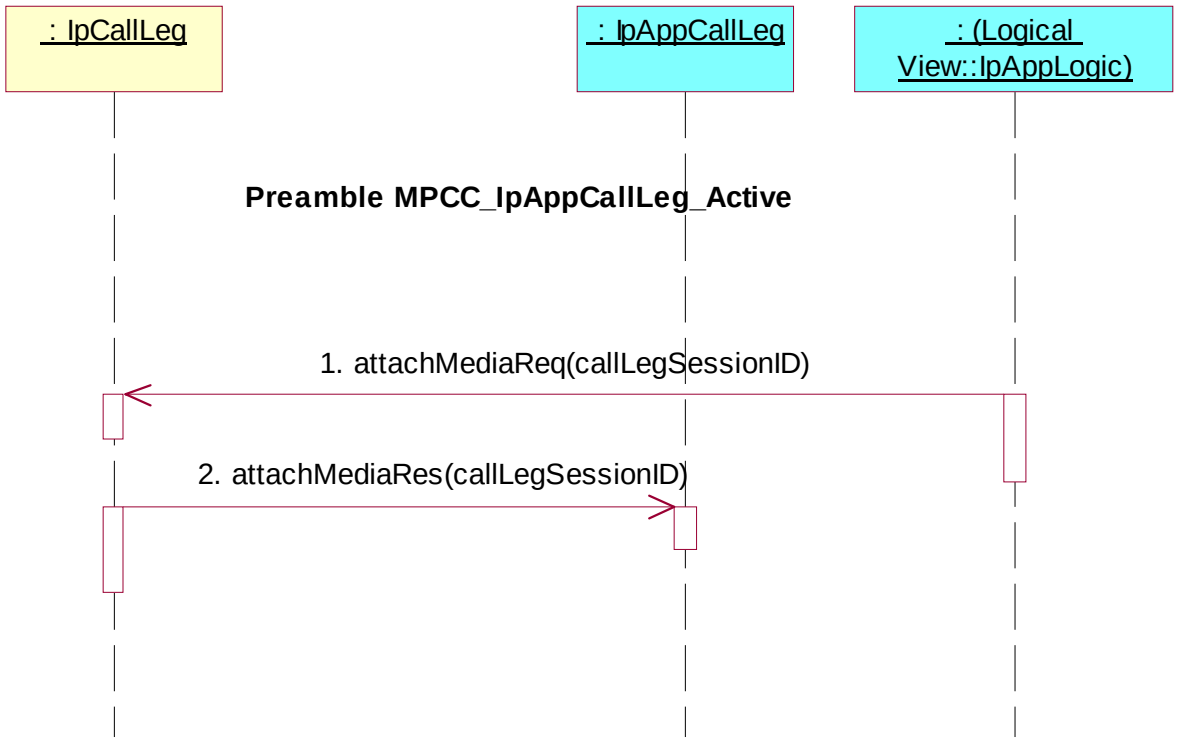
Precondition: IUT capable of invoking **attachMediaReq()**

Preamble: **MPCC_IpAppCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **attachMediaReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID

2. Method call **attachMediaRes()**
   Parameters: callLegSessionID
   Check: no exception is returned



Test MPCC_IpAppCallLeg_30

Summary: attach media, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.1

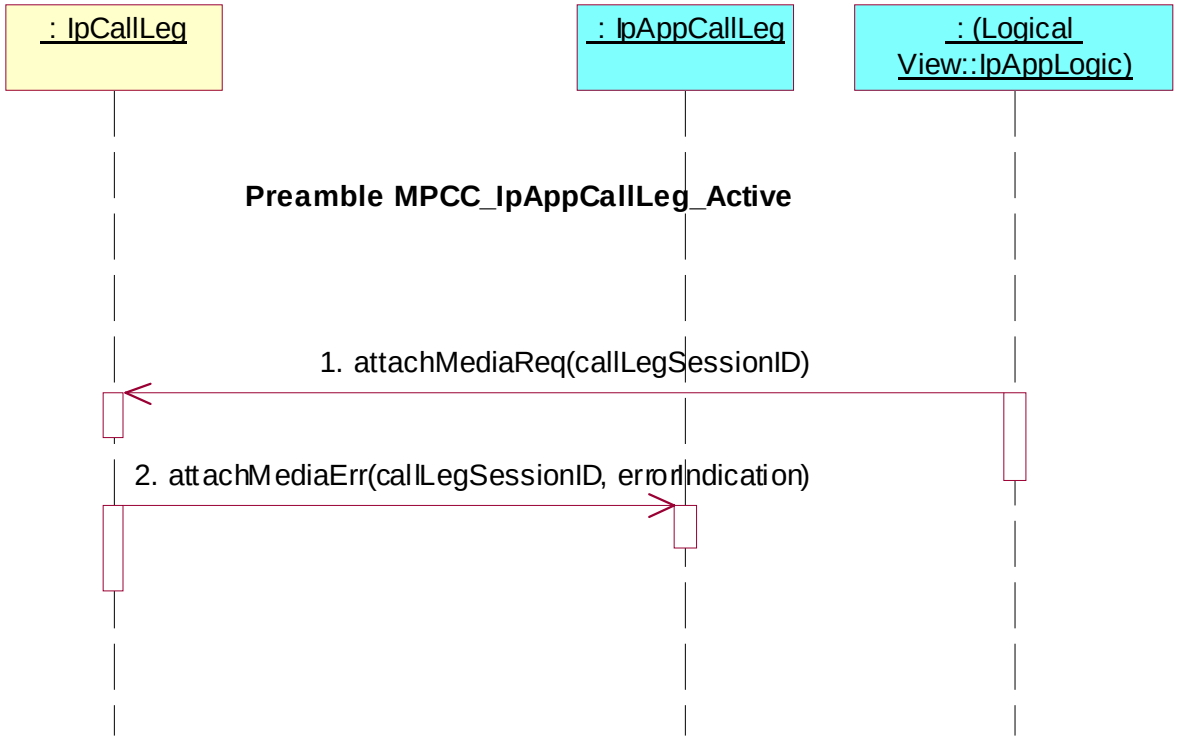
Precondition: IUT capable of invoking **attachMediaReq()**

Preamble: **MPCC_IpAppCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **attachMediaReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID

2. Method call **attachMediaErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MPCC_IpAppCallLeg_31

Summary: detach media, successful

Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **detachMediaReq()**

Preamble: **MPCC_IpAppCallLeg_Active**

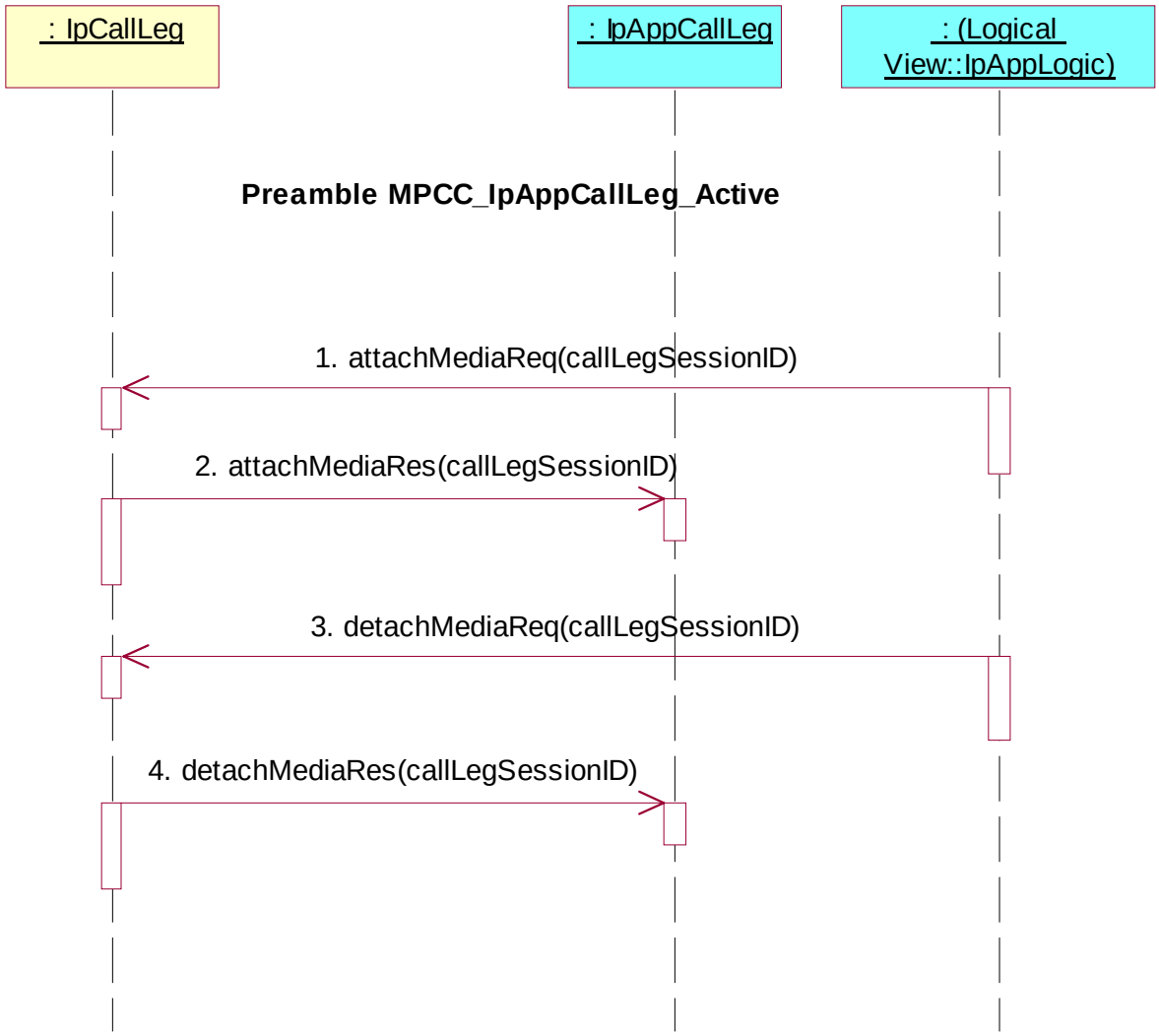
Test Sequence:

1. Triggered Action: cause IUT to call **attachMediaReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID

2. Method call **attachMediaRes()**
   Parameters: callLegSessionID
   Check: no exception is returned

3. Triggered Action: cause IUT to call **detachMediaReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID

4. Method call **detachMediaRes()**
   Parameters: callLegSessionID
   Check: no exception is returned



Test MPCC_IpAppCallLeg_32

Summary: detach media, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **detachMediaReq()**

Preamble: **MPCC_IpAppCallLeg_Active**

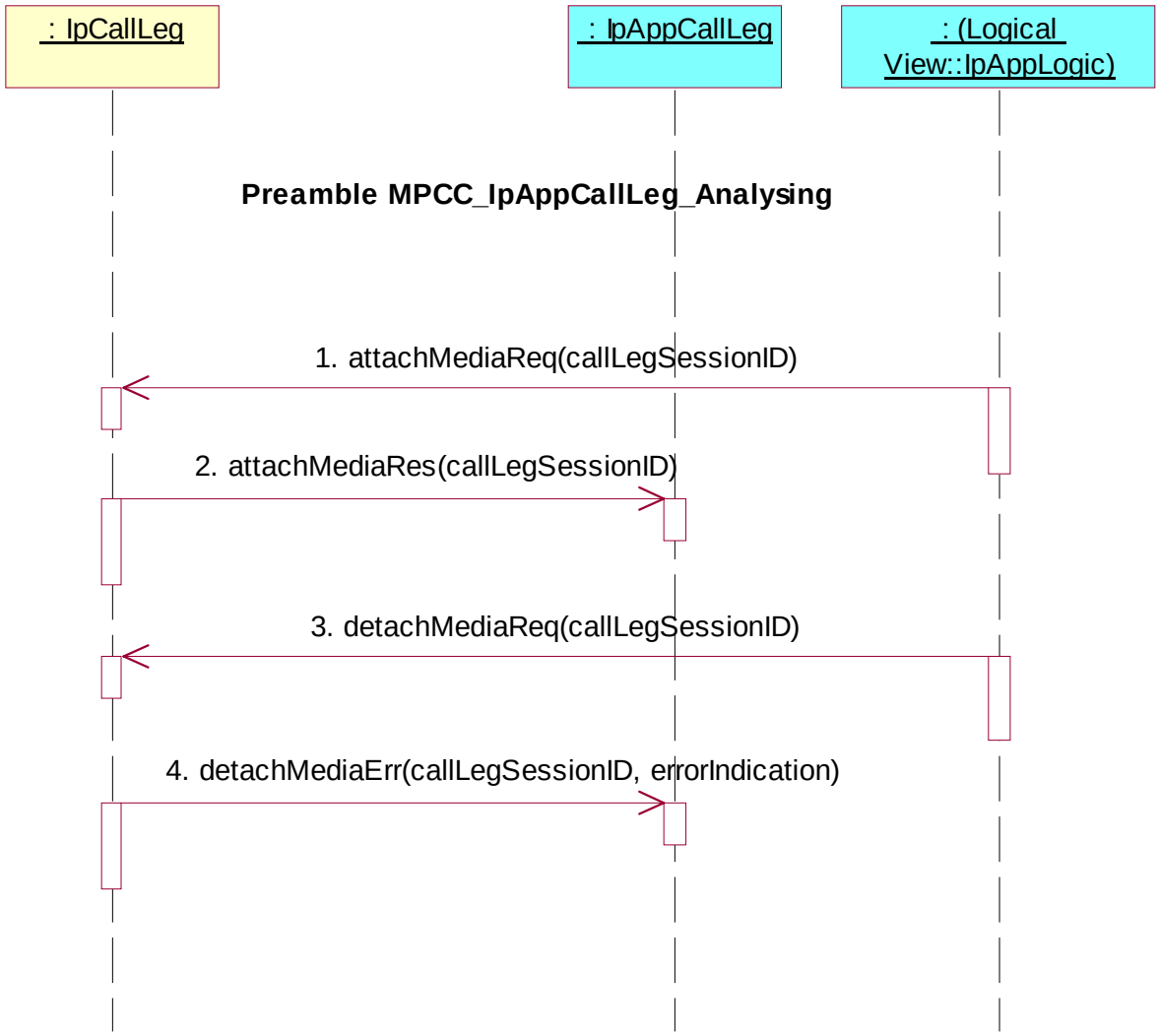
Test Sequence:

1. Triggered Action: cause IUT to call **attachMediaReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID

2. Method call **attachMediaRes()**
   Parameters: callLegSessionID
   Check: no exception is returned

3. Triggered Action: cause IUT to call **detachMediaReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID

4. Method call **detachMediaErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MPCC_IpAppCallLeg_33

Summary: request reference of call related to call leg

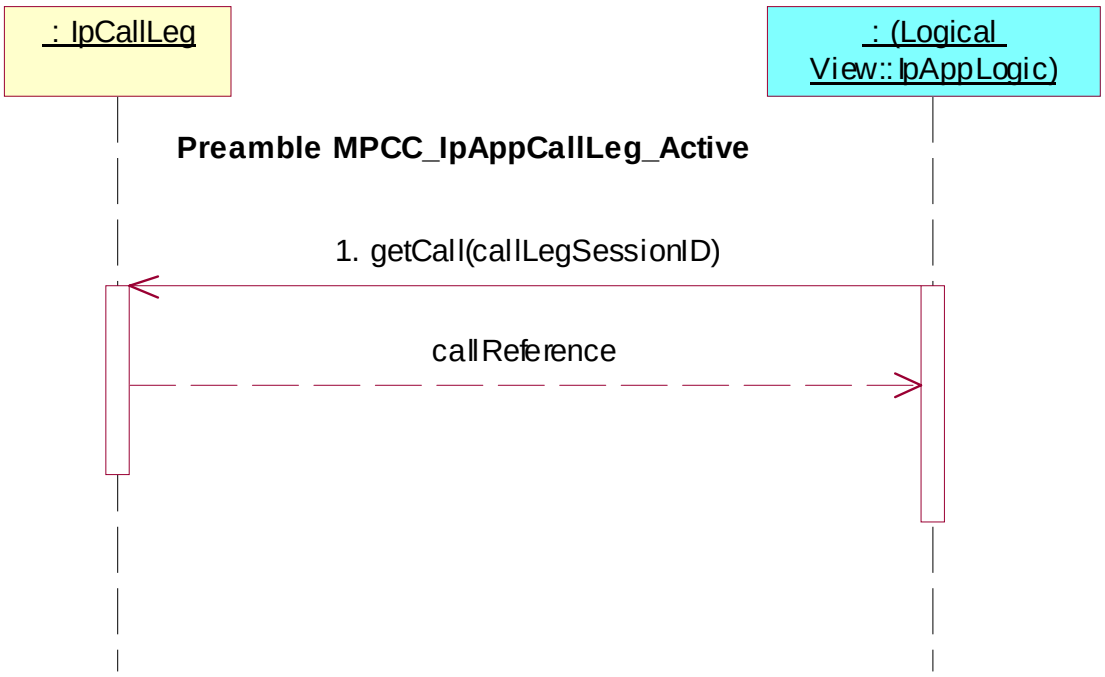
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **getCall()**

Preamble: **MPCC_IpAppCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **getCall()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID



Test MPCC_IpAppCallLeg_34

Summary: continue processing of call leg

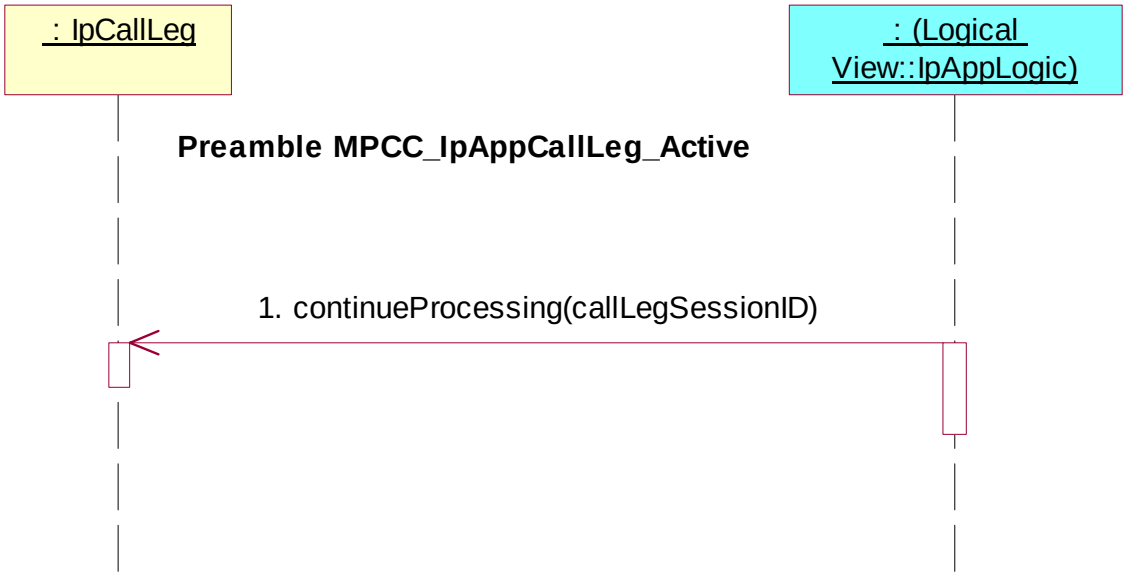
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **continueProcessing()**

Preamble: **MPCC_IpAppCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **continueProcessing()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID



Test MPCC_IpAppCallLeg_35

Summary: release call leg

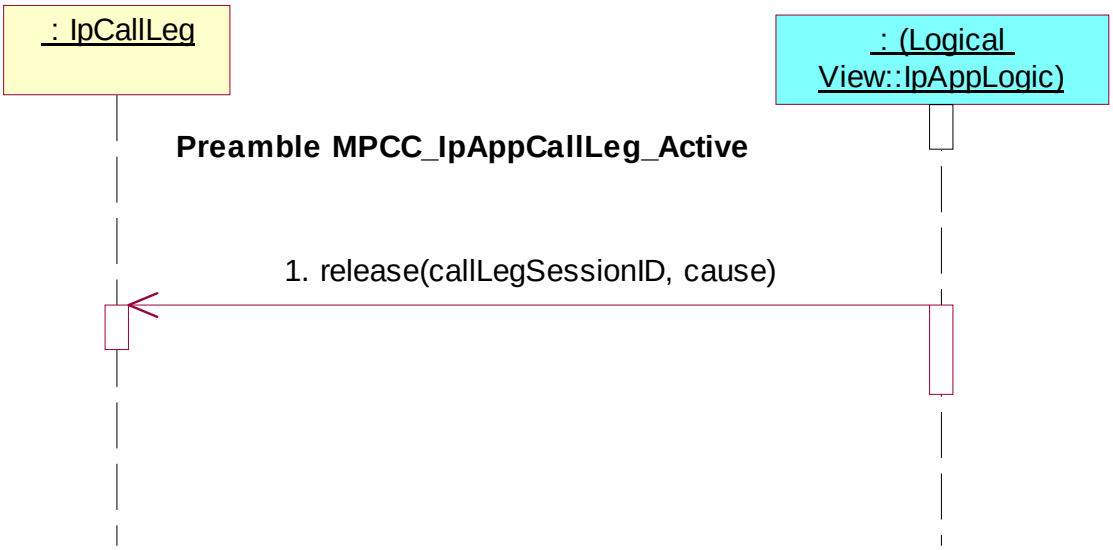
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **release()**

Preamble: **MPCC_IpAppCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID, cause



Test MPCC_IpAppCallLeg_36

Summary: de-assign call leg

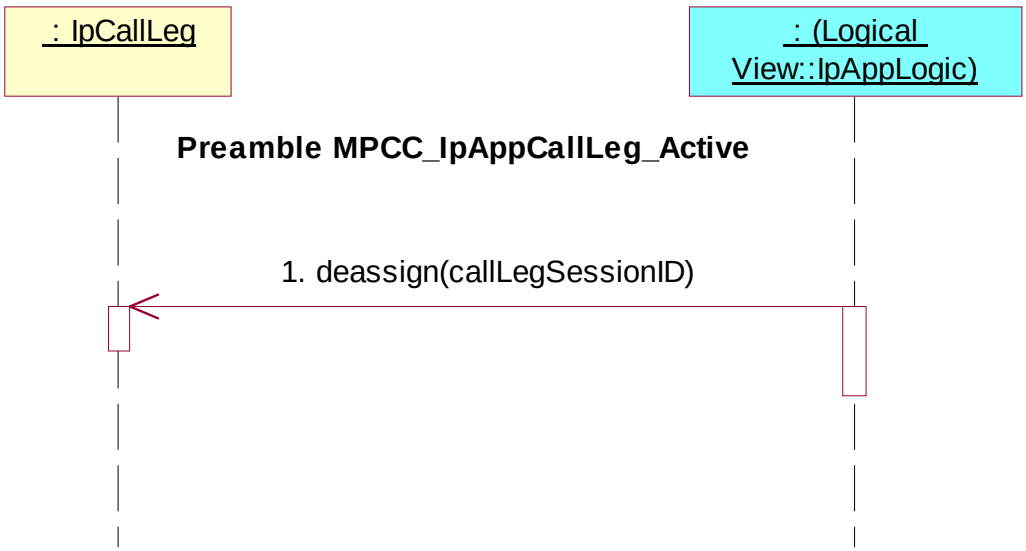
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **deassign()**

Preamble: **MPCC_IpAppCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **deassign()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID



Test MPCC_IpAppCallLeg_37

Summary: change or clear event criteria

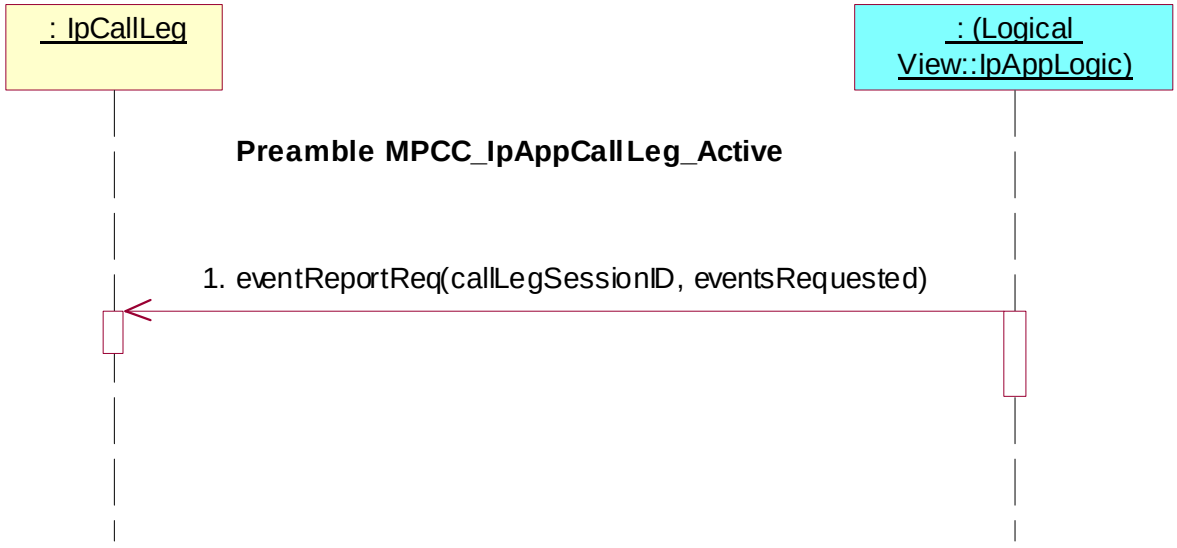
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **eventReportReq()**

Preamble: **MPCC_IpAppCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID, eventsRequested



Test MPCC_IpAppCallLeg_38

Summary: change or clear event criteria, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.1

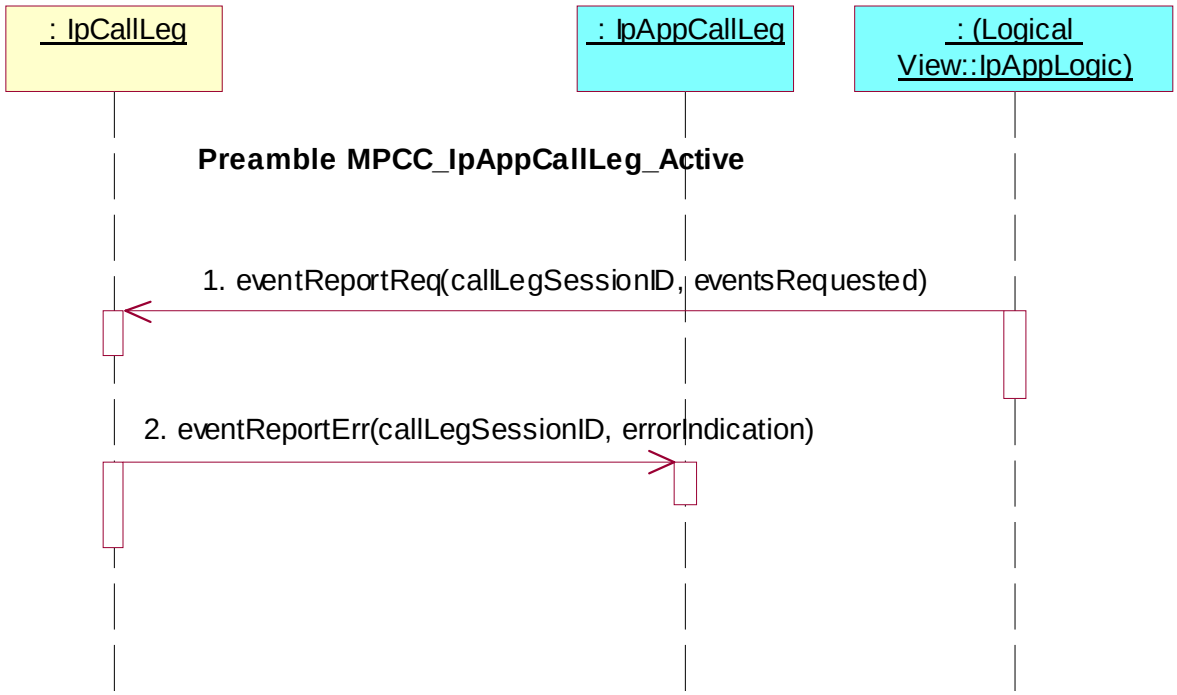
Precondition: IUT capable of invoking **eventReportReq()**

Preamble: **MPCC_IpAppCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID, eventsRequested

2. Method call **eventReportErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MPCC_IpAppCallLeg_39

Summary: get information about call leg

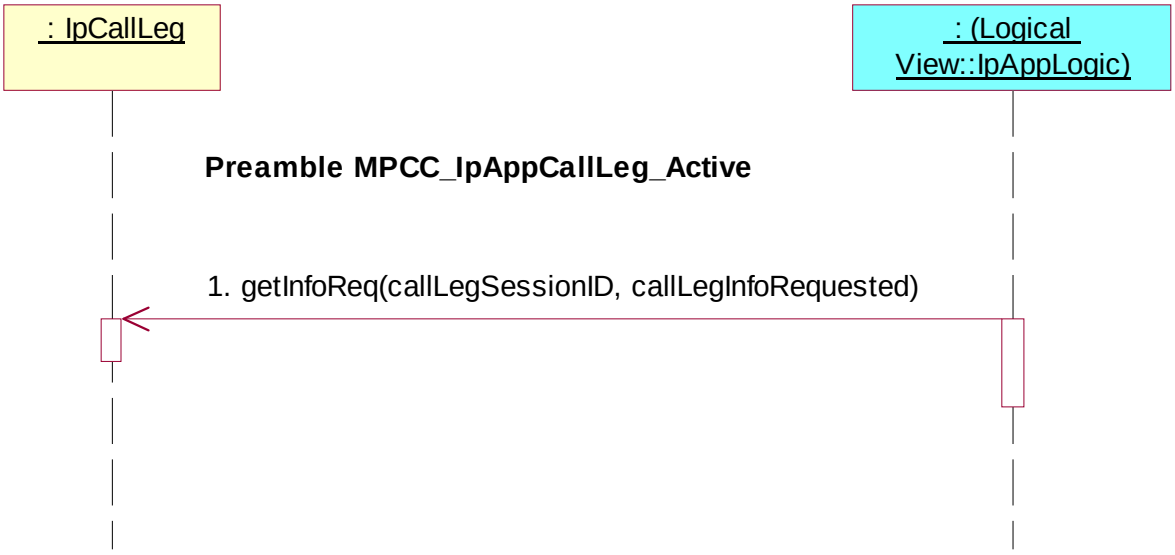
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **MPCC_IpAppCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID, callLegInfoRequested



Test MPCC_IpAppCallLeg_40

Summary: get information about call leg, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.1

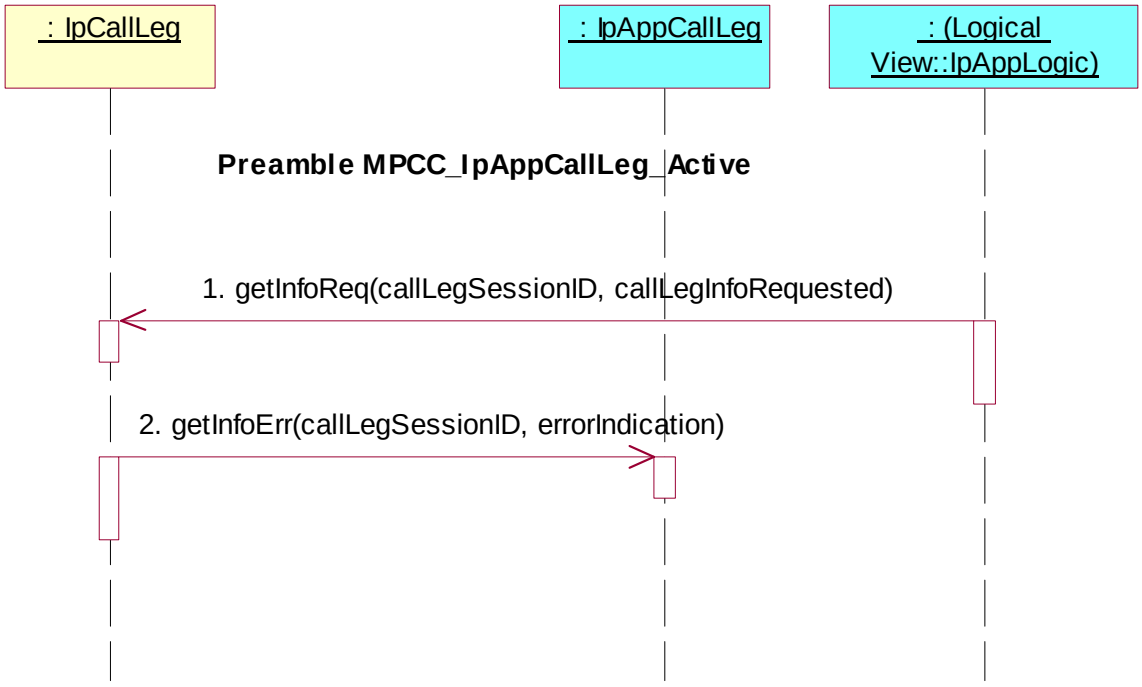
Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **MPCC_IpAppCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID, callLegInfoRequested

2. Method call **getInfoErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MPCC_IpAppCallLeg_41

Summary: set charge plan for call leg

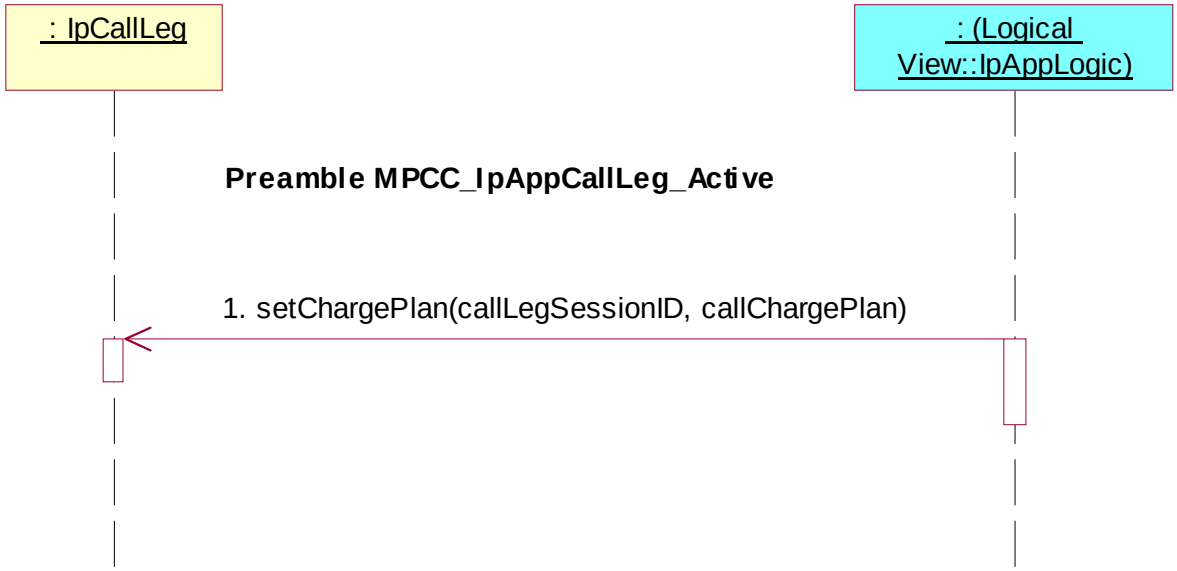
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **setChargePlan()**

Preamble: **MPCC_IpAppCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **setChargePlan()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID, callChargePlan



Test MPCC_IpAppCallLeg_42

Summary: allow advice of charge information

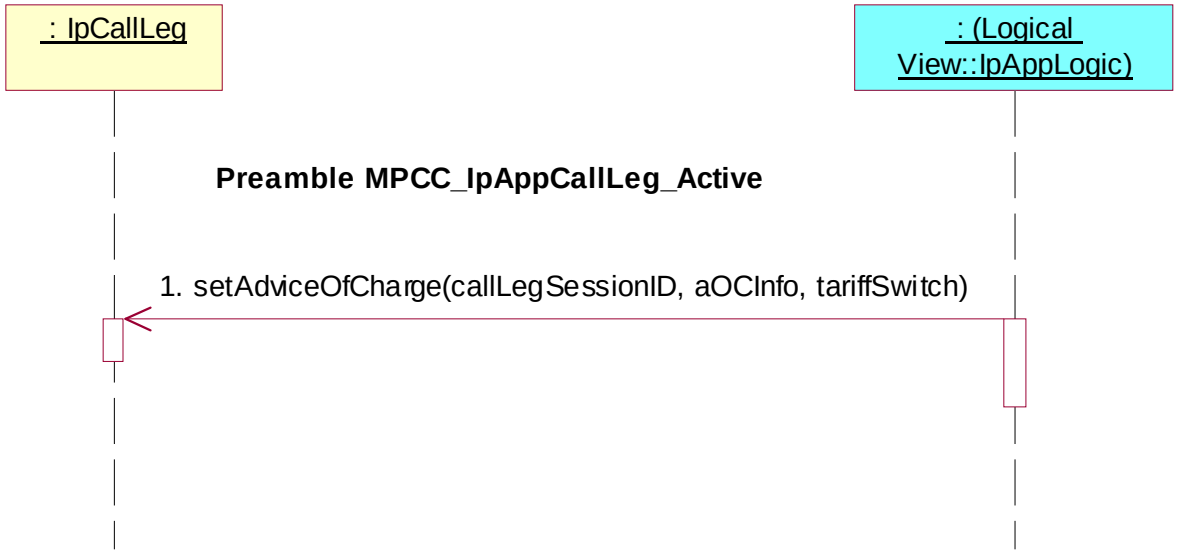
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **setAdviceOfCharge()**

Preamble: **MPCC_IpAppCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **setAdviceOfCharge()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID, aOCInfo, tariffSwitch



Test MPCC_IpAppCallLeg_43

Summary: supervise call leg,

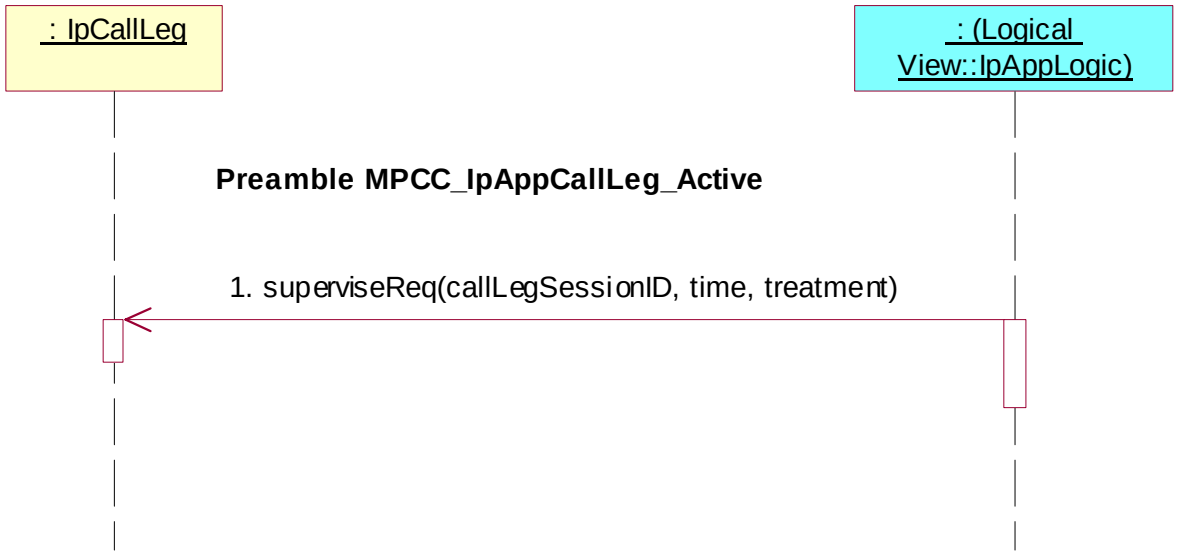
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **superviseReq()**

Preamble: **MPCC_IpAppCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID, time, treatment



Test MPCC_IpAppCallLeg_44

Summary: supervise call leg, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.1

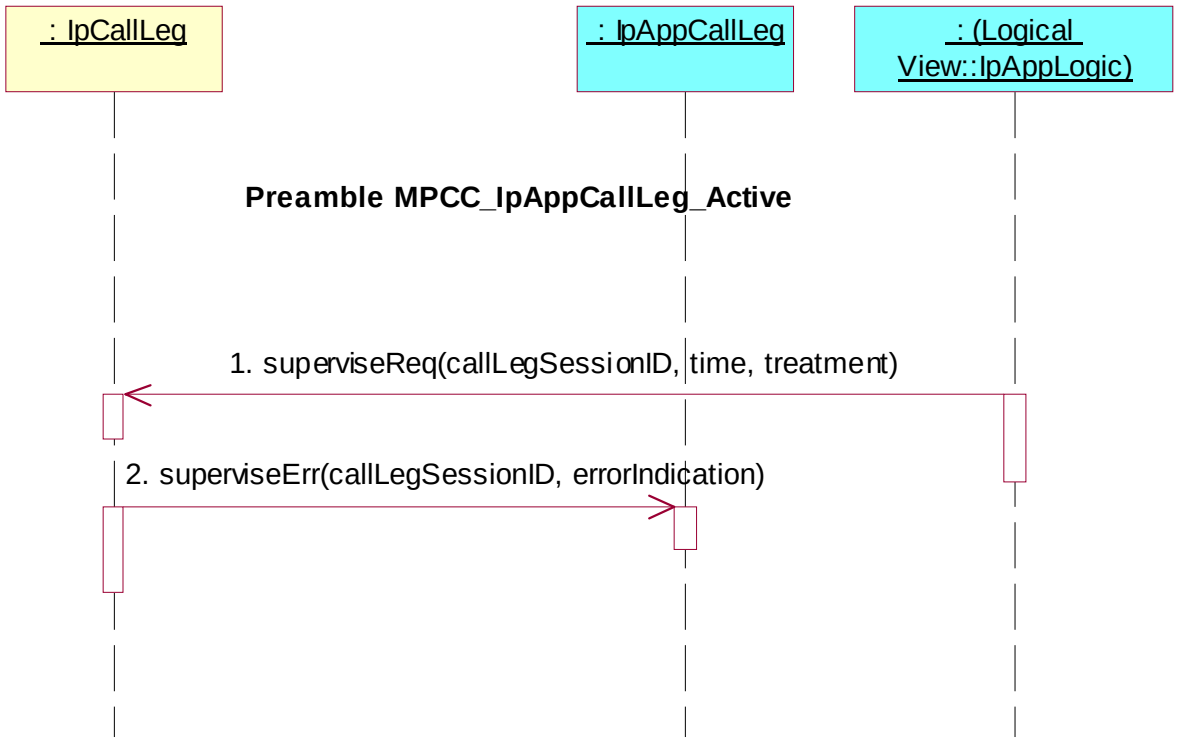
Precondition: IUT capable of invoking **superviseReq()**

Preamble: **MPCC_IpAppCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID, time, treatment

2. Method call **superviseErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



7.2.2.3.1.4 Releasing state

Preamble MPCC_IpAppCallLeg_Releasing

Reference: ES 202 915-4-3 [3], clause 7.3.1.4

Pre-preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiPartyCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiPartyCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Preamble Sequence:

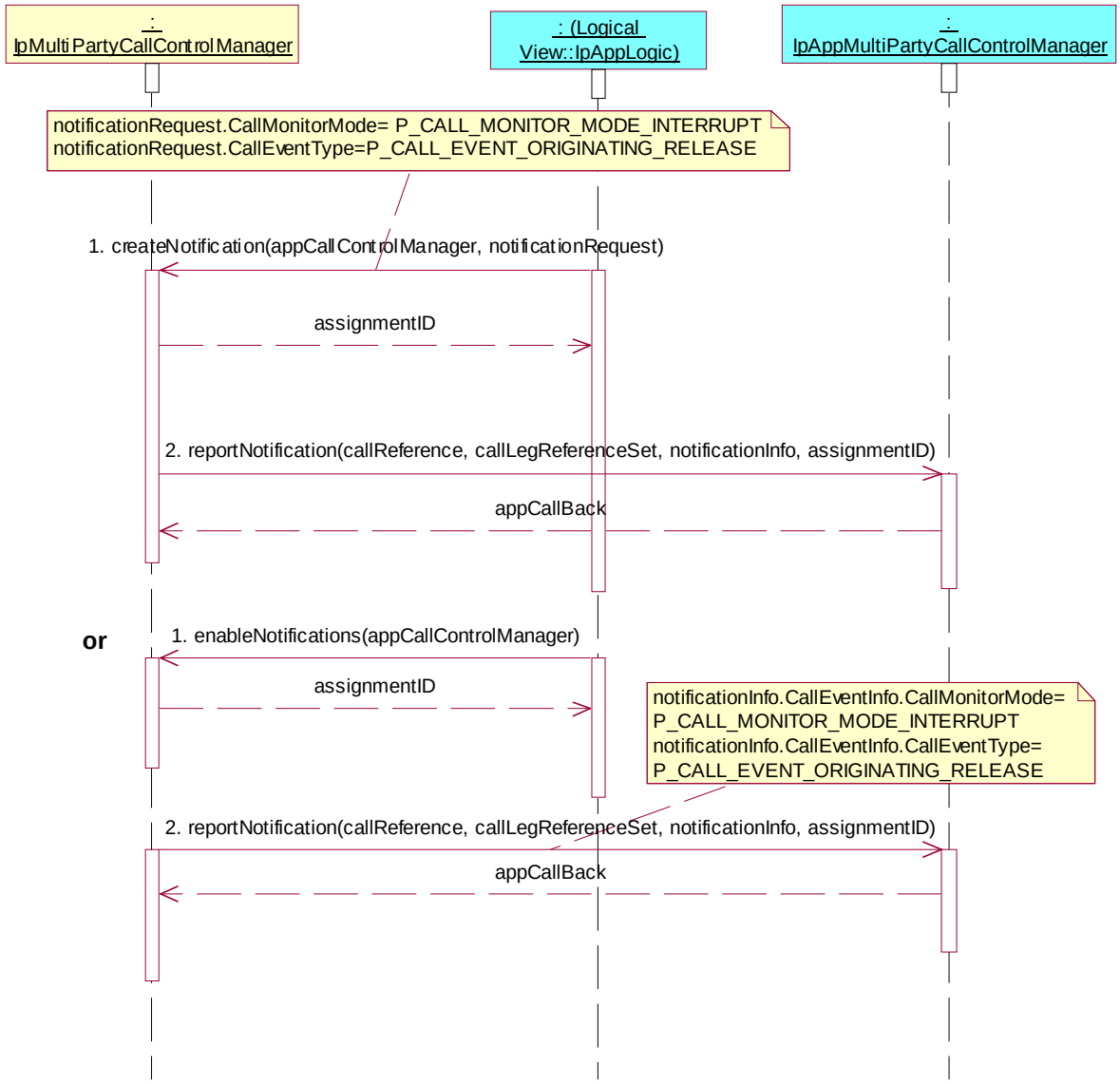
1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest
   notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   notificationRequest.CallEventType= P_CALL_EVENT_ORIGINATING_RELEASE

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   Check: valid value of TpAppMultiPartyCallBack is returned

or

1. Triggered Action: cause IUT to call **enableNotifications()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCallControlManager

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   notificationInfo.CallEventInfo.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   notificationInfo.CallEventInfo.CallEventType= P_CALL_EVENT_ORIGINATING_RELEASE
   Check: valid value of TpAppMultiPartyCallBack is returned



Test MPCC_IpAppCallLeg_45

Summary: request reference of call related to call leg

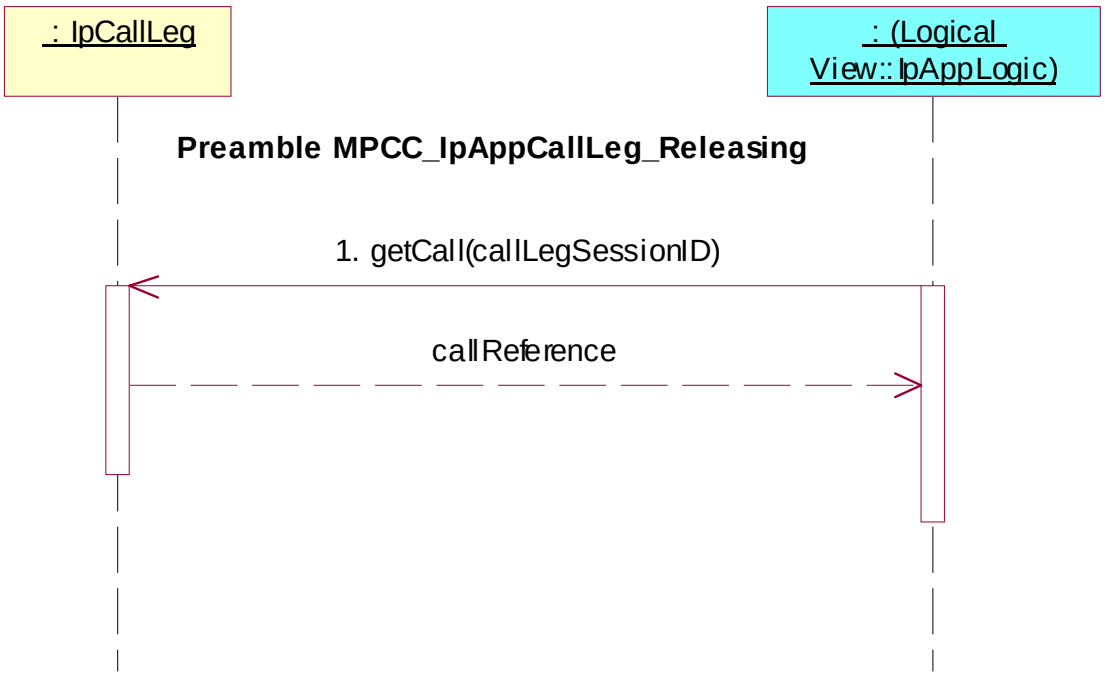
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **getCall()**

Preamble: **MPCC_IpAppCallLeg_Releasing**

Test Sequence:

1. Triggered Action: cause IUT to call **getCall()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID



Test MPCC_IpAppCallLeg_46

Summary: continue processing of call leg

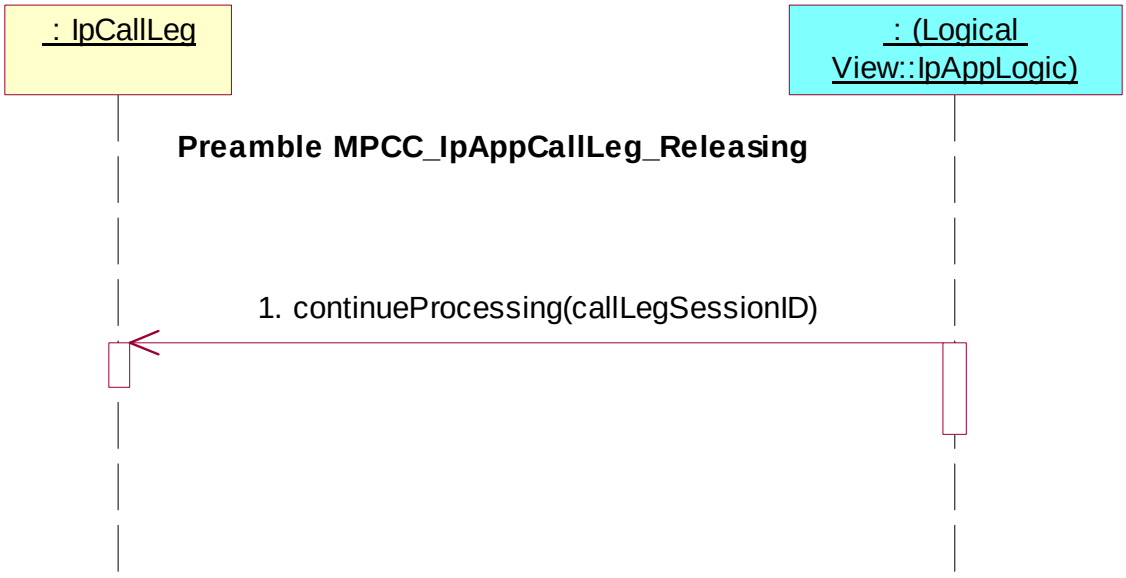
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **continueProcessing()**

Preamble: **MPCC_IpAppCallLeg_Releasing**

Test Sequence:

1. Triggered Action: cause IUT to call **continueProcessing()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID



Test MPCC_IpAppCallLeg_47

Summary: de-assign call leg

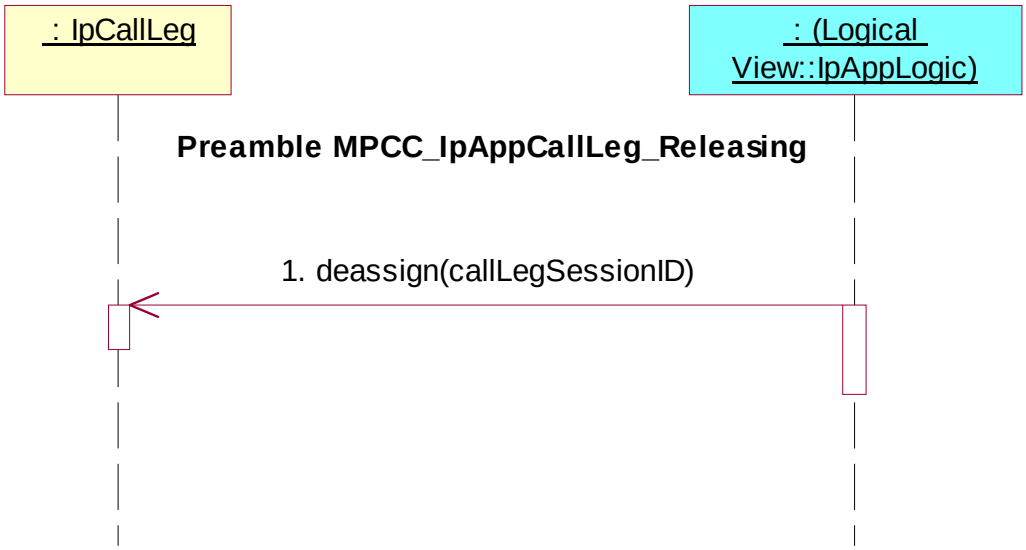
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **deassign()**

Preamble: **MPCC_IpAppCallLeg_Releasing**

Test Sequence:

1. Triggered Action: cause IUT to call **deassign()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID



7.2.2.3.2 Terminating Leg

Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **createNotification()** and **createCallLeg()**

or IUT capable of invoking **createCall()**

7.2.2.3.2.1 Idle state

Preamble MPCC_IpAppCallLeg_Idle

Reference: ES 202 915-4-3 [3], clause 7.3.2.1

Pre-preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiPartyCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiPartyCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Preamble Sequence:

1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest
   notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   Check: valid value of TpAppMultiPartyCallBack is returned

3. Triggered Action: cause IUT to call **createCallLeg()** method on the tester's (SCF's) IpMultiPartyCall interface.
   Parameters: callSessionID, appCallLeg

or

1. Triggered Action: cause IUT to call **createCall()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCall

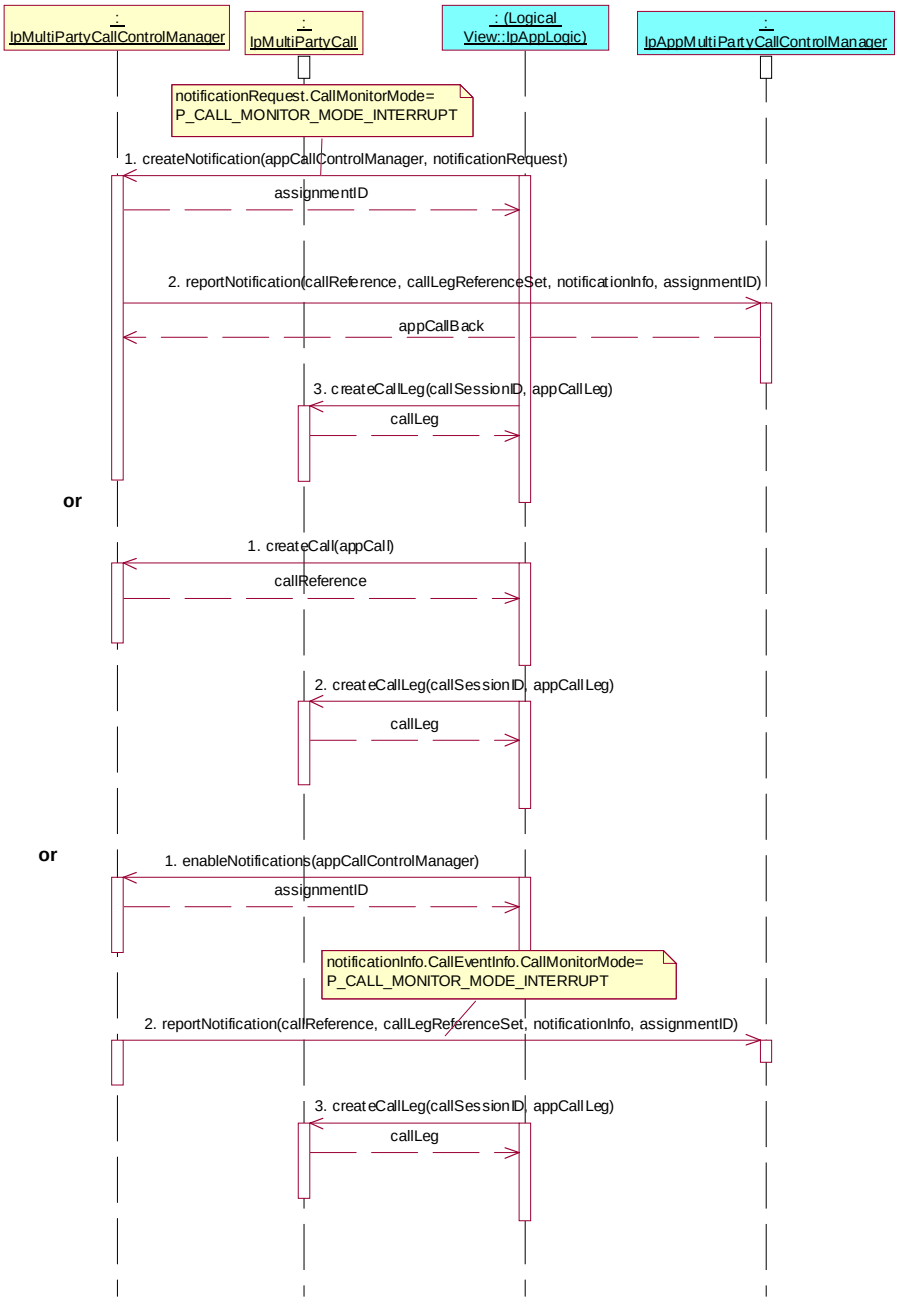
2. Triggered Action: cause IUT to call **createCallLeg()** method on the tester's (SCF's) IpMultiPartyCall interface.
   Parameters: callSessionID, appCallLeg

or

1. Triggered Action: cause IUT to call **enableNotifications()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCallControlManager

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   notificationInfo.CallEventInfo.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   Check: valid value of TpAppMultiPartyCallBack is returned

3. Triggered Action: cause IUT to call **createCallLeg()** method on the tester's (SCF's) IpMultiPartyCall interface.
   Parameters: callSessionID, appCallLeg


: IpMultiPartyCallControlManager
: IpMultiPartyCall
: (Logical View::IpAppLogic)
: IpAppMultiPartyCallControlManager
notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
1. createNotification(appCallControlManager, notificationRequest)
assignmentID
2. reportNotification(callReference, callLegReferenceSet, notificationInfo, assignmentID)
appCallBack
3. createCallLeg(callSessionID, appCallLeg)
callLeg
or
1. createCall(appCall)
callReference
2. createCallLeg(callSessionID, appCallLeg)
callLeg
or
1. enableNotifications(appCallControlManager)
assignmentID
notificationInfo.CallEventInfo.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
2. reportNotification(callReference, callLegReferenceSet, notificationInfo, assignmentID)
3. createCallLeg(callSessionID, appCallLeg)
callLeg

Test MPCC_IpAppCallLeg_48

Summary: route call leg, unsuccessful

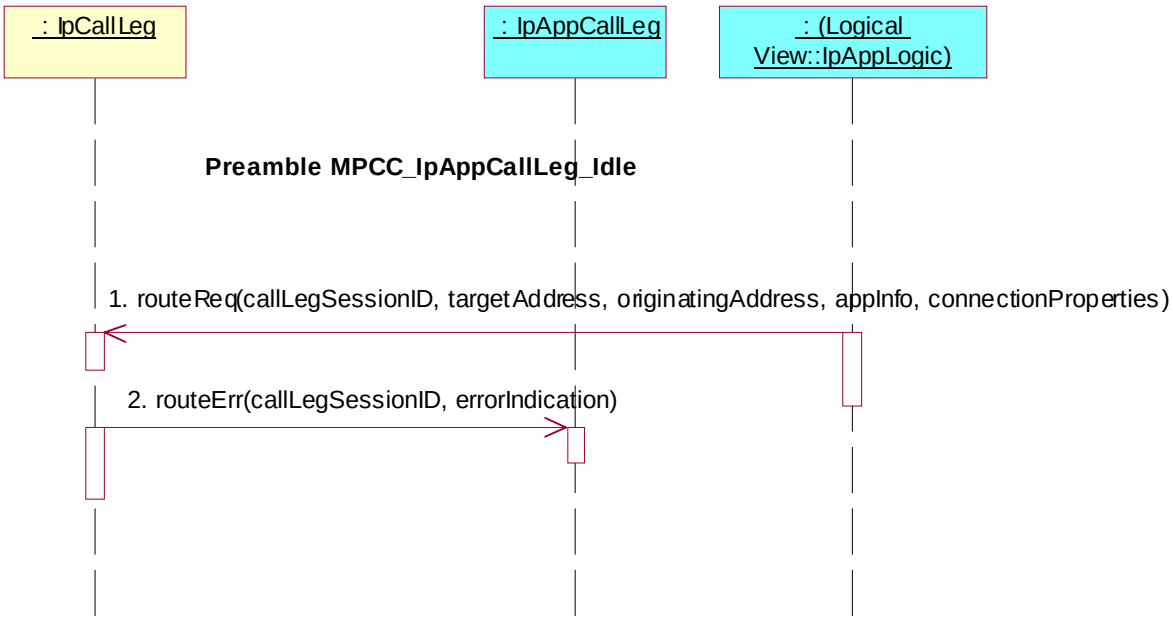
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **routeReq()**

Preamble: **MPCC_IpAppCallLeg_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **routeReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties
2. Method call **routeErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MPCC_IpAppCallLeg_49

Summary: request reference of call related to call leg

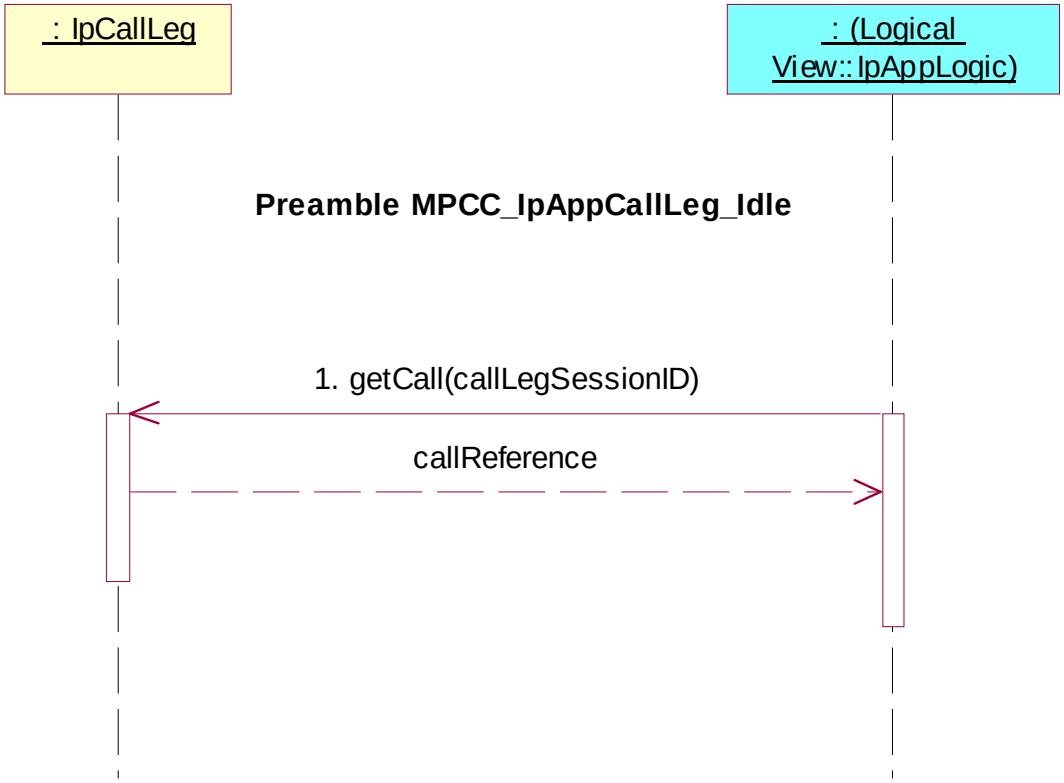
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **getCall()**

Preamble: **MPCC_IpAppCallLeg_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **getCall()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID



Test MPCC_IpAppCallLeg_50

Summary: release call leg

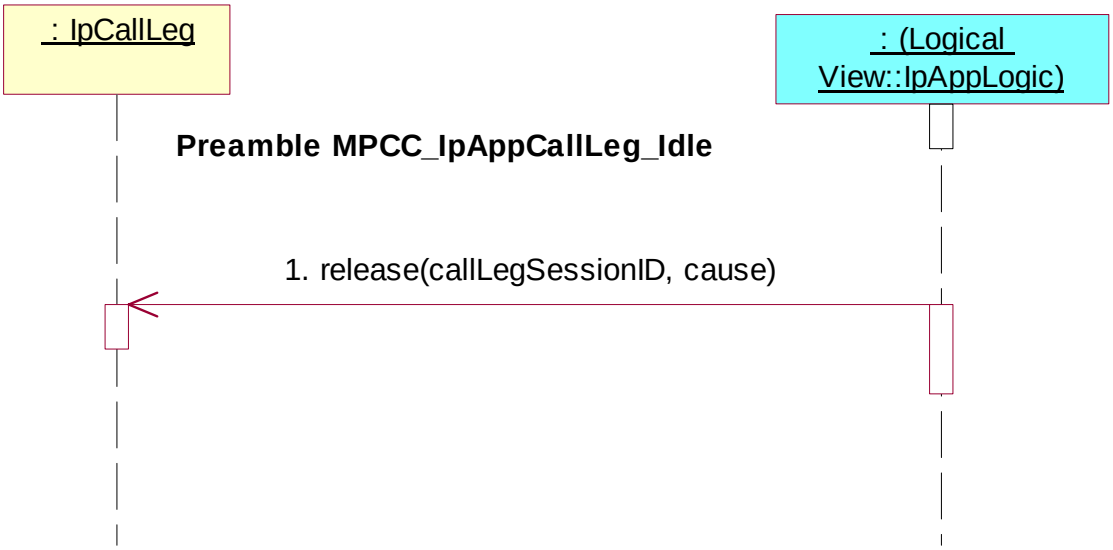
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **release()**

Preamble: **MPCC_IpAppCallLeg_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, cause



Test MPCC_IpAppCallLeg_51

Summary: change or clear event criteria

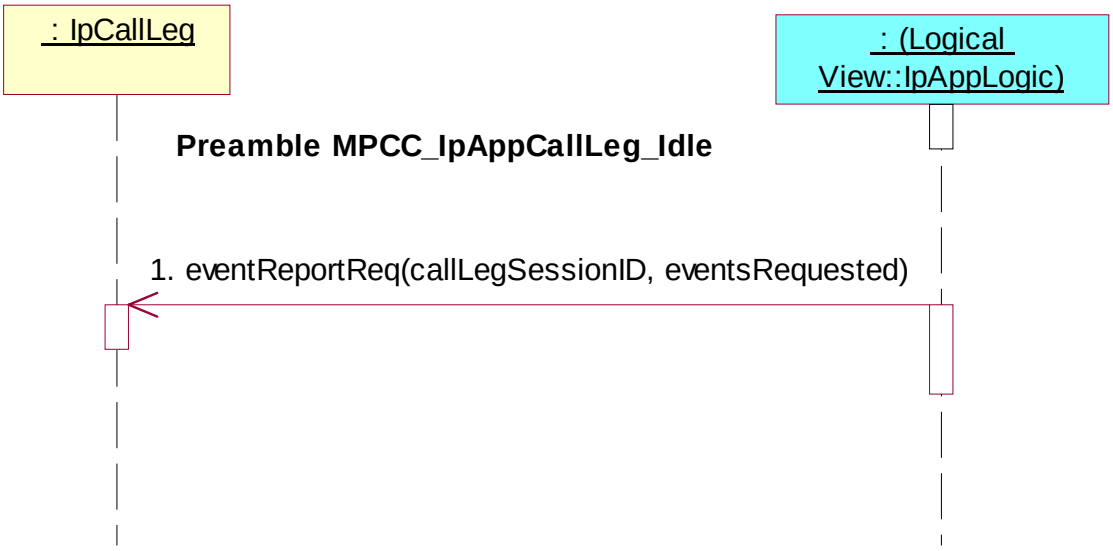
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **eventReportReq()**

Preamble: **MPCC_IpAppCallLeg_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, eventsRequested



Test MPCC_IpAppCallLeg_52

Summary: change or clear event criteria, successful

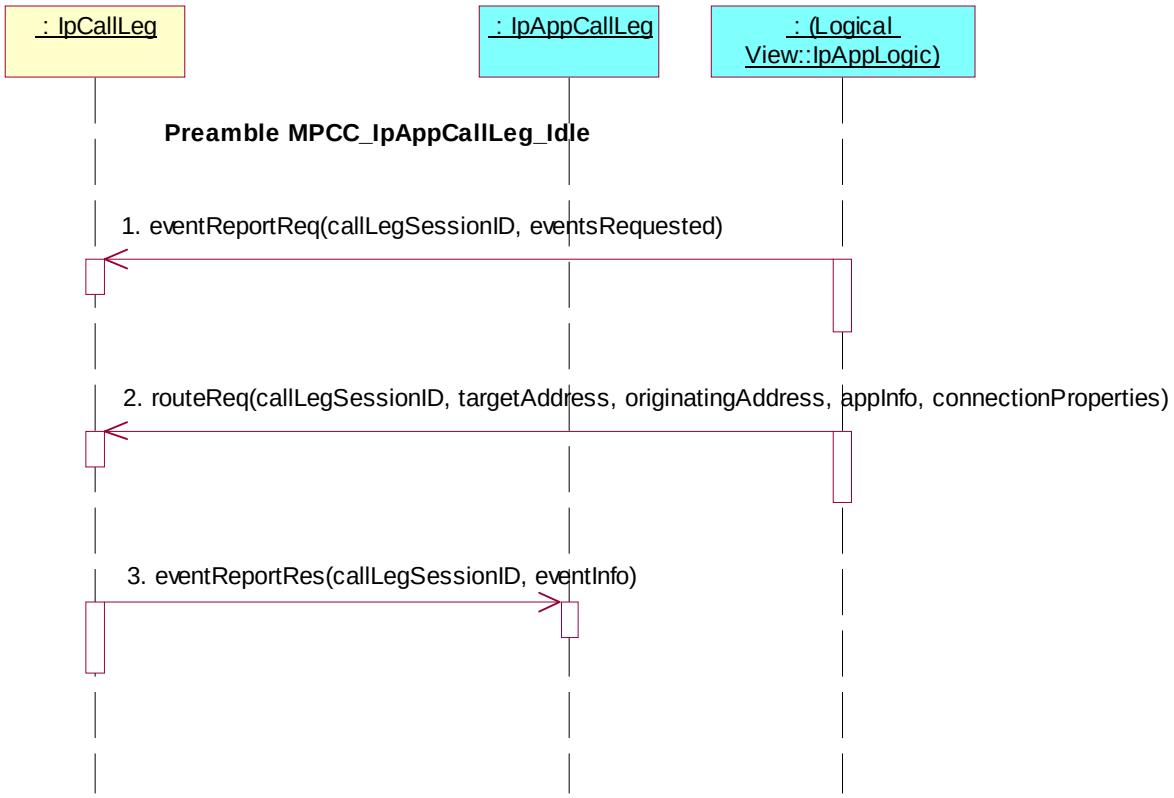
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **eventReportReq(), routeReq()**

Preamble: **MPCC_IpAppCallLeg_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, eventsRequested
2. Triggered Action: cause IUT to call **routeReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties
3. Method call **eventReportRes()**
   Parameters: callLegSessionID, eventInfo
   Check: no exception is returned



Test MPCC_IpAppCallLeg_53

Summary: change or clear event criteria, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.2

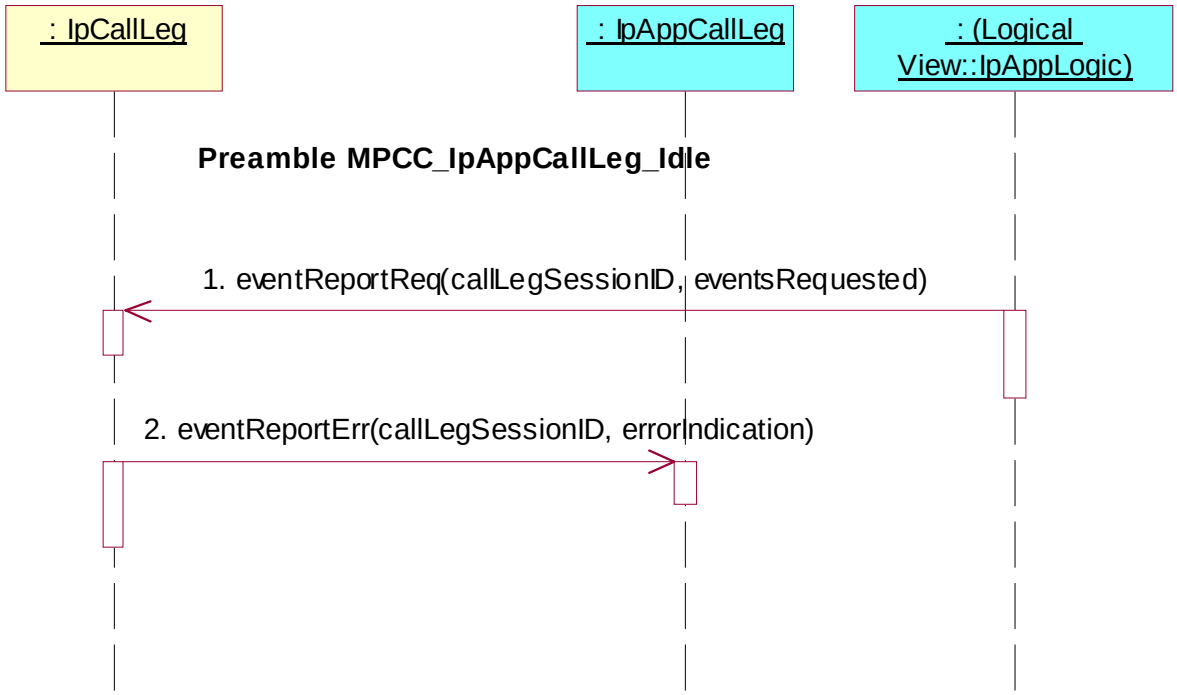
Precondition: IUT capable of invoking **eventReportReq()**

Preamble: **MPCC_IpAppCallLeg_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, eventsRequested

2. Method call **eventReportErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MPCC_IpAppCallLeg_54

Summary: get information about call leg

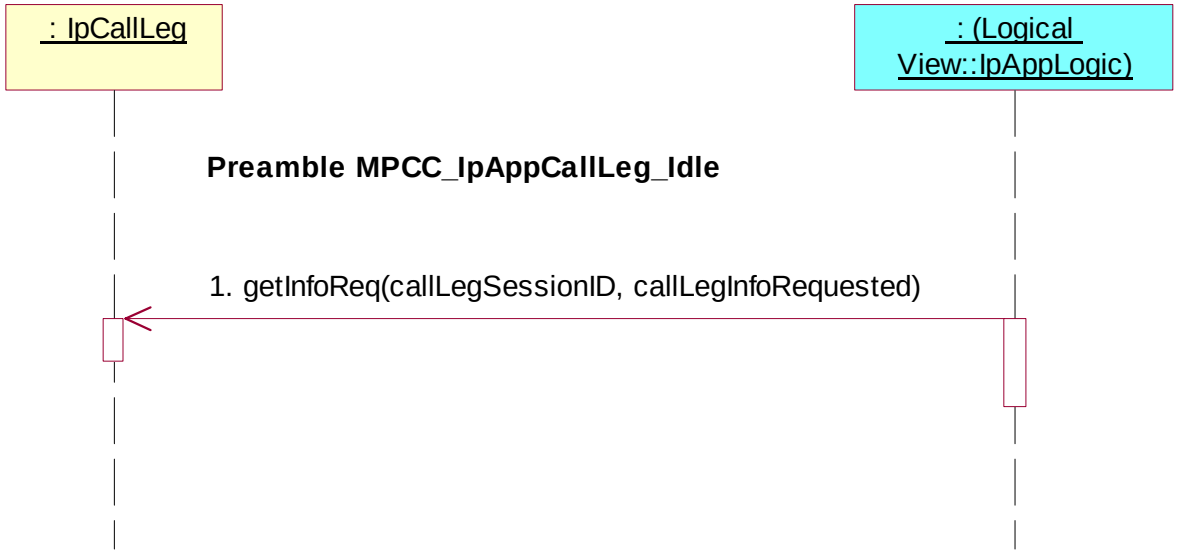
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **MPCC_IpAppCallLeg_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, callLegInfoRequested



Test MPCC_IpAppCallLeg_55

Summary: get information about call leg, unsuccessful

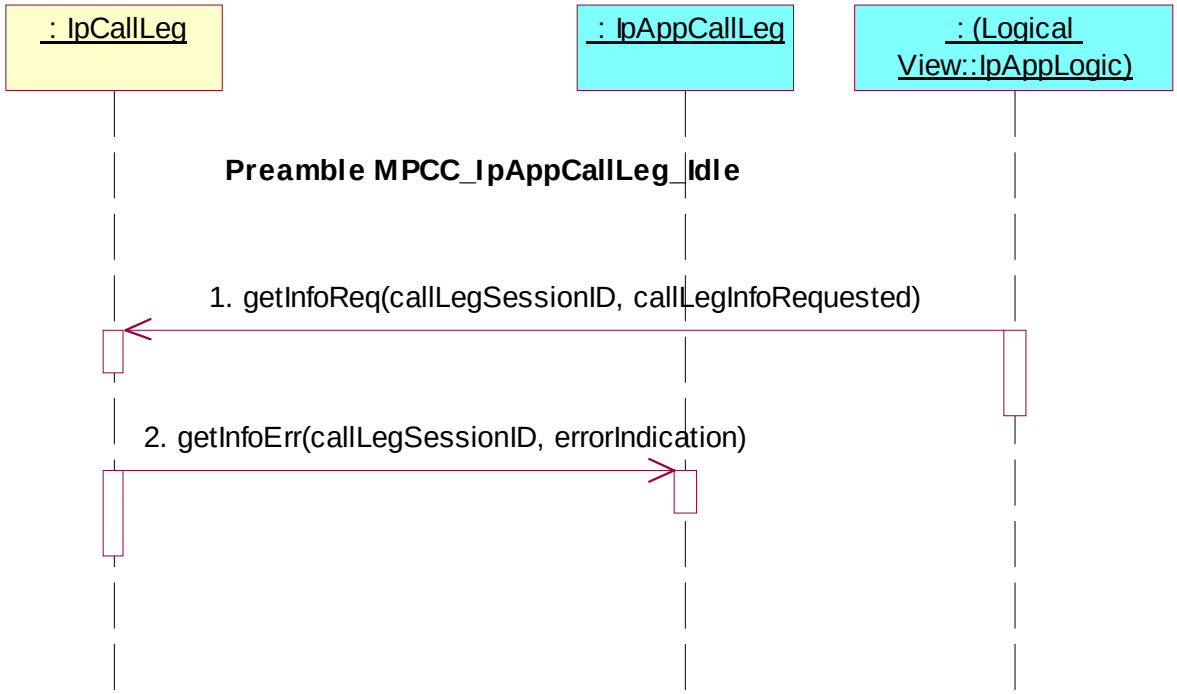
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **MPCC_IpAppCallLeg_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, callLegInfoRequested
2. Method call **getInfoErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MPCC_IpAppCallLeg_56

Summary: set charge plan for call leg

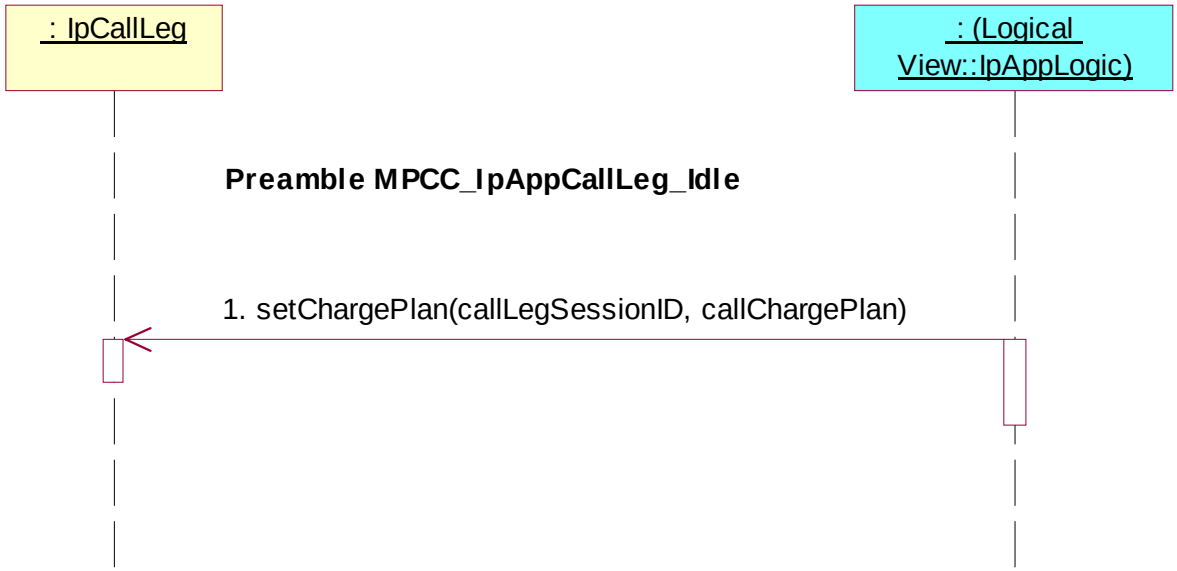
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **setChargePlan()**

Preamble: **MPCC_IpAppCallLeg_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **setChargePlan()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, callChargePlan



Test MPCC_IpAppCallLeg_57

Summary: allow advice of charge information

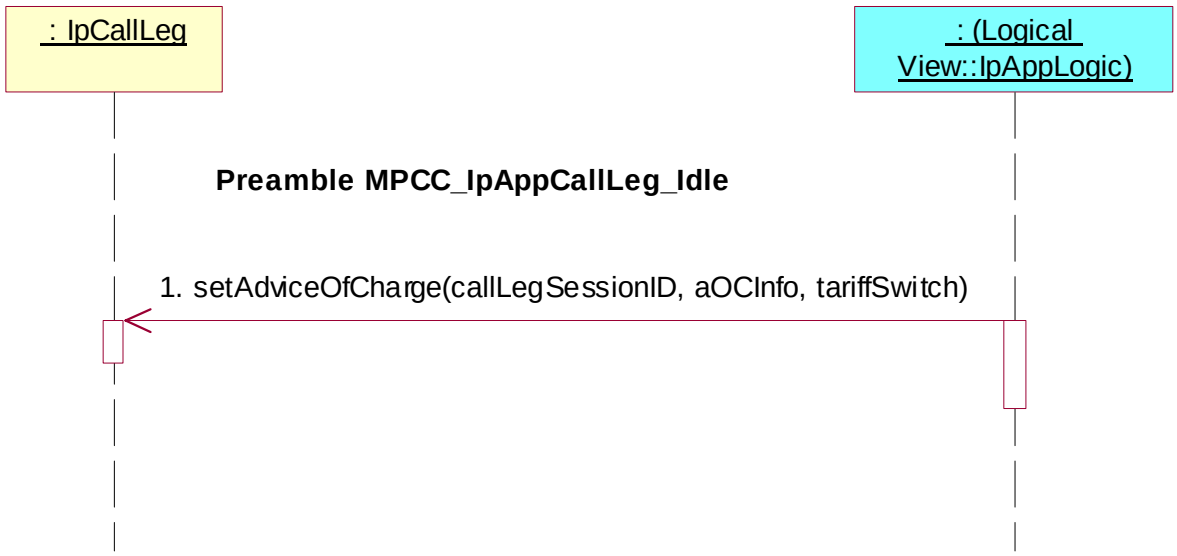
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **setAdviceOfCharge()**

Preamble: **MPCC_IpAppCallLeg_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **setAdviceOfCharge()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, aOCInfo, tariffSwitch



Test MPCC_IpAppCallLeg_58

Summary: supervise call leg

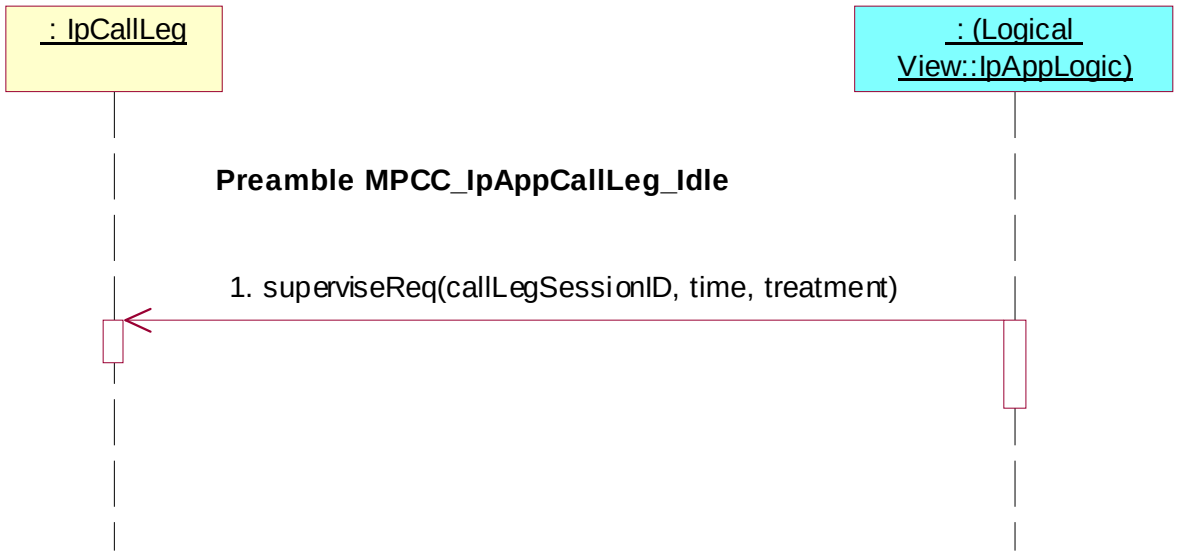
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **superviseReq()**

Preamble: **MPCC_IpAppCallLeg_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, time, treatment



Test MPCC_IpAppCallLeg_59

Summary: supervise call leg, unsuccessful

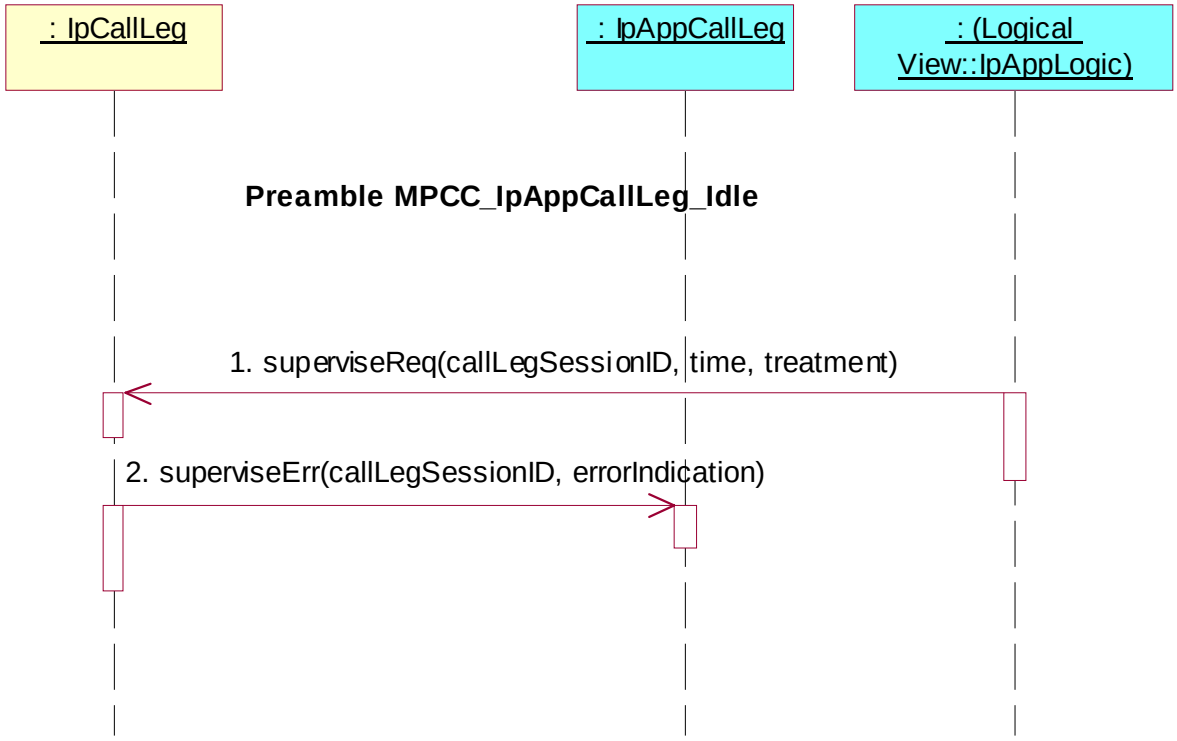
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **superviseReq()**

Preamble: **MPCC_IpAppCallLeg_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, time, treatment
2. Method call **superviseErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



7.2.2.3.2.2 Active (Terminating) state

Precondition: IUT capable of invoking **eventReportReq()** and **routeReq()**

Preamble MPCC_IpAppCallLeg_Active_Terminating

Reference: ES 202 915-4-3 [3], clause 7.3.2.2

Pre-preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiPartyCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiPartyCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Preamble Sequence:

1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest
   notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   Check: valid value of TpAppMultiPartyCallBack is returned

3. Triggered Action: cause IUT to call **createCallLeg()** method on the tester's (SCF's) IpMultiPartyCall interface.
   Parameters: callSessionID, appCallLeg

4. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, eventsRequested

5. Triggered Action: cause IUT to call **routeReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties

6. Method call **eventReportRes()**
   Parameters: callLegSessionID, eventInfo
   Check: no exception is returned

or

1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest
   notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   notificationRequest.CallEventType= P_CALL_EVENT_TERMINATING_CALL_ATTEMPT or P_CALL_EVENT_TERMINATING_CALL_ATTEMPT_AUTHORISED or P_CALL_EVENT_ALERTING or P_CALL_EVENT_ANSWER or P_CALL_EVENT_REDIRECTED or P_CALL_EVENT_QUEUED or P_CALL_EVENT_TERMINATING_SERVICE_CODE

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   Check: valid value of TpAppMultiPartyCallBack is returned

or

1. Triggered Action: cause IUT to call **createCall()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCall

2. Triggered Action: cause IUT to call **createCallLeg()** method on the tester's (SCF's) IpMultiPartyCall interface.
   Parameters: callSessionID, appCallLeg

3. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, eventsRequested

4. Triggered Action: cause IUT to call **routeReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties

5. Method call **eventReportRes()**
   Parameters: callLegSessionID, eventInfo
   Check: no exception is returned

or

1. Triggered Action: cause IUT to call **enableNotifications()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCallControlManager

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   notificationInfo.CallEventInfo.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   Check: valid value of TpAppMultiPartyCallBack is returned

3. Triggered Action: cause IUT to call **createCallLeg()** method on the tester's (SCF's) IpMultiPartyCall interface.
   Parameters: callSessionID, appCallLeg

4. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, eventsRequested

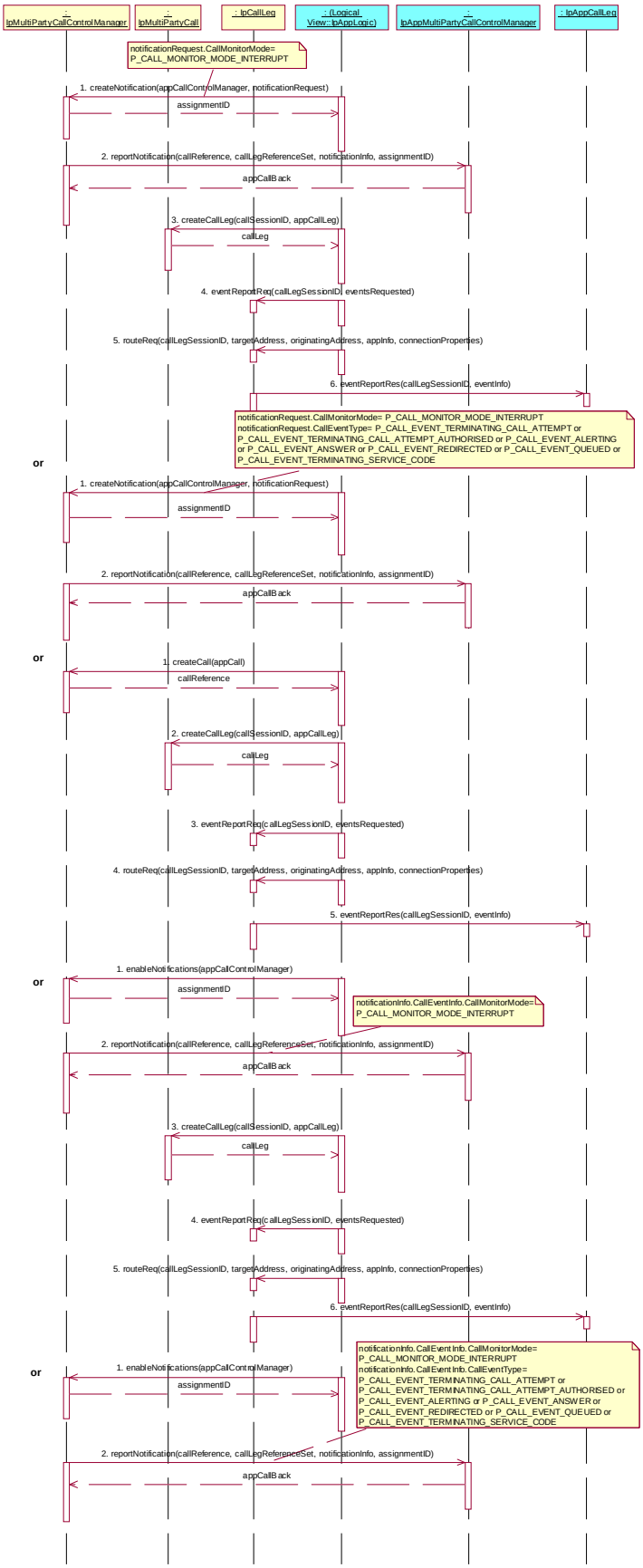
5. Triggered Action: cause IUT to call **routeReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties

6. Method call **eventReportRes()**
   Parameters: callLegSessionID, eventInfo
   Check: no exception is returned

or

1. Triggered Action: cause IUT to call **enableNotifications()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCallControlManager

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   notificationInfo.CallEventInfo.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   notificationInfo.CallEventInfo.CallEventType=
   P_CALL_EVENT_TERMINATING_CALL_ATTEMPT or
   P_CALL_EVENT_TERMINATING_CALL_ATTEMPT_AUTHORISED or
   P_CALL_EVENT_ALERTING or P_CALL_EVENT_ANSWER or
   P_CALL_EVENT_REDIRECTED or P_CALL_EVENT_QUEUED or
   P_CALL_EVENT_TERMINATING_SERVICE_CODE
   Check: valid value of TpAppMultiPartyCallBack is returned


: IpMultiPartyCallControlManager
: IpMultiPartyCall
: IpCallLeg
: (Logical View::IpAppLogic)
: IpAppMultiPartyCallControlManager
: IpAppCallLeg
notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
1. createNotification(appCallControlManager, notificationRequest)
assignmentID
2. reportNotification(callReference, callLegReferenceSet, notificationInfo, assignmentID)
appCallBack
3. createCallLeg(callSessionID, appCallLeg)
callLeg
4. eventReportReq(callLegSessionID, eventsRequested)
5. routeReq(callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties)
6. eventReportRes(callLegSessionID, eventInfo)
notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT notificationRequest.CallEventType= P_CALL_EVENT_TERMINATING_CALL_ATTEMPT or P_CALL_EVENT_TERMINATING_CALL_ATTEMPT_AUTHORISED or P_CALL_EVENT_ALERTING or P_CALL_EVENT_ANSWER or P_CALL_EVENT_REDIRECTED or P_CALL_EVENT_QUEUED or P_CALL_EVENT_TERMINATING_SERVICE_CODE
or
1. createNotification(appCallControlManager, notificationRequest)
assignmentID
2. reportNotification(callReference, callLegReferenceSet, notificationInfo, assignmentID)
appCallBack
or
1. createCall(appCall)
callReference
2. createCallLeg(callSessionID, appCallLeg)
callLeg
3. eventReportReq(callLegSessionID, eventsRequested)
4. routeReq(callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties)
5. eventReportRes(callLegSessionID, eventInfo)
or
1. enableNotifications(appCallControlManager)
assignmentID
notificationInfo.CallEventInfo.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
2. reportNotification(callReference, callLegReferenceSet, notificationInfo, assignmentID)
appCallBack
3. createCallLeg(callSessionID, appCallLeg)
callLeg
4. eventReportReq(callLegSessionID, eventsRequested)
5. routeReq(callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties)
6. eventReportRes(callLegSessionID, eventInfo)
or
1. enableNotifications(appCallControlManager)
assignmentID
notificationInfo.CallEventInfo.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT notificationInfo.CallEventInfo.CallEventType= P_CALL_EVENT_TERMINATING_CALL_ATTEMPT or P_CALL_EVENT_TERMINATING_CALL_ATTEMPT_AUTHORISED or P_CALL_EVENT_ALERTING or P_CALL_EVENT_ANSWER or P_CALL_EVENT_REDIRECTED or P_CALL_EVENT_QUEUED or P_CALL_EVENT_TERMINATING_SERVICE_CODE
2. reportNotification(callReference, callLegReferenceSet, notificationInfo, assignmentID)
appCallBack

Test MPCC_IpAppCallLeg_60

Summary: attach media, successful

Reference: ES 202 915-4-3 [3], clause 7.3.2

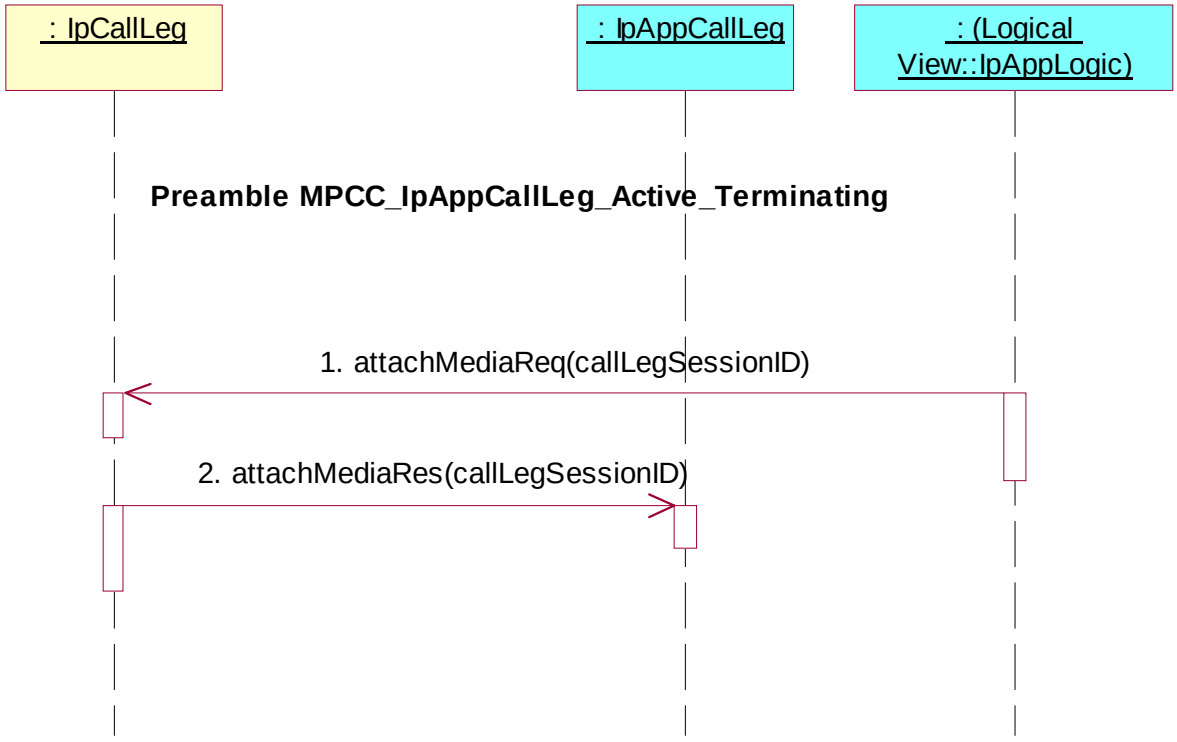
Precondition: IUT capable of invoking **attachMediaReq()**

Preamble: **MPCC_IpAppCallLeg_Active_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **attachMediaReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID

2. Method call **attachMediaRes()**
   Parameters: callLegSessionID
   Check: no exception is returned



Test MPCC_IpAppCallLeg_61

Summary: attach media, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.2

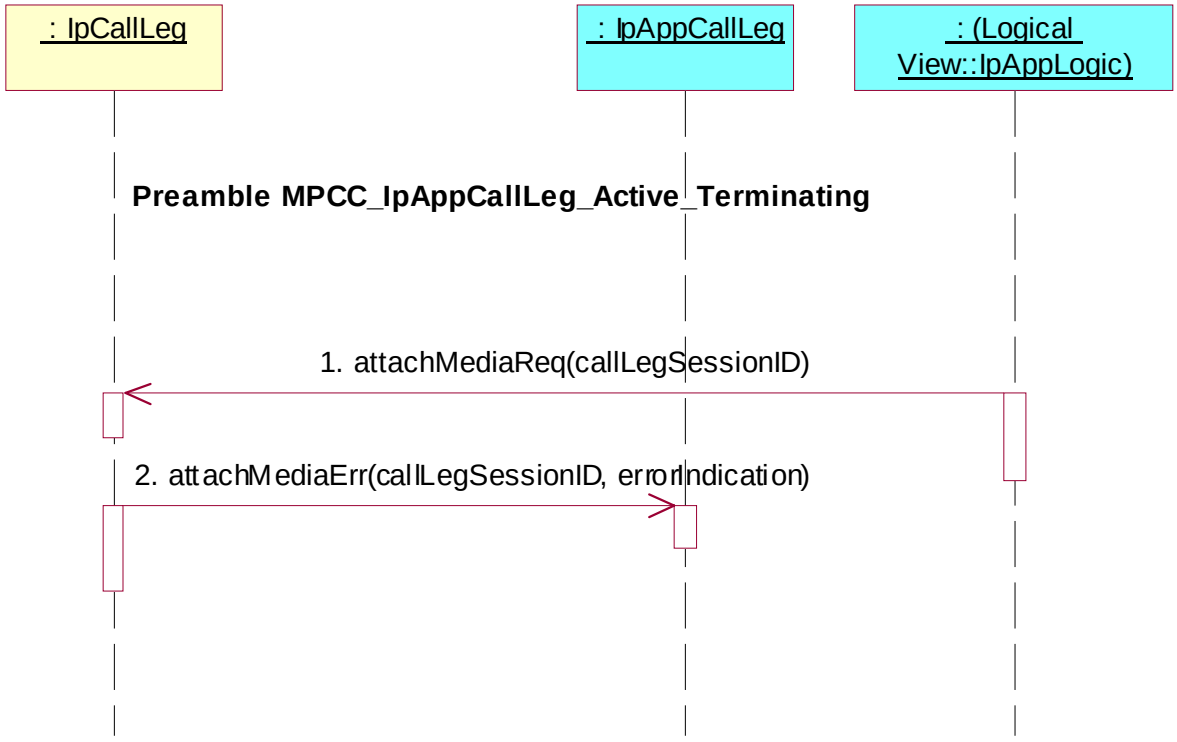
Precondition: IUT capable of invoking **attachMediaReq()**

Preamble: **MPCC_IpAppCallLeg_Active_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **attachMediaReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID

2. Method call **attachMediaErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MPCC_IpAppCallLeg_62

Summary: detach media, successful

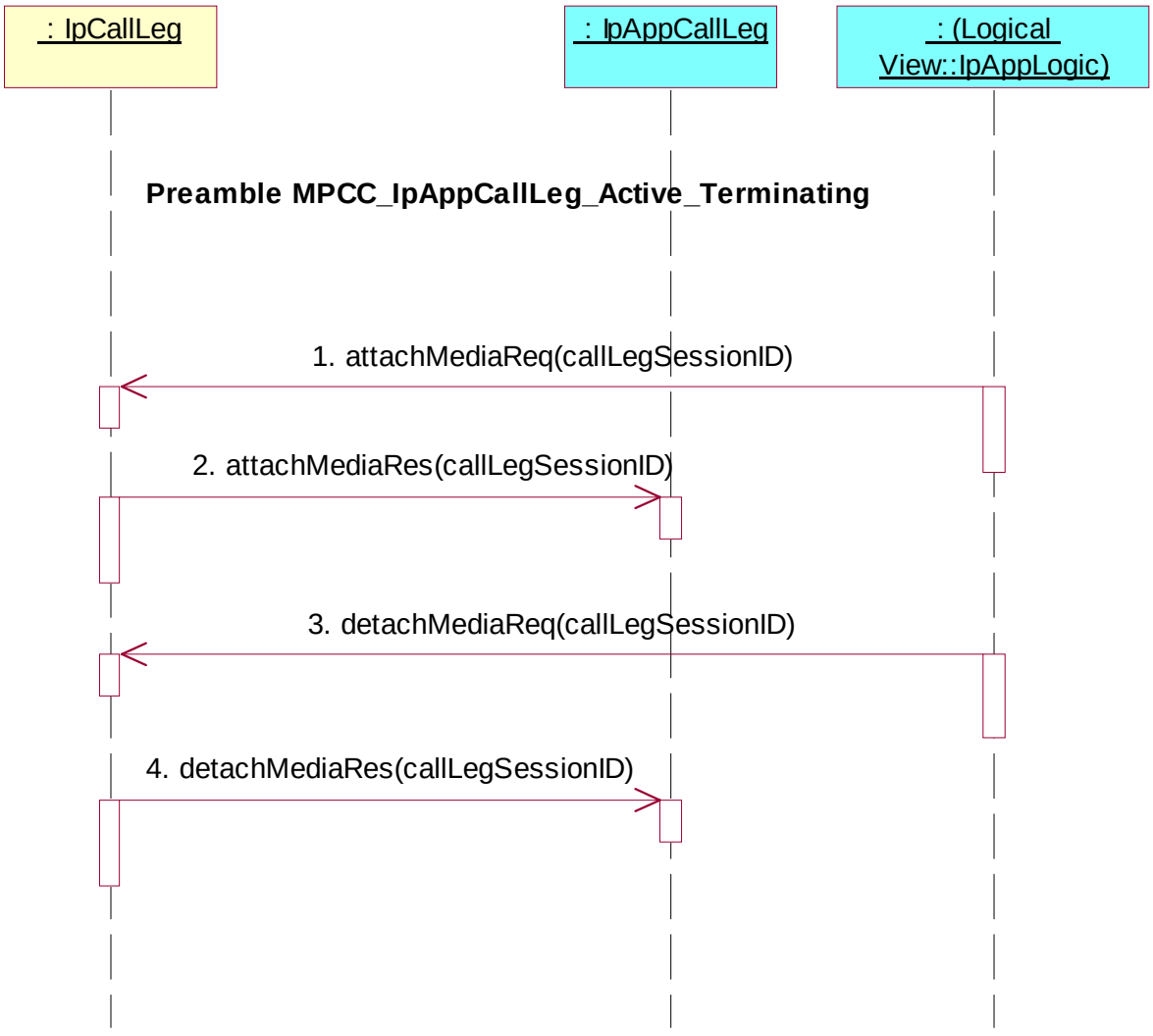
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **detachMediaReq()**

Preamble: **MPCC_IpAppCallLeg_Active_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **attachMediaReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID
2. Method call **attachMediaRes()**
   Parameters: callLegSessionID
   Check: no exception is returned
3. Triggered Action: cause IUT to call **detachMediaReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID
4. Method call **detachMediaRes()**
   Parameters: callLegSessionID
   Check: no exception is returned



Test MPCC_IpAppCallLeg_63

Summary: detach media, unsuccessful

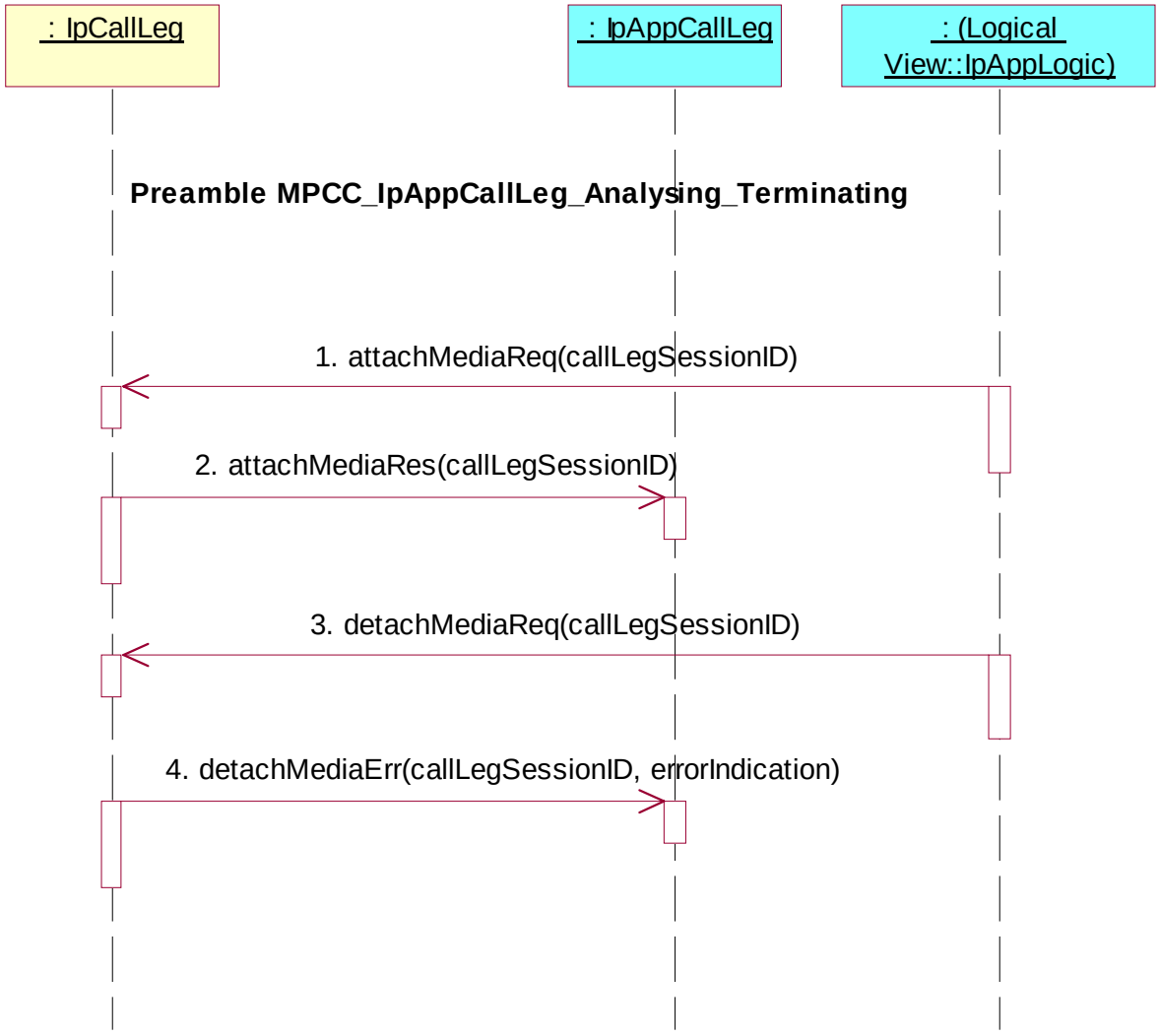
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **detachMediaReq()**

Preamble: **MPCC_IpAppCallLeg_Active_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **attachMediaReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID
2. Method call **attachMediaRes()**
   Parameters: callLegSessionID
   Check: no exception is returned
3. Triggered Action: cause IUT to call **detachMediaReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID
4. Method call **detachMediaErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MPCC_IpAppCallLeg_64

Summary: request reference of call related to call leg

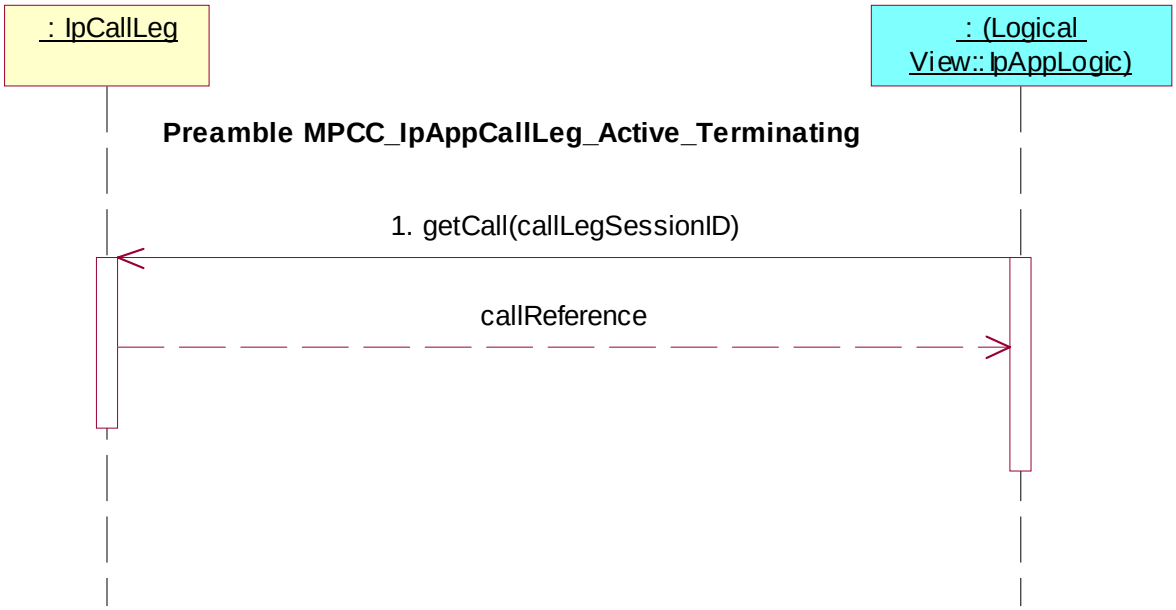
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **getCall()**

Preamble: **MPCC_IpAppCallLeg_Active_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **getCall()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID



Test MPCC_IpAppCallLeg_65

Summary: request reference of call related to call leg

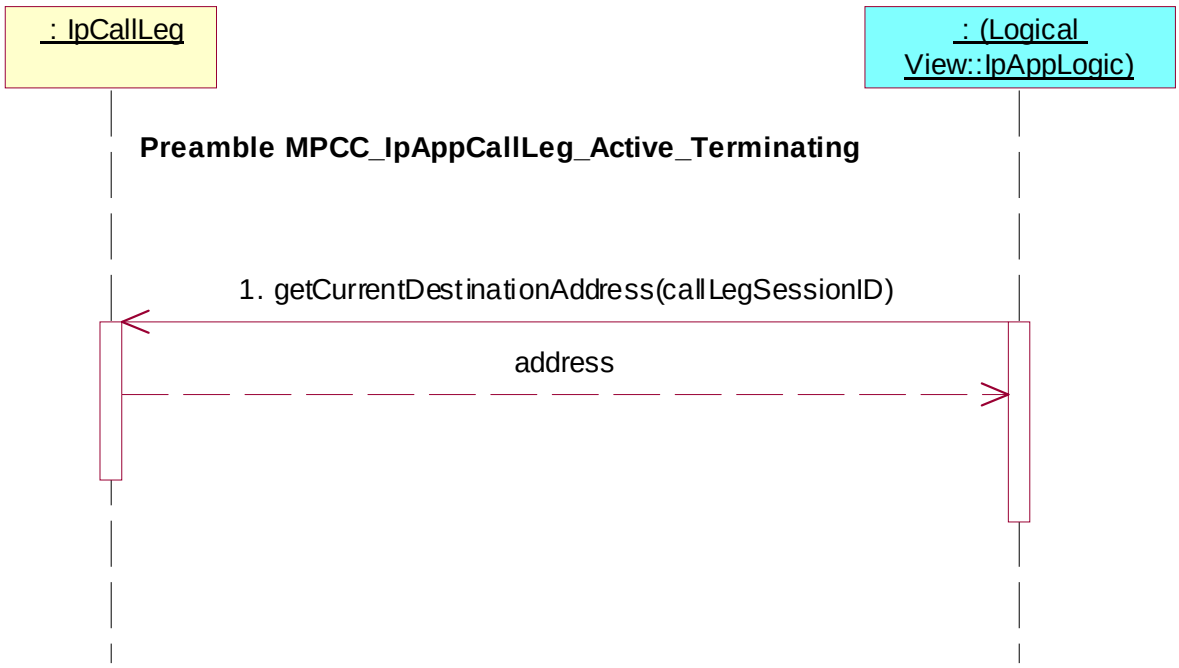
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **getCurrentDestinationAddress()**

Preamble: **MPCC_IpAppCallLeg_Active_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **getCurrentDestinationAddress()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID



Test MPCC_IpAppCallLeg_66

Summary: continue processing of call leg

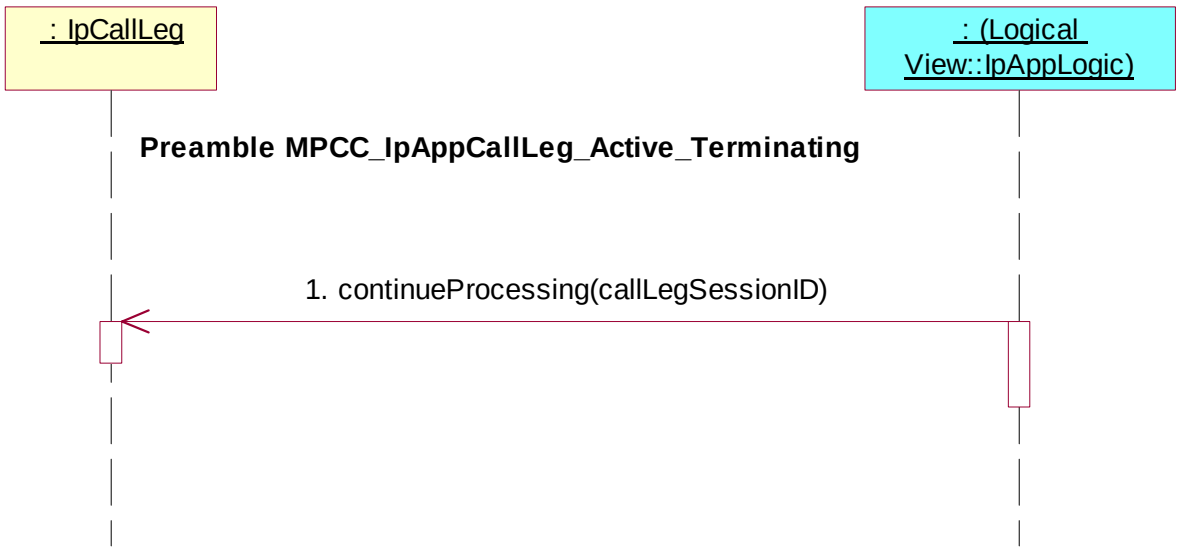
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **continueProcessing()**

Preamble: **MPCC_IpAppCallLeg_Active_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **continueProcessing()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID



Test MPCC_IpAppCallLeg_67

Summary: release call leg

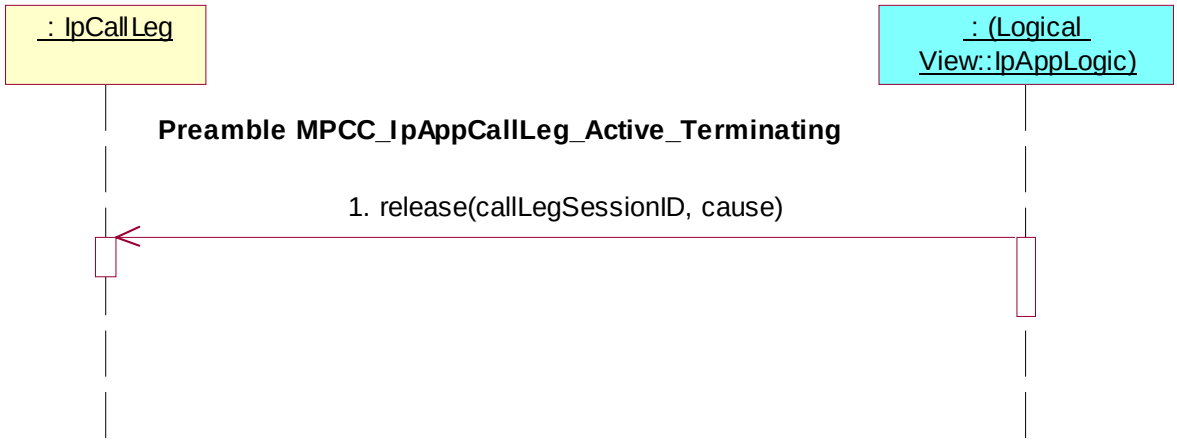
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **release()**

Preamble: **MPCC_IpAppCallLeg_Active_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, cause



Test MPCC_IpAppCallLeg_68

Summary: de-assign call leg

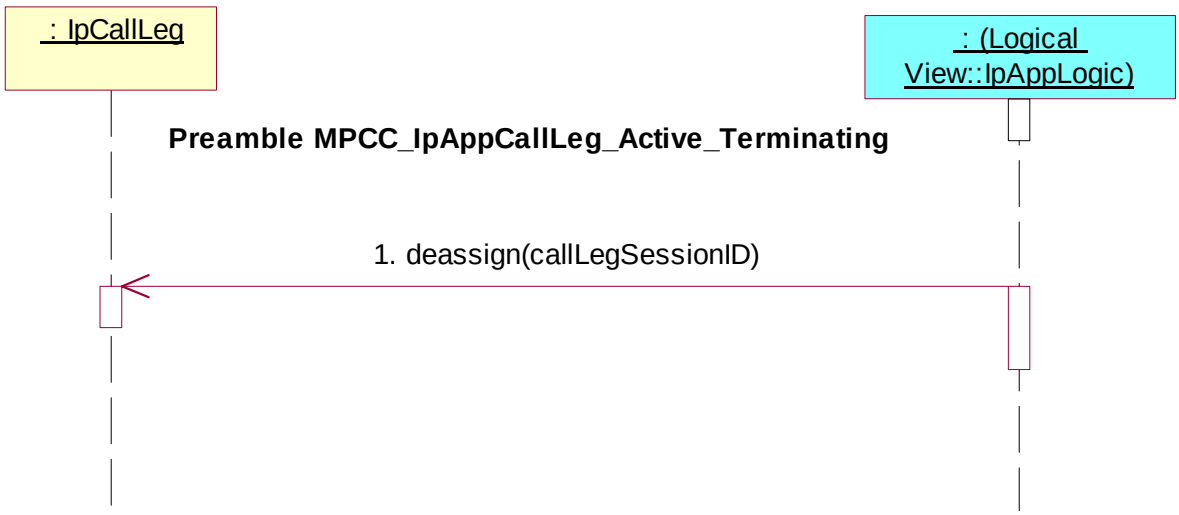
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **deassign()**

Preamble: **MPCC_IpAppCallLeg_Active_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **deassign()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID



Test MPCC_IpAppCallLeg_69

Summary: get information about call leg

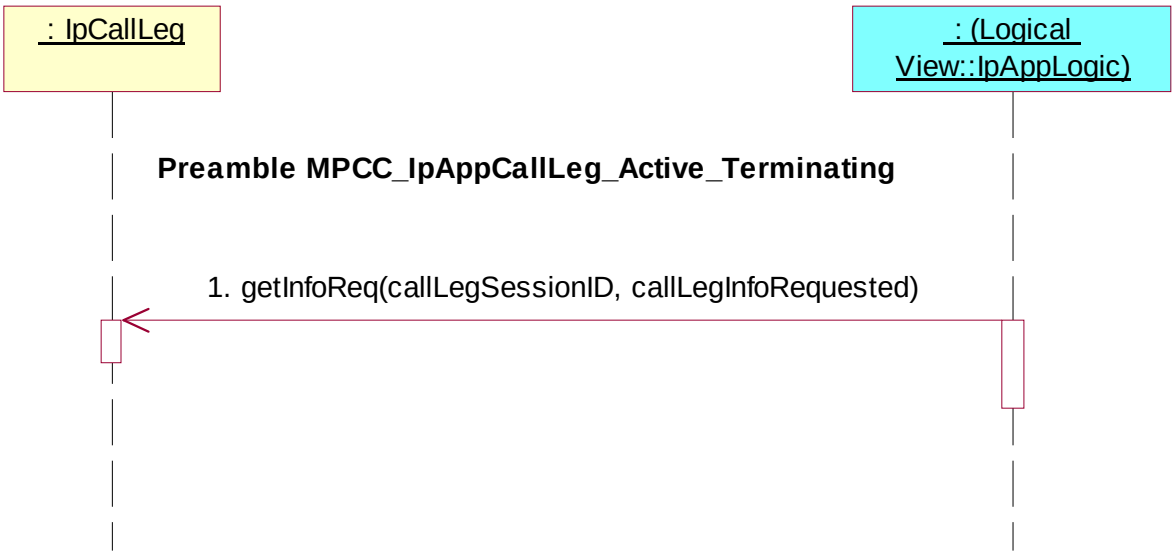
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **MPCC_IpAppCallLeg_Active_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, callLegInfoRequested



Test MPCC_IpAppCallLeg_70

Summary: get information about call leg, unsuccessful

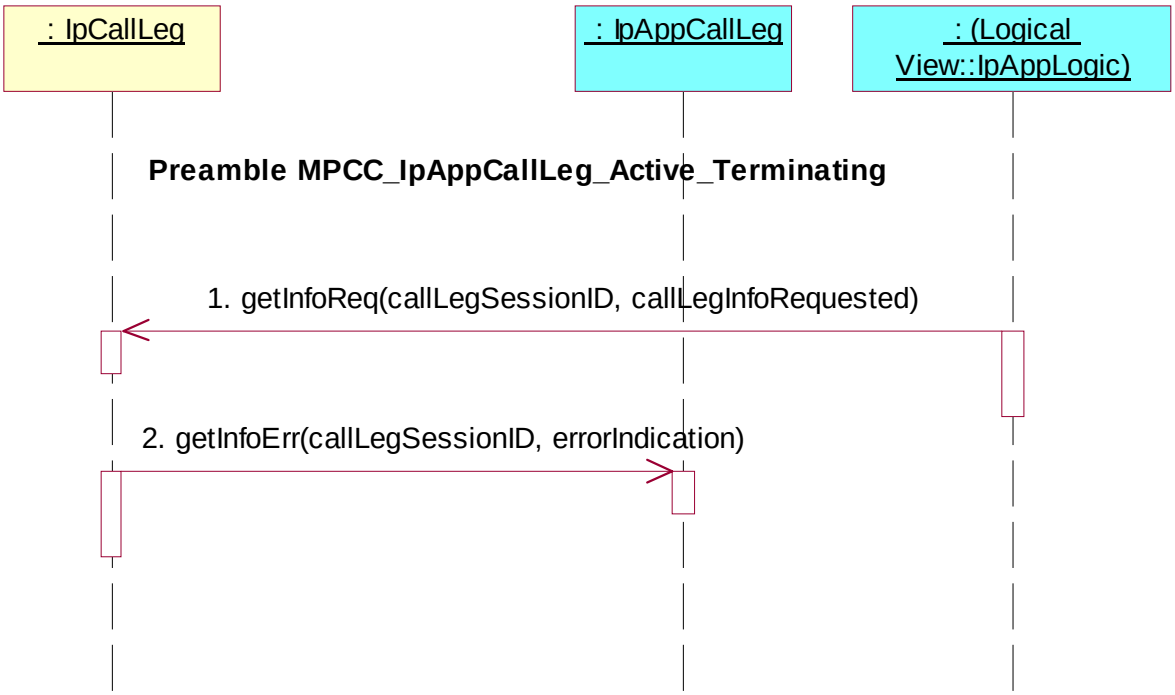
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **MPCC_IpAppCallLeg_Active_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, callLegInfoRequested
2. Method call **getInfoErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MPCC_IpAppCallLeg_71

Summary: supervise call leg

Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **superviseReq()**

Preamble: **MPCC_IpAppCallLeg_Active_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID, time, treatment

Test MPCC_IpAppCallLeg_72

Summary: supervise call leg, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.2

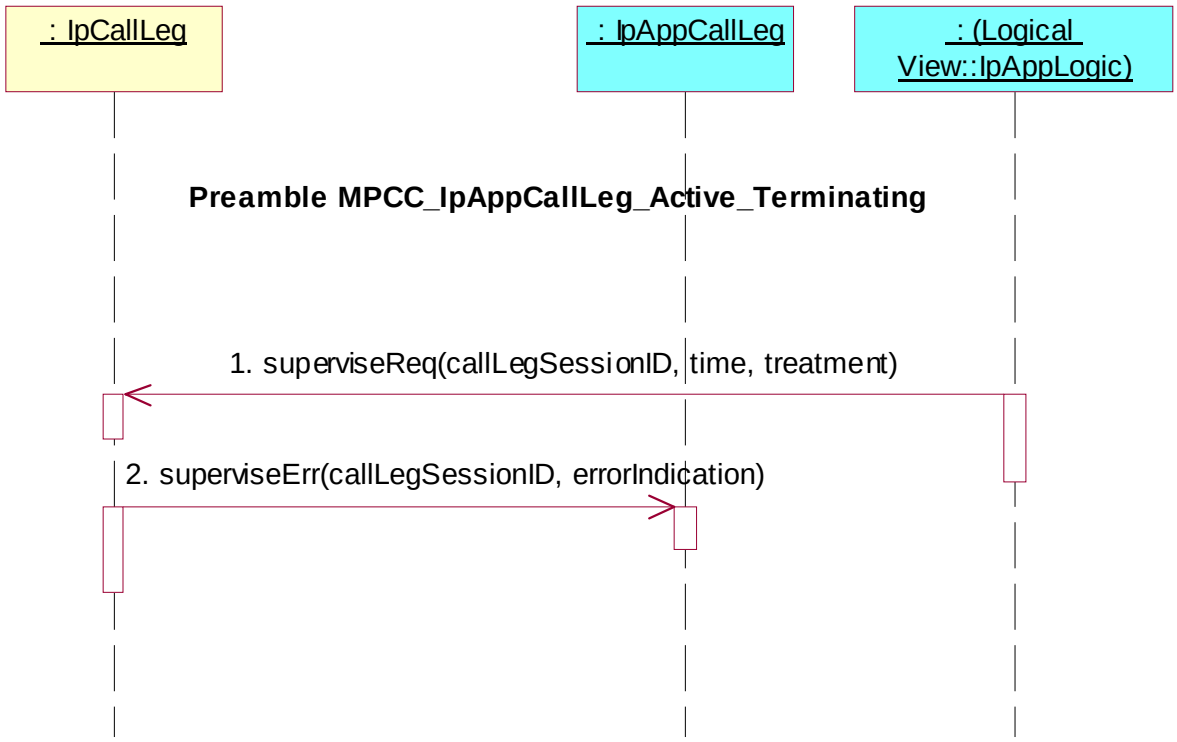
Precondition: IUT capable of invoking **superviseReq()**

Preamble: **MPCC_IpAppCallLeg_Active_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) IpCallLeg interface.
   Parameters: callLegSessionID, time, treatment

2. Method call **superviseErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



7.2.2.3.2.3 Releasing (Terminating) state

Precondition: IUT capable of invoking **eventReportReq()**, **routeReq()** and **release()**

Preamble MPCC_IpAppCallLeg_Releasing_Terminating

Reference: ES 202 915-4-3 [3], clause 7.3.2.3

Pre-preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiPartyCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiPartyCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Preamble Sequence:

1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest
   notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   Check: valid value of TpAppMultiPartyCallBack is returned

3. Triggered Action: cause IUT to call **createCallLeg()** method on the tester's (SCF's) IpMultiPartyCall interface.
   Parameters: callSessionID, appCallLeg

4. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, eventsRequested

5. Triggered Action: cause IUT to call **routeReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties

6. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, cause

or

1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest
   notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   notificationRequest.CallEventType= P_CALL_EVENT_TERMINATING_RELEASE

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   Check: valid value of TpAppMultiPartyCallBack is returned

or

1. Triggered Action: cause IUT to call **createCall()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCall

2. Triggered Action: cause IUT to call **createCallLeg()** method on the tester's (SCF's) IpMultiParty interface.
   Parameters: callSessionID, appCallLeg

3. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, eventsRequested

4. Triggered Action: cause IUT to call **routeReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties

5. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, cause

or

1. Triggered Action: cause IUT to call **enableNotifications()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCallControlManager

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   notificationInfo.CallEventInfo.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   Check: valid value of TpAppMultiPartyCallBack is returned

3. Triggered Action: cause IUT to call **createCallLeg()** method on the tester's (SCF's) IpMultiPartyCall interface.
   Parameters: callSessionID, appCallLeg

4. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, eventsRequested

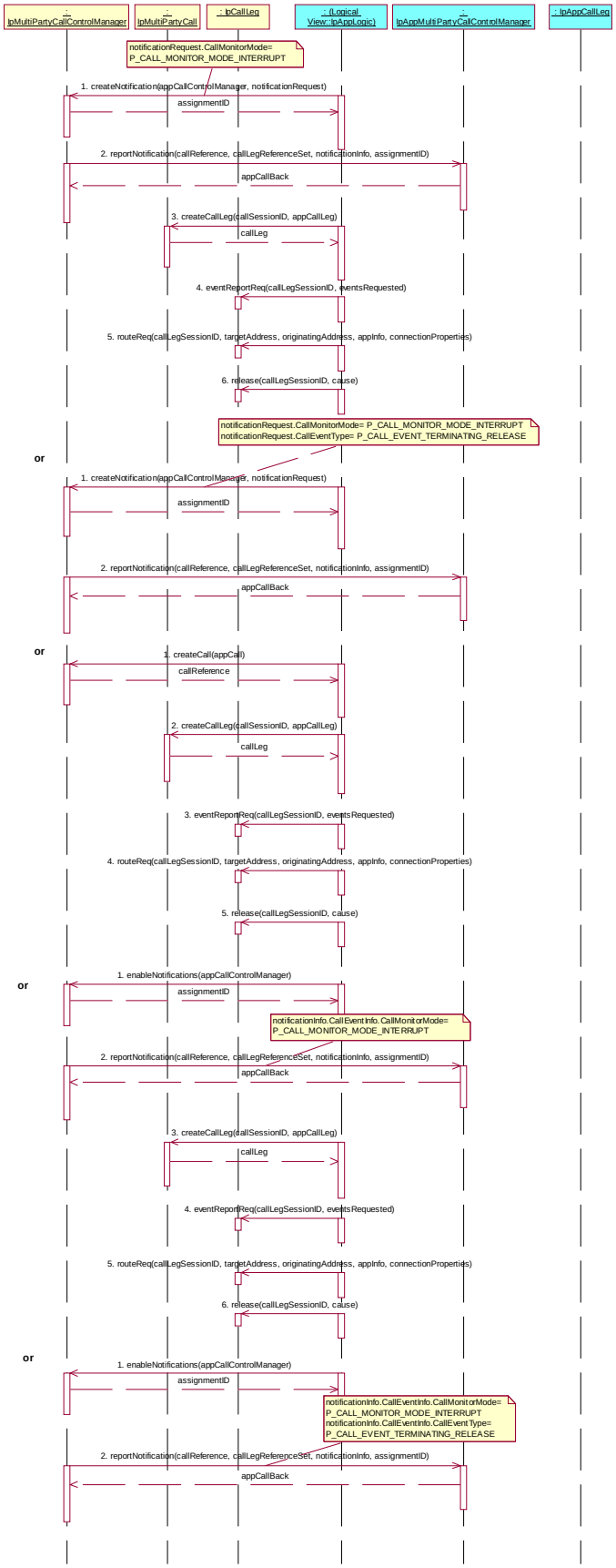
5. Triggered Action: cause IUT to call **routeReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties

6. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, cause

or

1. Triggered Action: cause IUT to call **enableNotifications()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCallControlManager

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   notificationInfo.CallEventInfo.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   notificationInfo.CallEventInfo.CallEventType= P_CALL_EVENT_TERMINATING_RELEASE
   Check: valid value of TpAppMultiPartyCallBack is returned


: IpMultiPartyCallControlManager
: IpMultiPartyCall
: IpCallLeg
: (Logical View::IpAppLogic)
: IpAppMultiPartyCallControlManager
: IpAppCallLeg
notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
1. createNotification(appCallControlManager, notificationRequest)
assignmentID
2. reportNotification(callReference, callLegReferenceSet, notificationInfo, assignmentID)
appCallBack
3. createCallLeg(callSessionID, appCallLeg)
callLeg
4. eventReportReq(callLegSessionID, eventsRequested)
5. routeReq(callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties)
6. release(callLegSessionID, cause)
notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
notificationRequest.CallEventType= P_CALL_EVENT_TERMINATING_RELEASE
or
1. createNotification(appCallControlManager, notificationRequest)
assignmentID
2. reportNotification(callReference, callLegReferenceSet, notificationInfo, assignmentID)
appCallBack
or
1. createCall(appCall)
callReference
2. createCallLeg(callSessionID, appCallLeg)
callLeg
3. eventReportReq(callLegSessionID, eventsRequested)
4. routeReq(callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties)
5. release(callLegSessionID, cause)
or
1. enableNotifications(appCallControlManager)
assignmentID
notificationInfo.CallEventInfo.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
2. reportNotification(callReference, callLegReferenceSet, notificationInfo, assignmentID)
appCallBack
3. createCallLeg(callSessionID, appCallLeg)
callLeg
4. eventReportReq(callLegSessionID, eventsRequested)
5. routeReq(callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties)
6. release(callLegSessionID, cause)
or
1. enableNotifications(appCallControlManager)
assignmentID
notificationInfo.CallEventInfo.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
notificationInfo.CallEventInfo.CallEventType= P_CALL_EVENT_TERMINATING_RELEASE
2. reportNotification(callReference, callLegReferenceSet, notificationInfo, assignmentID)
appCallBack

Test MPCC_IpAppCallLeg_73

Summary: request reference of call related to call leg

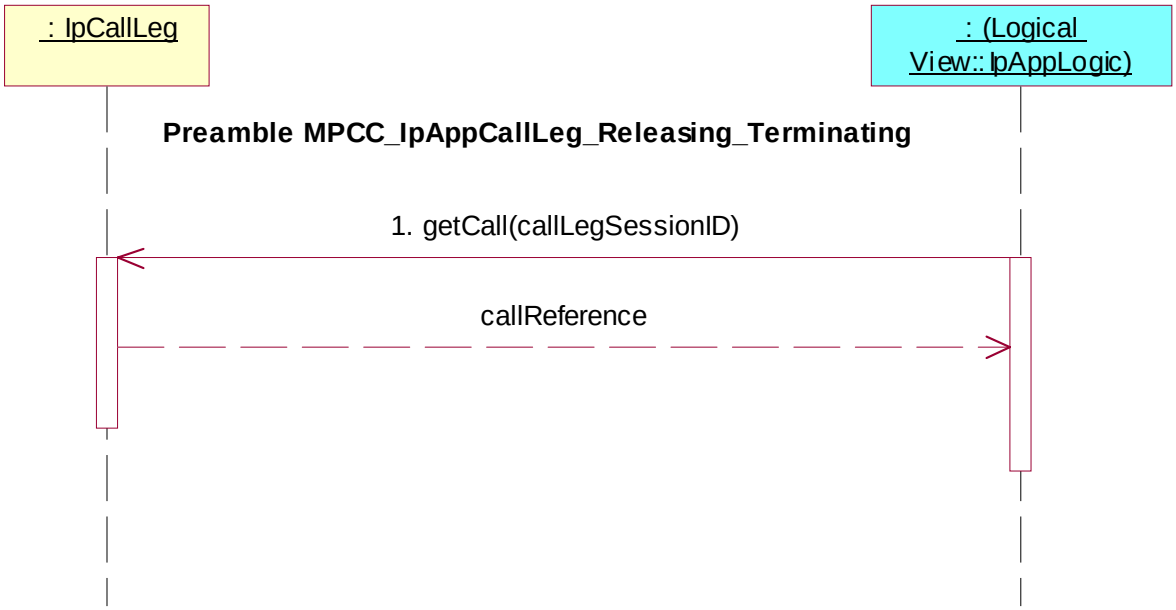
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **getCall()**

Preamble: **MPCC_IpAppCallLeg_Releasing_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **getCall()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID



Test MPCC_IpAppCallLeg_74

Summary: request reference of call related to call leg

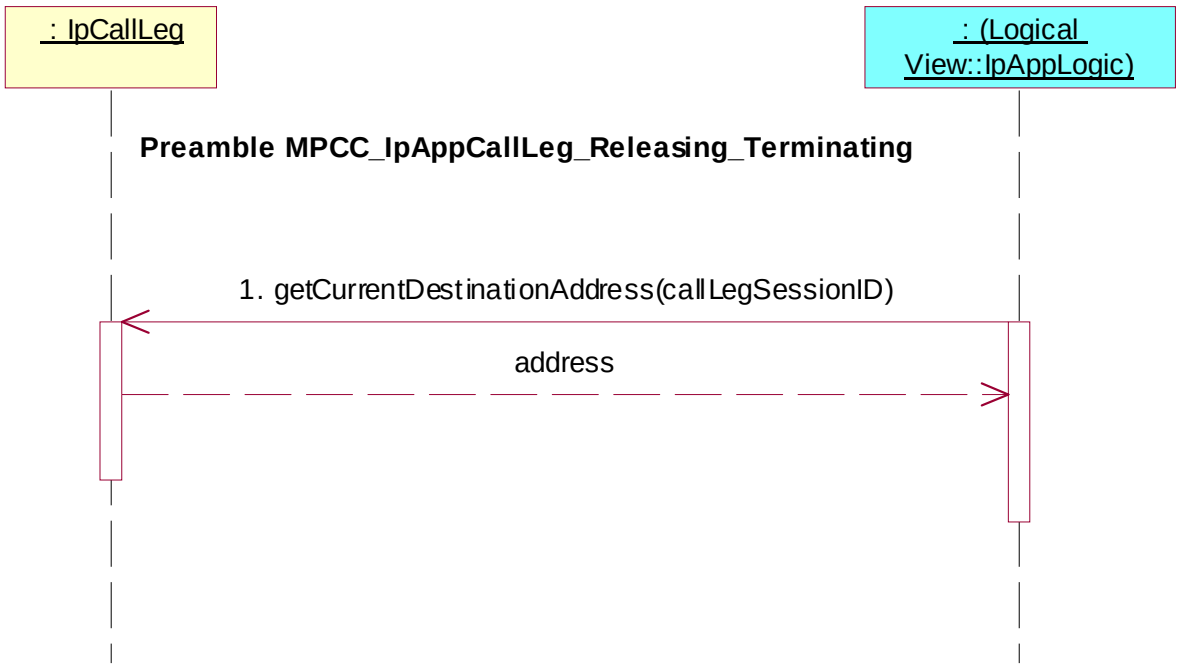
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **getCurrentDestinationAddress()**

Preamble: **MPCC_IpAppCallLeg_Releasing_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **getCurrentDestinationAddress()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID



Test MPCC_IpAppCallLeg_75

Summary: continue processing of call leg

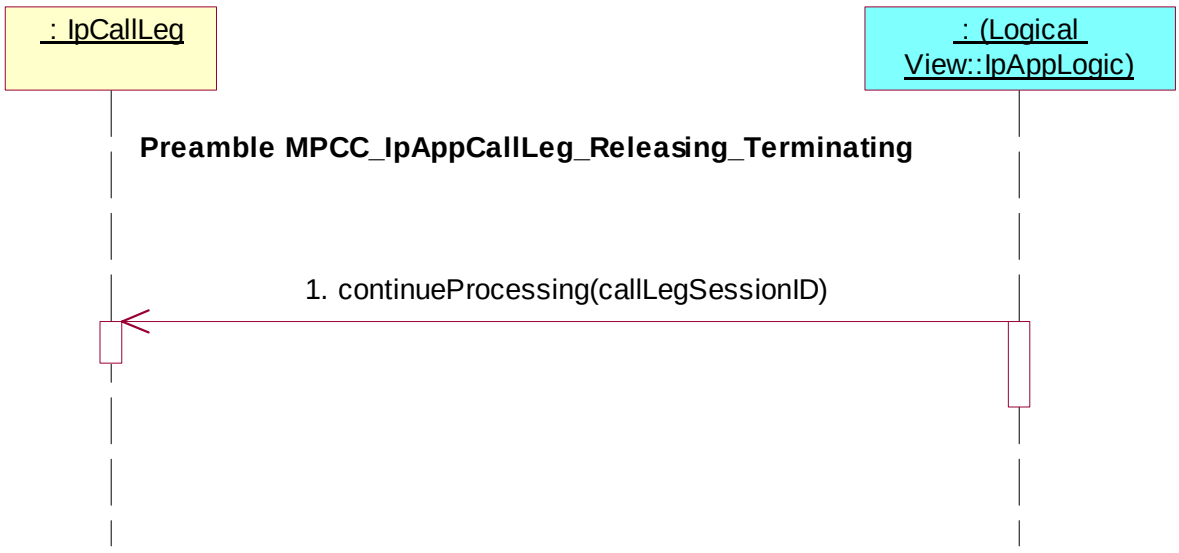
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **continueProcessing()**

Preamble: **MPCC_IpAppCallLeg_Releasing_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **continueProcessing()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID



Test MPCC_IpAppCallLeg_76

Summary: continue processing of call leg

Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **getInfoReq()**, **continueProcessing()** and **release()**

Preamble: **MPCC_IpAppCallLeg_Active_Terminating**

Test Sequence:

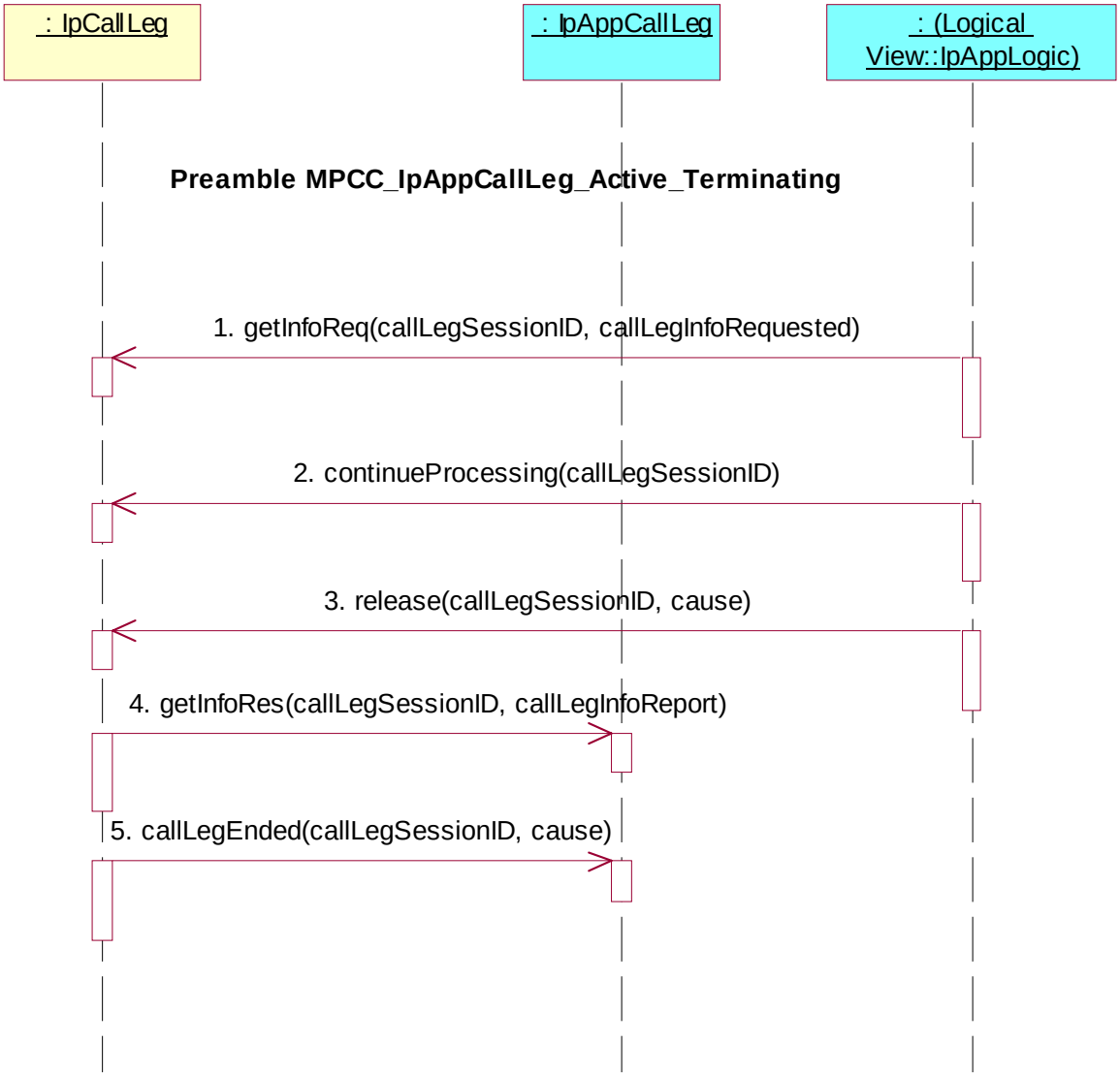
1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, callLegInfoRequested

2. Triggered Action: cause IUT to call **continueProcessing()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID

3. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, cause

4. Method call **getInfoRes()**
   Parameters: callLegSessionID, callLegInfoReport
   Check: no exception is returned

5. Method call **callLegEnded()**
   Parameters: callLegSessionID, cause
   Check: no exception is returned



Test MPCC_IpAppCallLeg_77

Summary: continue processing of call leg

Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **superviseReq()**, **continueProcessing()** and **release()**

Preamble: **MPCC_IpAppCallLeg_Active_Terminating**

Test Sequence:

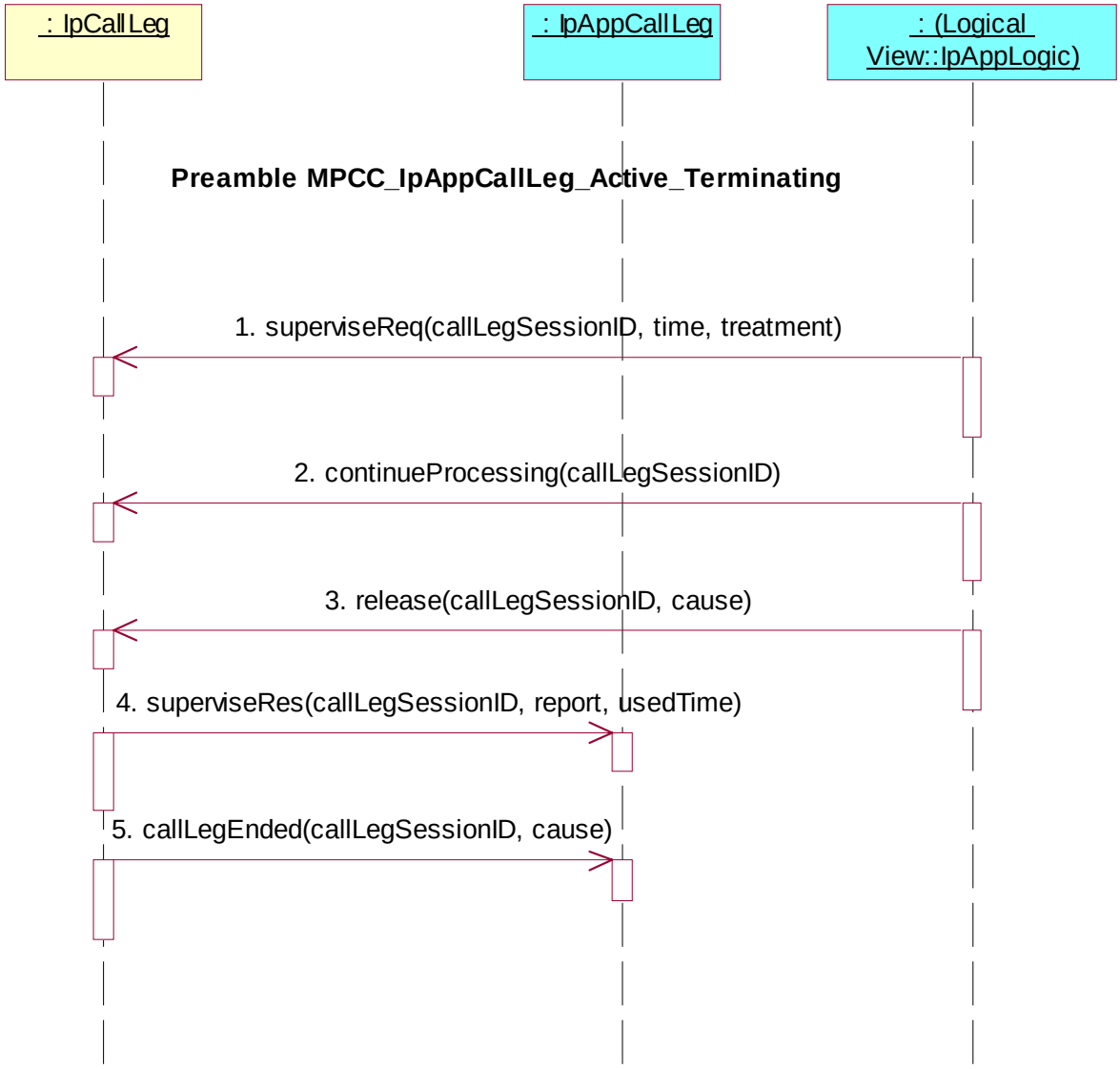
1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, time, treatment

2. Triggered Action: cause IUT to call **continueProcessing()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID

3. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID, cause

4. Method call **superviseRes()**
   Parameters: callLegSessionID, report, usedTime
   Check: no exception is returned

5. Method call **callLegEnded()**
   Parameters: callLegSessionID, cause
   Check: no exception is returned



Test MPCC_IpAppCallLeg_78

Summary: de-assign call leg

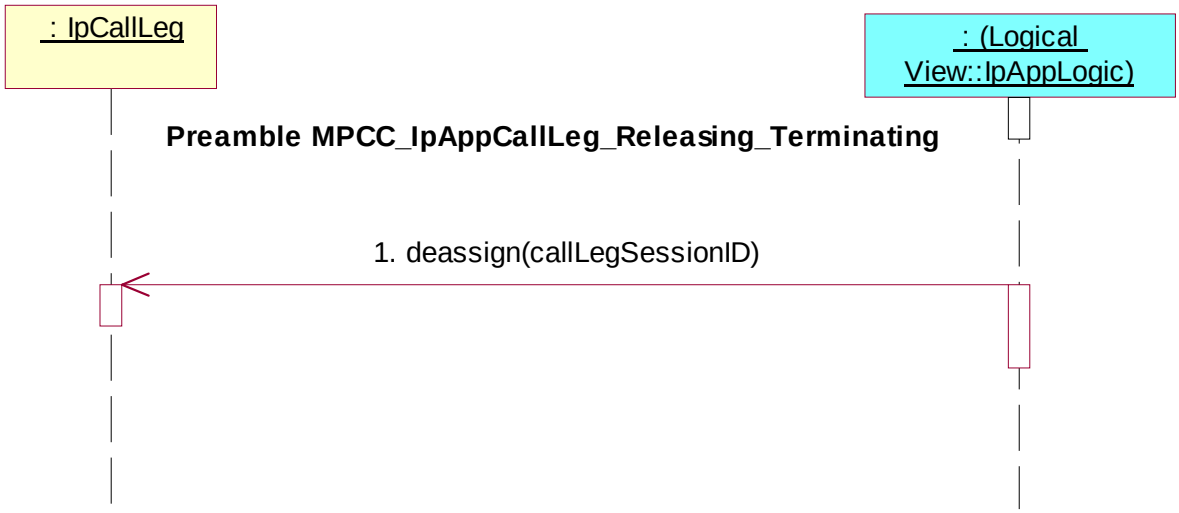
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **deassign()**

Preamble: **MPCC_IpAppCallLeg_Releasing_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **deassign()** method on the tester's (SCF's) terminating IpCallLeg interface.
   Parameters: callLegSessionID



7.2.3 MultiMedia Call Control Service (MMCC)

The TPs in this clause are based on ES 202 915-4-4 [4].

7.2.3.1 IpAppMultiMediaCallControlManager

Test MMCC_IpAppMultiMediaCallControlManager_01

Summary: create call object

Reference: ES 202 915-4-3 [3], clause 6.1

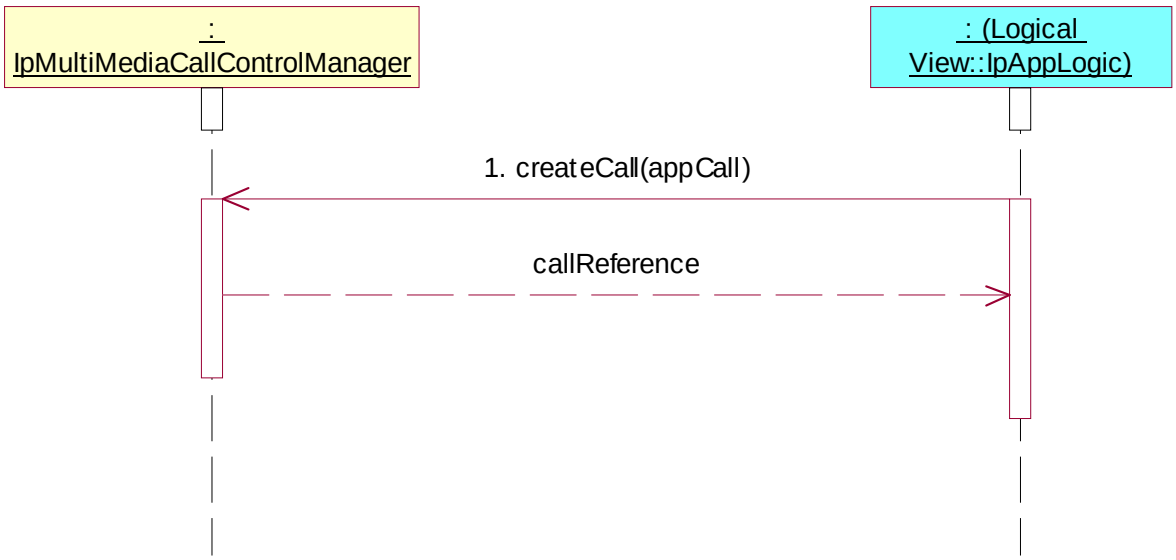
Precondition: IUT capable of invoking **createCall()**

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

The application is permitted to provide its IpAppMultiMediaCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **createCall()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCall



Test MMCC_IpAppMultiMediaCallControlManager_02

Summary: create call object and accept abort

Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.2

Precondition: IUT capable of invoking **createCall()**; **callAborted()** implemented

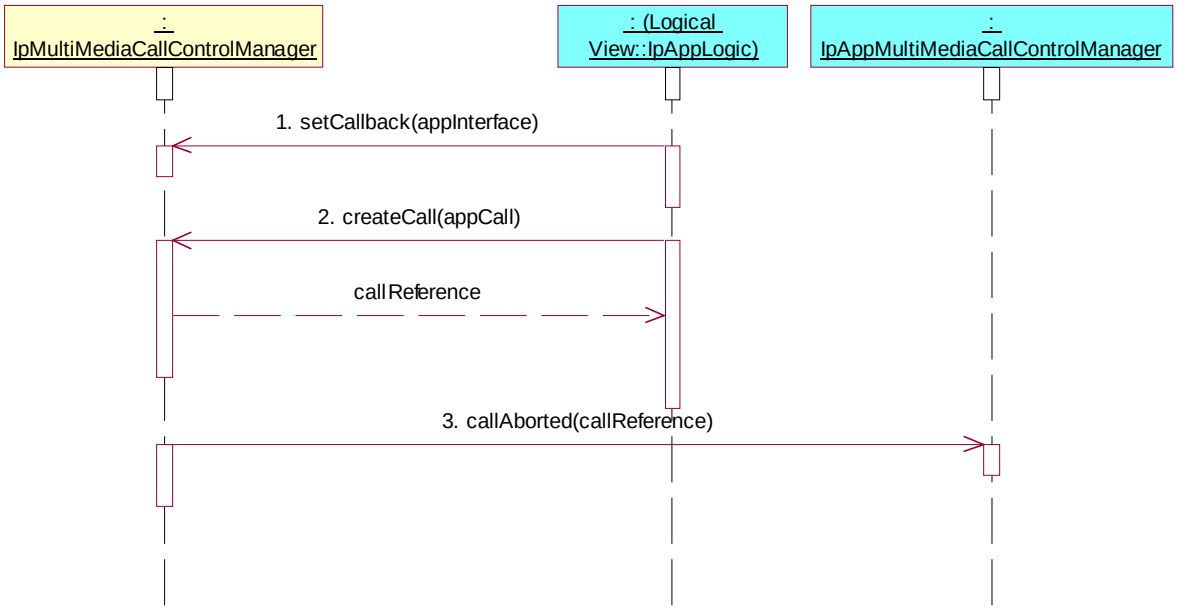
Preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiMediaCallControlManager interface through selecting that service and signing the required service agreement.

Test Sequence:

1. Triggered Action: cause IUT to call **setCallBack()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: valid, non NULL, value of appInterface

2. Triggered Action: cause IUT to call **createCall()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCall

3. Method call **callAborted()**
   Parameters: callReference
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallControlManager_03

Summary: enable and accept call notifications

Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.2

Precondition: IUT capable of invoking **createNotification()**, **reportNotification()** implemented

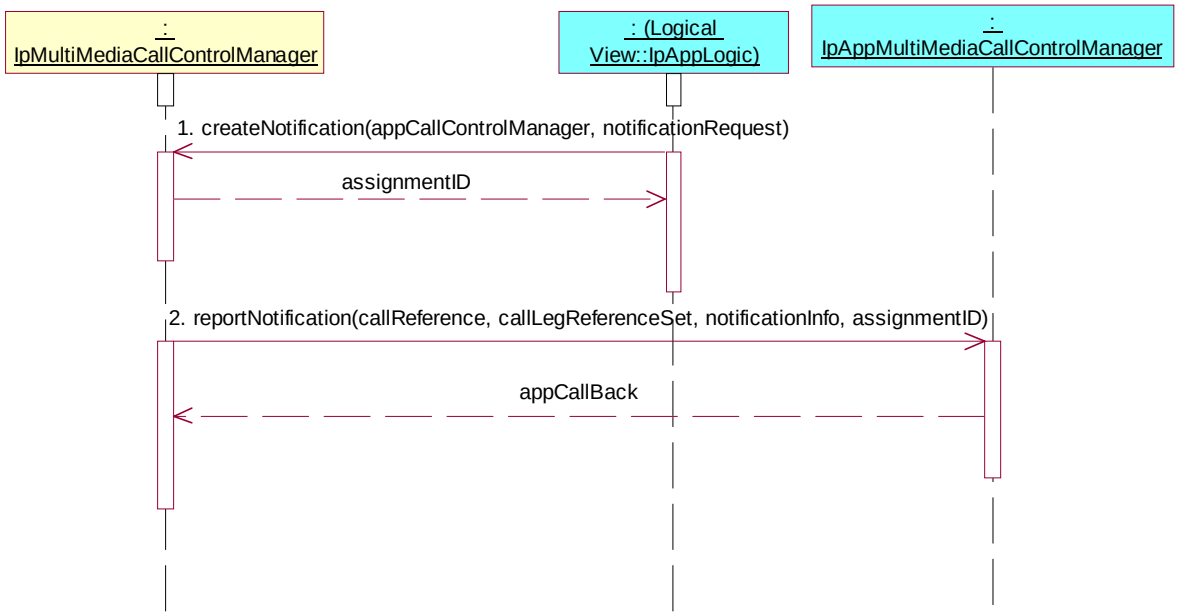
Preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiMediaCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiMediaCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   Check: valid value of TpAppMultiMediaCallBack is returned



Test MMCC_IpAppMultiMediaCallControlManager_04

Summary: interrupt and continue call notifications

Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.2

Precondition: IUT capable of invoking **createNotification()**, **managerInterrupted(), managerContinued()** and **reportNotification()** implemented

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

The application is permitted to provide its IpAppMultiMediaCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest

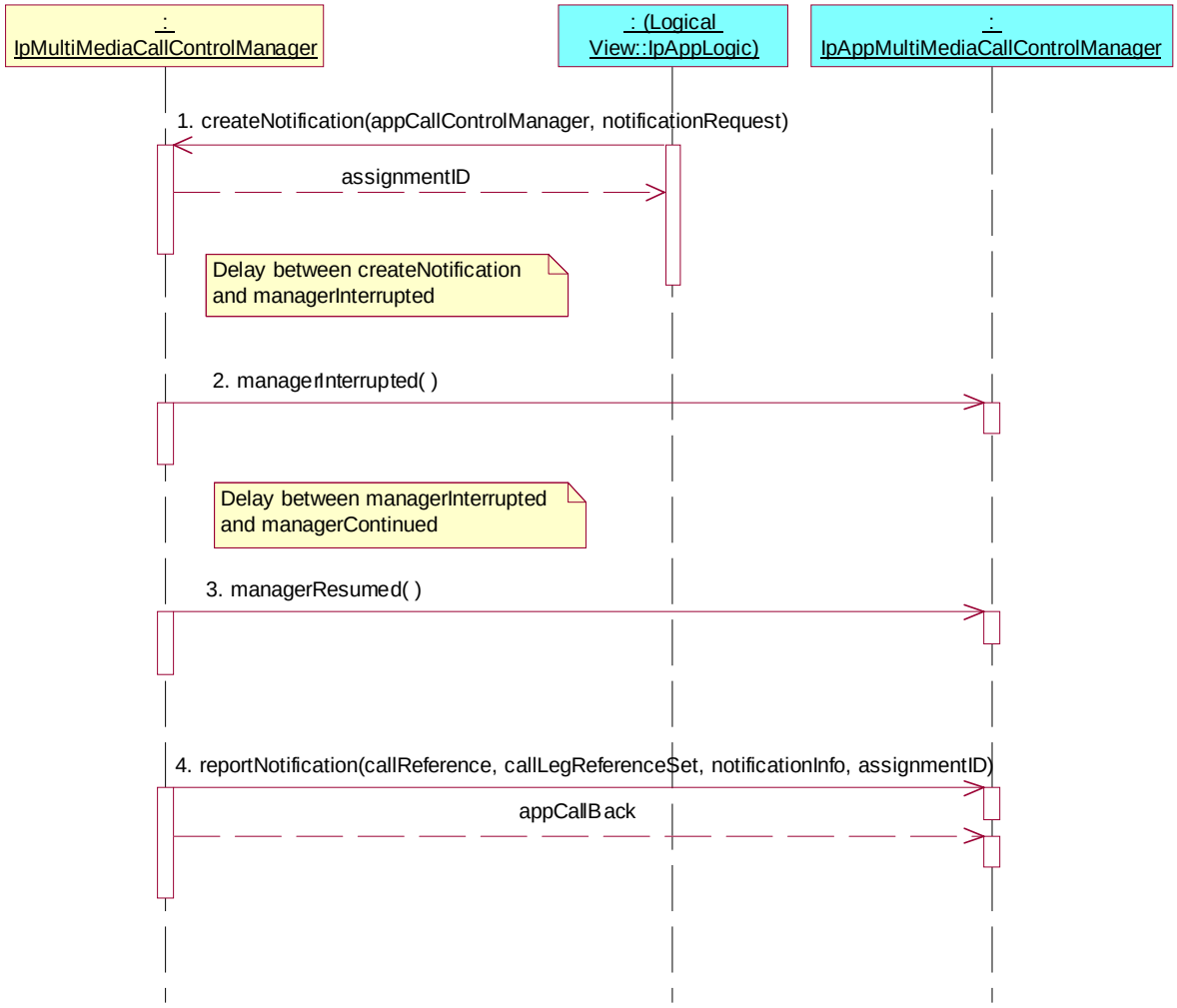
Delay between createNotification and managerInterrupted

2. Method call **managerInterrupted()**
   Parameters: none
   Check: no exception is returned

Delay between managerInterrupted and managerContinued

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

4. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   Check: valid value of TpAppMultiMediaCallBack is returned



Test MMCC_IpAppMultiMediaCallControlManager_05

Summary: enable and change call notifications

Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.2

Precondition: IUT capable of invoking **createNotification()** and **changeNotification()**, **reportNotification()** implemented

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

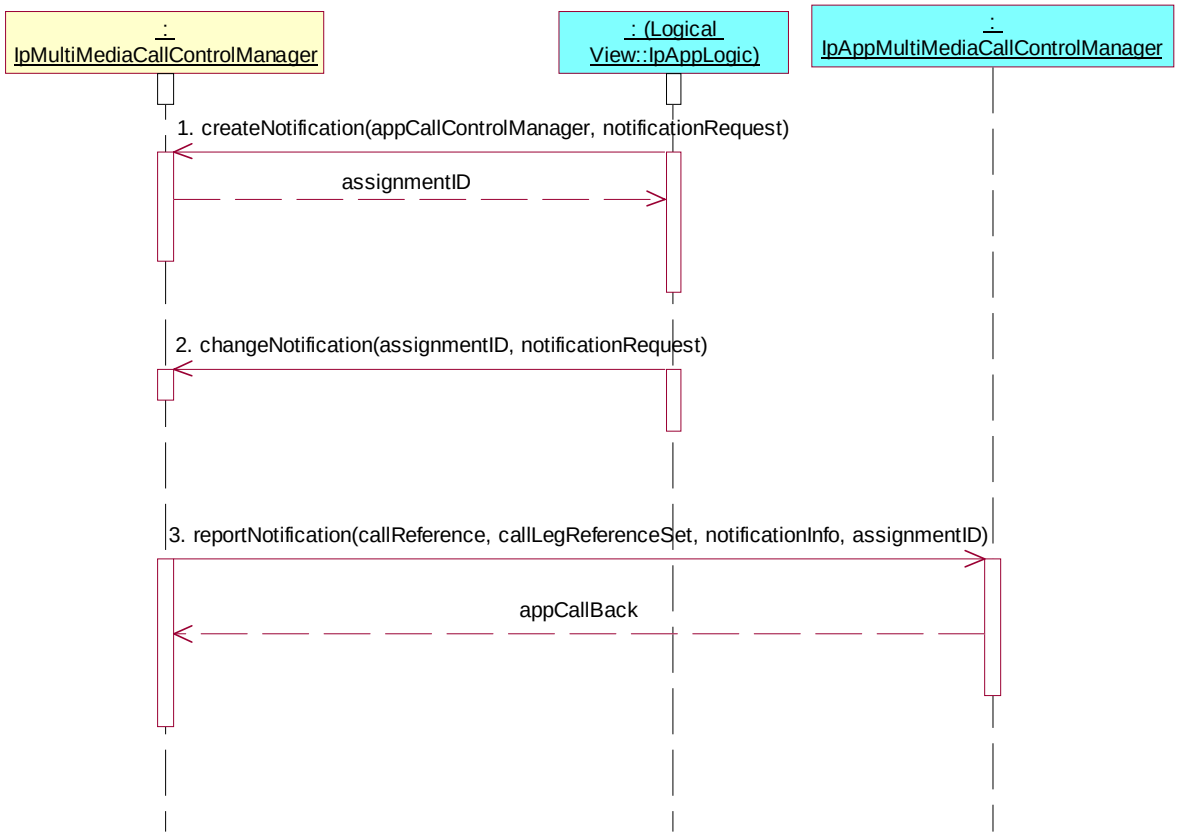
The application is permitted to provide its IpAppMultiMediaCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest

2. Triggered Action: cause IUT to call **changeNotification()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: assignmentID, notificationRequest

3. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   Check: valid value of TpAppMultiMediaCallBack is returned



Test MMCC_IpAppMultiMediaCallControlManager_06

Summary: enable and destroy call notifications

Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.2

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

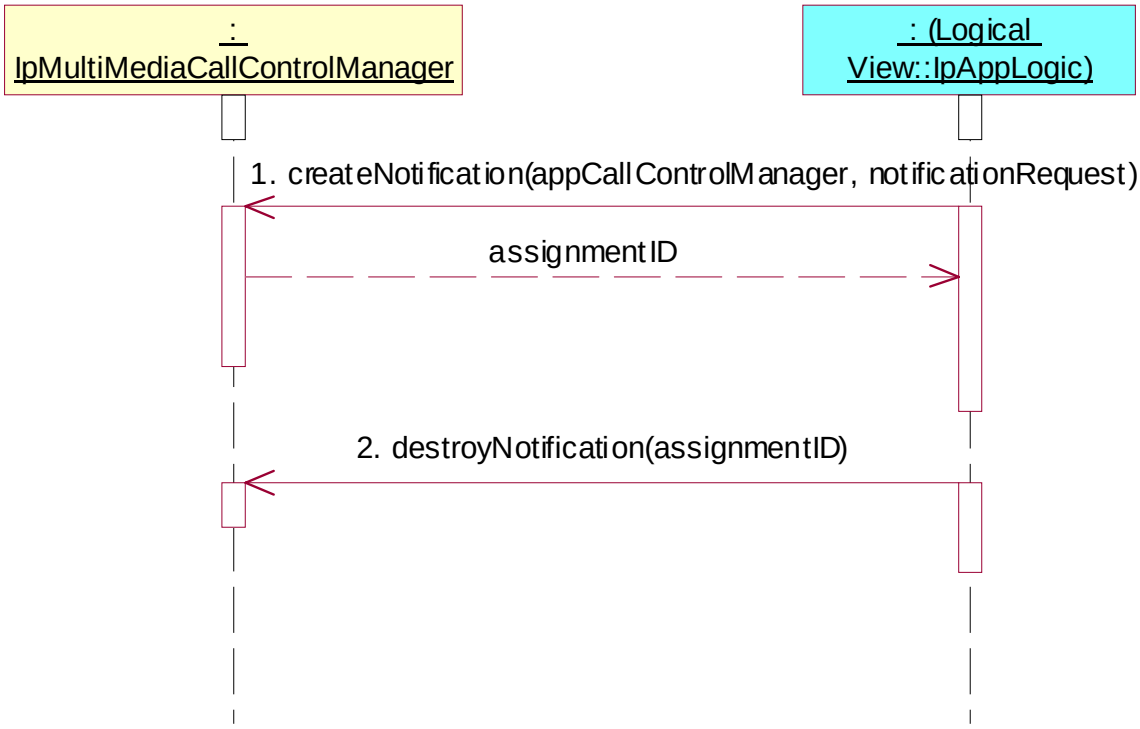
Preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiMediaCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiMediaCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest

2. Triggered Action: cause IUT to call **destroyNotification()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: assignmentID



Test MMCC_IpAppMultiMediaCallControlManager_07

Summary: enable call notifications and get criteria

Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.2

Precondition: IUT capable of invoking **createNotification()** and **getNextNotification()**

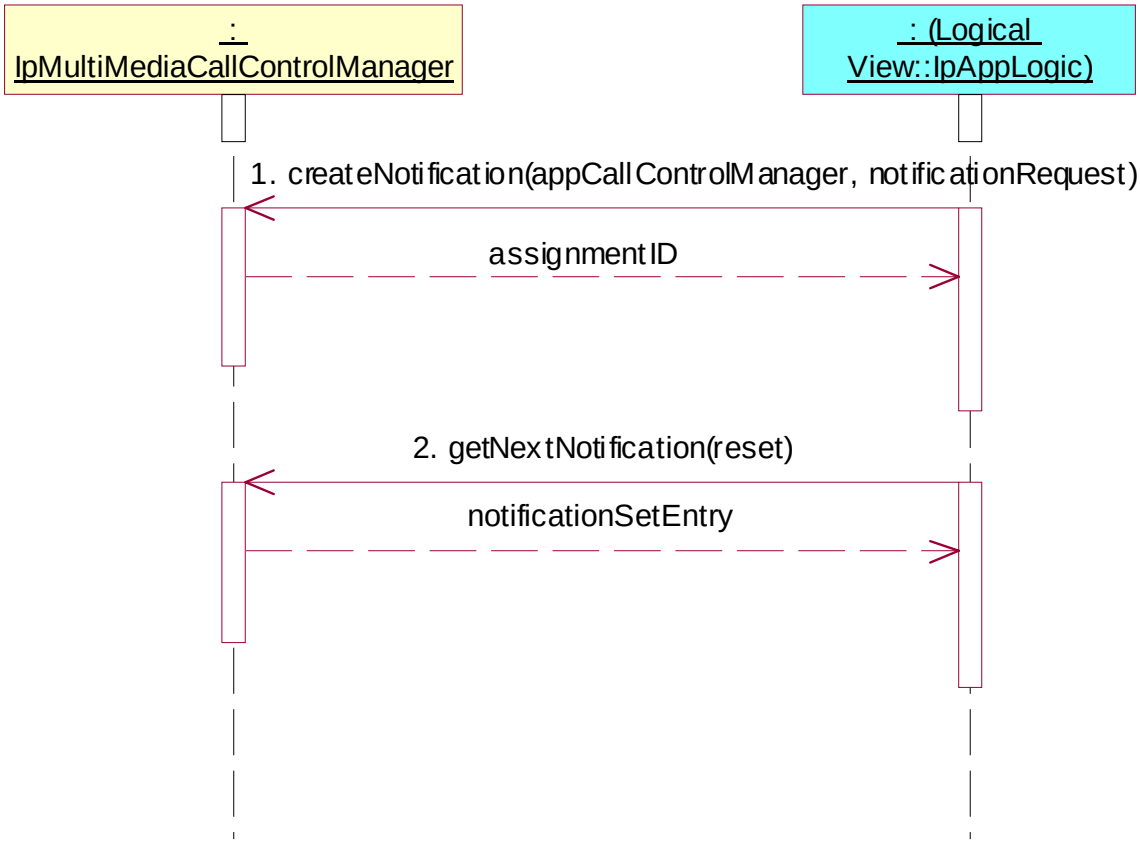
Preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiMediaCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiMediaCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest

2. Triggered Action: cause IUT to call **getNextNotification()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: reset



Test MMCC_IpAppMultiMediaCallControlManager_08

Summary: enable load control and accept overload notifications

Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.2

Precondition: IUT capable of invoking **setCallLoadControl()**, **callOverloadEncountered()** and **callOverloadCeased()** implemented

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

The application is permitted to provide its IpAppMultiMediaCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest

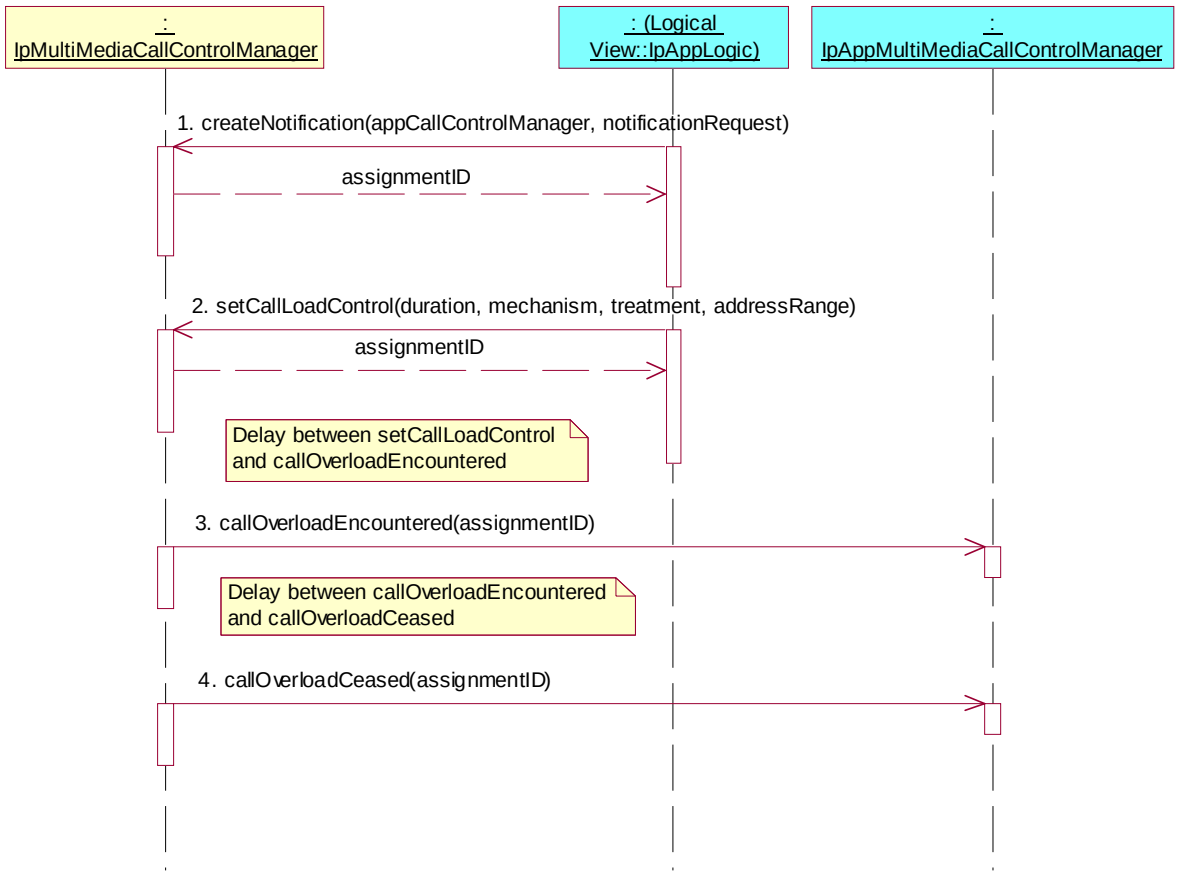
2. Triggered Action: cause IUT to call **setCallLoadControl()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: duration, mechanism, treatment, addressRange

Delay between setCallLoadControl and callOverloadEncountered

3. Method call **callOverloadEncountered()**
   Parameters: assignmentID
   Check: no exception is returned

Delay between callOverloadEncountered and callOverloadCeased

4. Method call **callOverloadCeased()**
   Parameters: assignmentID
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallControlManager_09

Summary: enable and accept call notifications

Reference: ES 202 915-4-3 [3], clauses 6.1 and 6.2

Precondition: IUT capable of invoking **enableNotifications()**, **reportNotification()** implemented

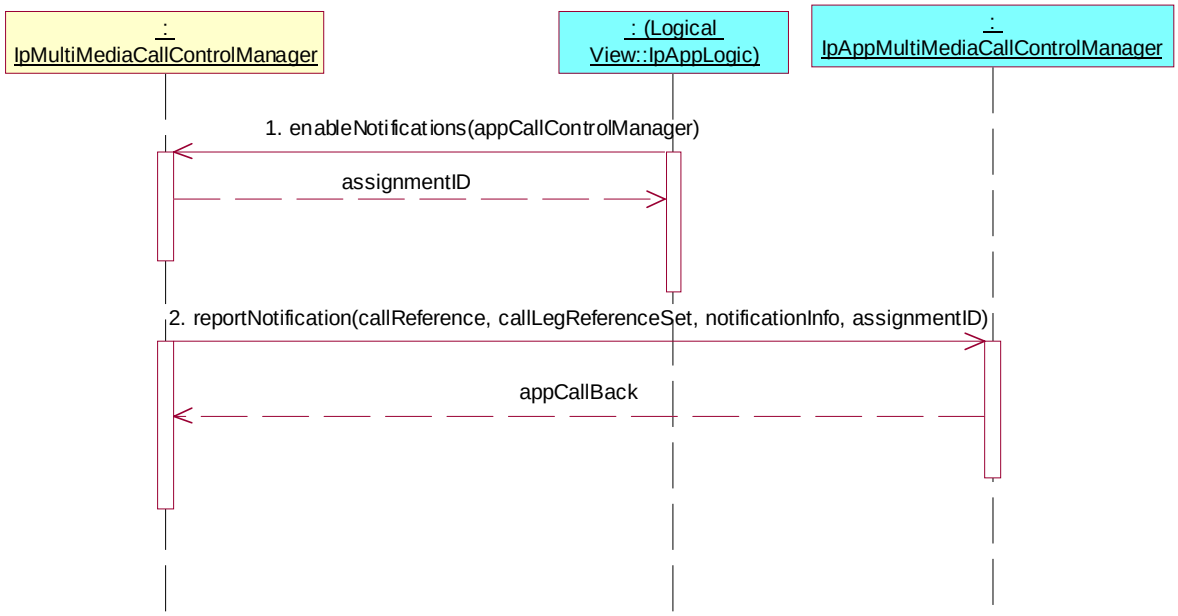
Preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiMediaCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiMediaCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **enableNotifications()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCallControlManager

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   Check: valid value of TpAppMultiPartyCallBack is returned



Test MMCC_IpAppMultiMediaCallControlManager_10

Summary: enable and disable call notifications

Reference: ES 202 915-4-3 [3], clause 6.1

Precondition: IUT capable of invoking **enableNotifications()** and **disableNotifications()**

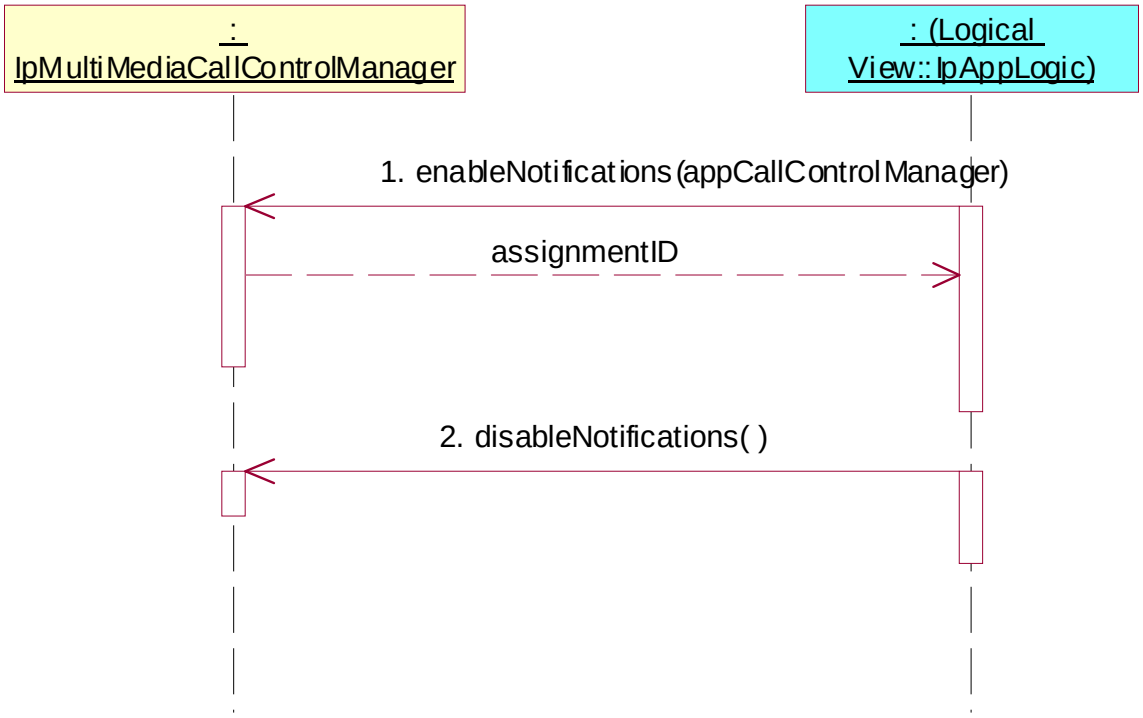
Preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiMediaCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiMediaCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **enableNotifications()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCallControlManager

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



Test MMCC_IpAppMultiMediaCallControlManager_11

Summary: create media stream notifications and accept report

Reference: ES 202 915-4-4 [4], clauses 6.1 and 6.2

Precondition: IUT capable of invoking **createMediaNotification()**, **reportMediaNotification()** implemented

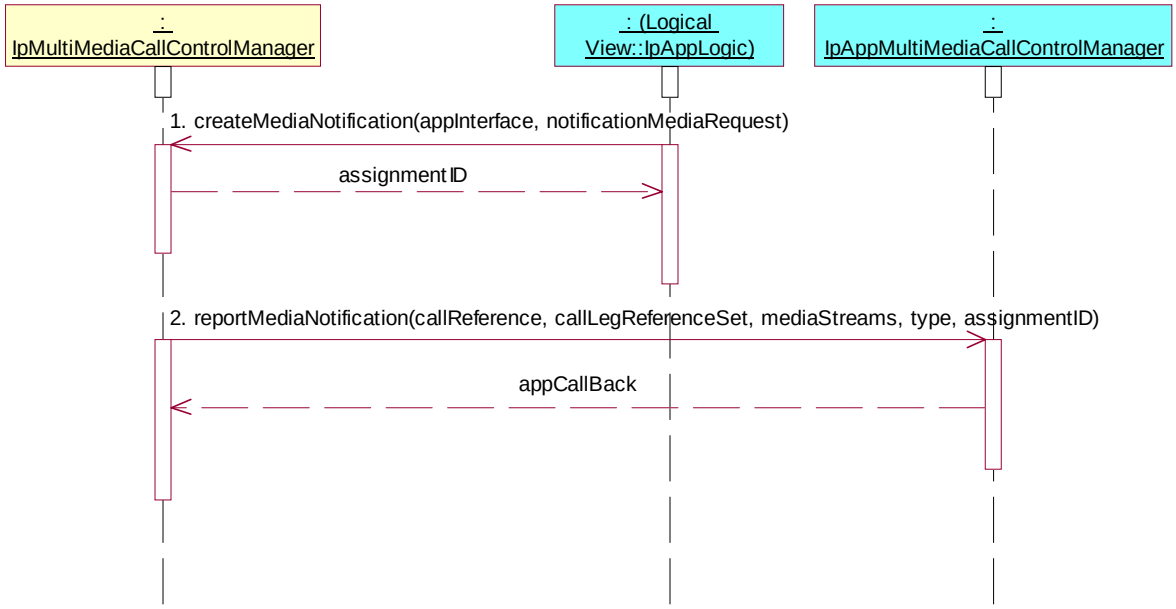
Preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiMediaCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiMediaCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **createMediaNotification()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appInterface, notificationMediaRequest

2. Method call **reportMediaNotification()**
   Parameters: callReference, callLegReferenceSet, mediaStreams, type
   Check: valid value of TpAppMultiMediaCallBack is returned



Test MMCC_IpAppMultiMediaCallControlManager_12

Summary: create and destroy media stream notifications

Reference: ES 202 915-4-4 [4], clauses 6.1 and 6.2

Precondition: IUT capable of invoking **createMediaNotification()** and **destroyMediaNotification()**

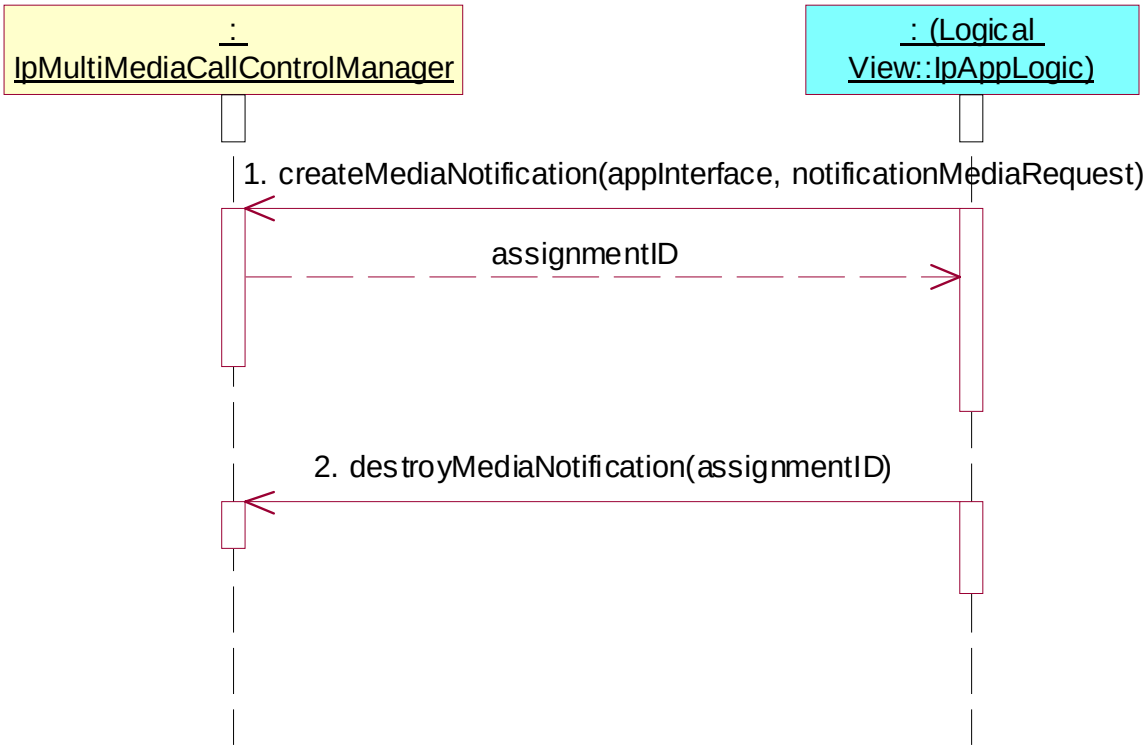
Preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiMediaCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiMediaCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **createMediaNotification()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appInterface, notificationMediaRequest

2. Triggered Action: cause IUT to call **destroyMediaNotification()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: assignmentID



Test MMCC_IpAppMultiMediaCallControlManager_13

Summary: create and modify media stream notifications

Reference: ES 202 915-4-4 [4], clauses 6.1 and 6.2

Precondition: IUT capable of invoking **createMediaNotification()** and **changeMediaNotification()**, **reportMediaNotification()** implemented

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

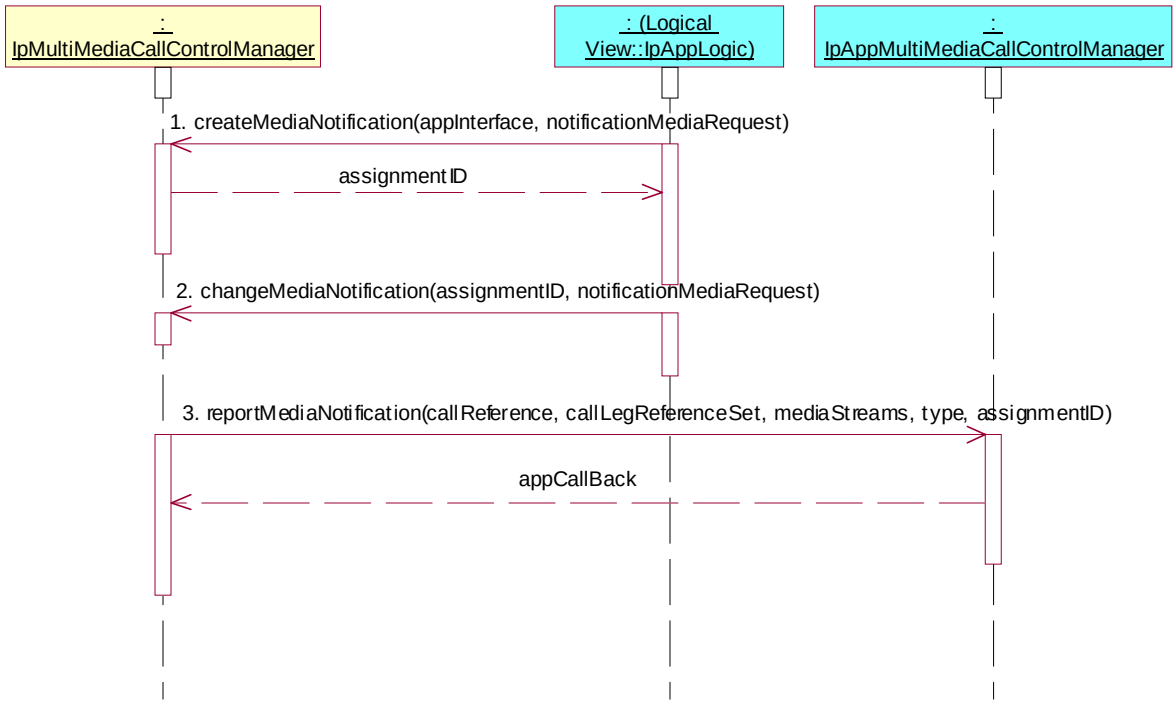
The application is permitted to provide its IpAppMultiMediaCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **createMediaNotification()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appInterface, notificationMediaRequest

2. Triggered Action: cause IUT to call **changeMediaNotification()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: assignmentID, notificationMediaRequest

3. Method call **reportMediaNotification()**
   Parameters: callReference, callLegReferenceSet, mediaStreams, type
   Check: valid value of TpAppMultiMediaCallBack is returned



Test MMCC_IpAppMultiMediaCallControlManager_14

Summary: create and get media stream notifications

Reference: ES 202 915-4-4 [4], clauses 6.1 and 6.2

Precondition: IUT capable of invoking **createMediaNotification()** and **getMediaNotification()**

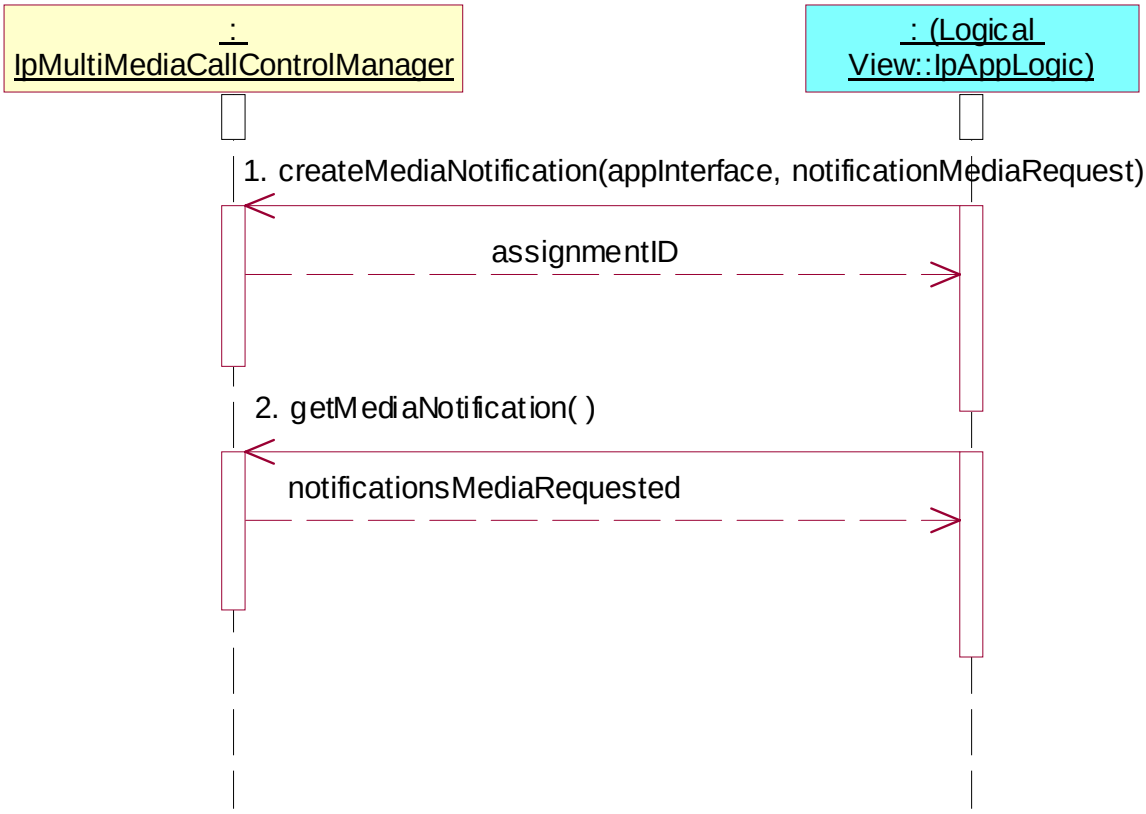
Preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiMediaCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiMediaCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **createMediaNotification()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appInterface, notificationMediaRequest

2. Triggered Action: cause IUT to call **getMediaNotification()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: none



7.2.3.2 IpAppMultiMediaCall

Applications need not be capable of performing each of the sequences below, even if they support the methods indicated below.

Reference: ES 202 915-4-3 [3], clause 7.4.2, and ES 202 915-4-4 [4] clauses 6.3 and 6.4

7.2.3.2.1 Idle state

Precondition: IUT capable of invoking **createCall()**

Preamble MMCC_IpAppMultiMediaCall_Idle

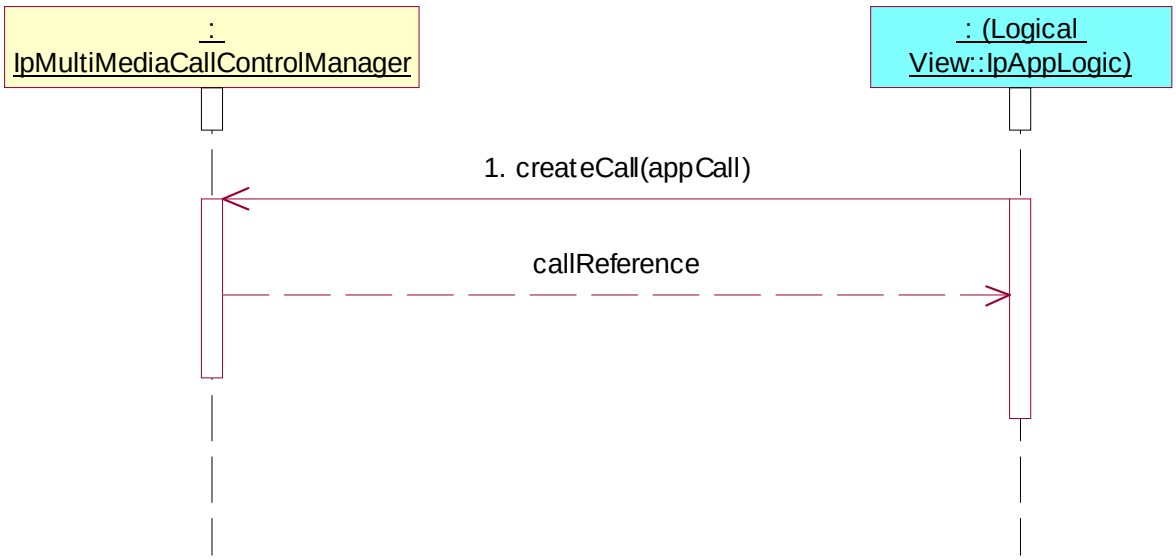
Reference: ES 202 915-4-3 [3], clause 7.4.2.1

Pre-preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiMediaCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiMediaCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Preamble Sequence:

1. Triggered Action: cause IUT to call **createCall()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCall



Test MMCC_IpAppMultiMediaCall_01

Summary: create call leg

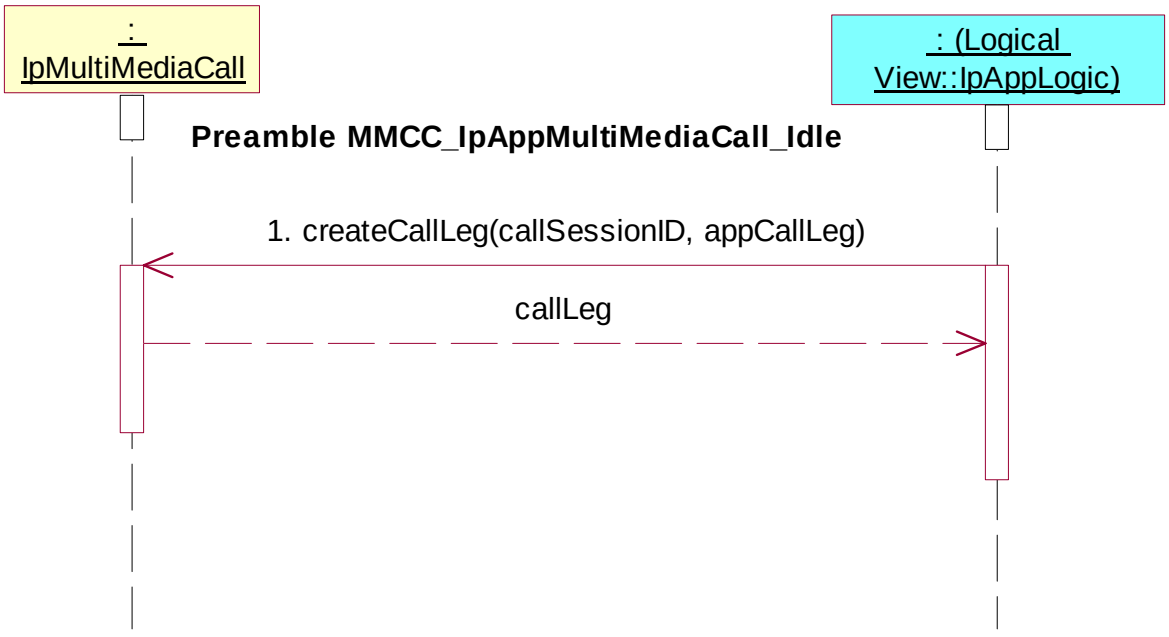
Reference: ES 202 915-4-3 [3], clause 7.2.1

Precondition: IUT capable of invoking **createCallLeg()**

Preamble: **MMCC_IpAppMultiMediaCall_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **createCallLeg()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID, appCallLeg



Test MMCC_IpAppMultiMediaCall_02

Summary: create and route call leg, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.2.1

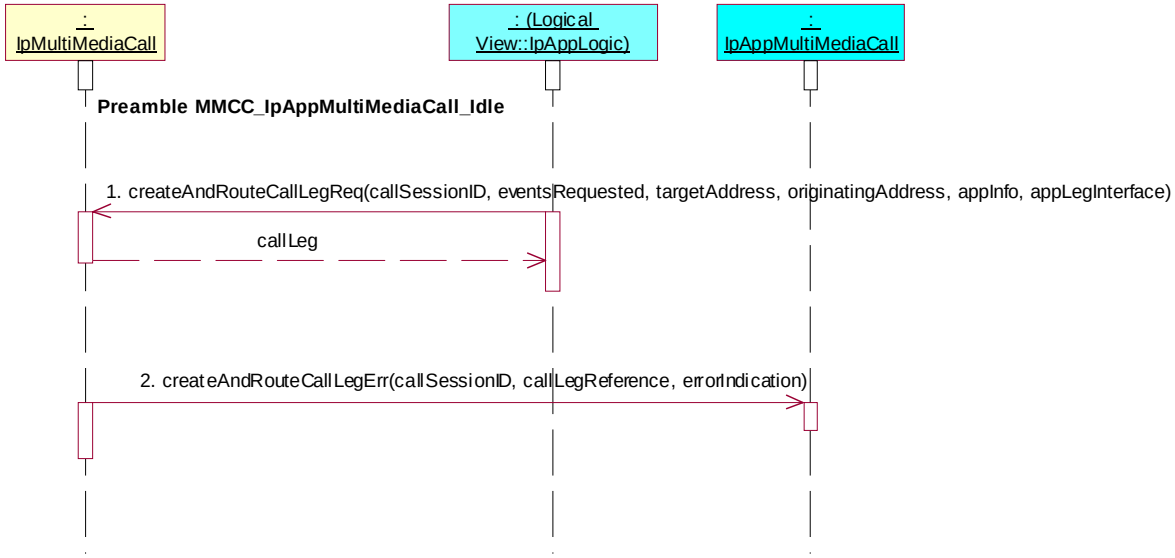
Precondition: IUT capable of invoking **createAndRouteCallLegReq()**

Preamble: **MMCC_IpAppMultiMediaCall_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **createAndRouteCallLegReq()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID, eventsRequested, targetAddress, originatingAddress, appInfo, appLegInterface

2. Method call **createAndRouteCallLegErr()**
   Parameters: callSessionID, appCallLegReference, errorIndication
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCall_03

Summary: supervise call

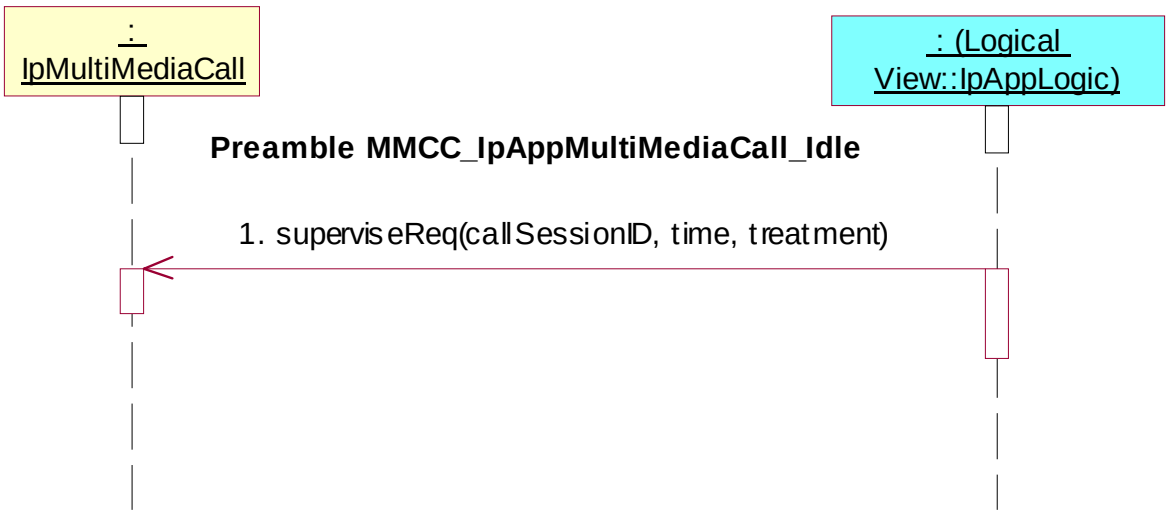
Reference: ES 202 915-4-3 [3], clause 7.2.1

Precondition: IUT capable of invoking **superviseReq()**

Preamble: **MMCC_IpAppMultiMediaCall_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID, time, treatment



Test MMCC_IpAppMultiMediaCall_04

Summary: request call information

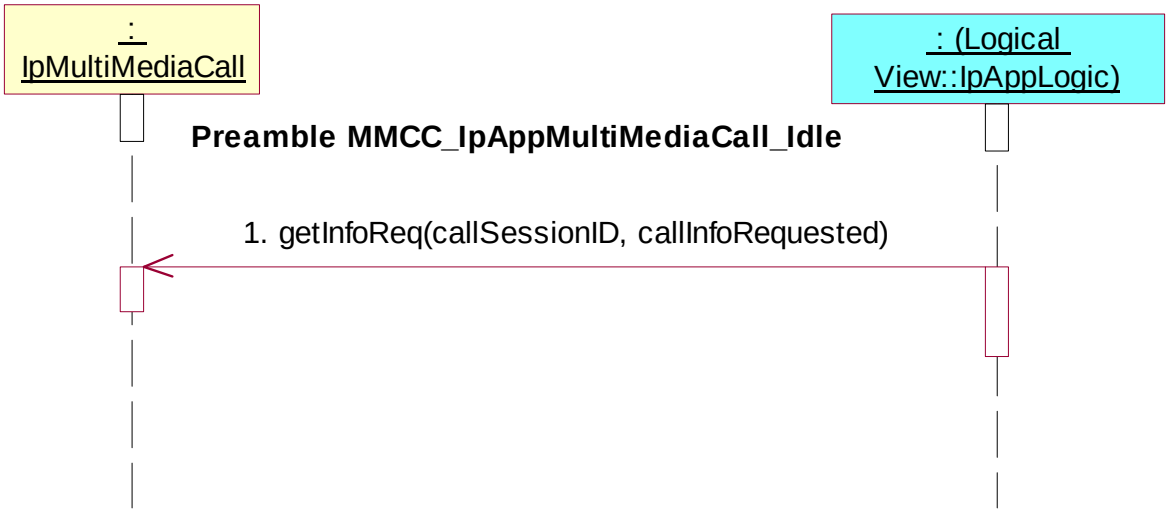
Reference: ES 202 915-4-3 [3], clause 7.2.1

Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **MMCC_IpAppMultiMediaCall_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID, callInfoRequested



Test MMCC_IpAppMultiMediaCall_05

Summary: request call information, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.2.1

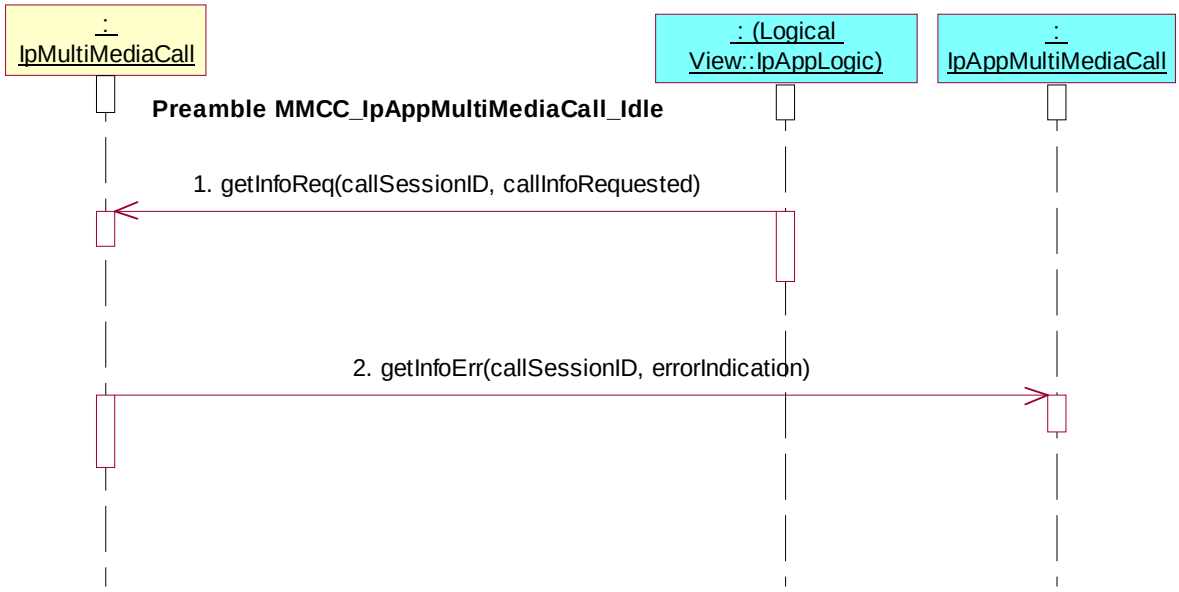
Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **MMCC_IpAppMultiMediaCall_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID, callInfoRequested

2. Method call **getInfoErr()**
   Parameters: callSessionID, errorIndication
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCall_06

Summary: set charge plan for the call

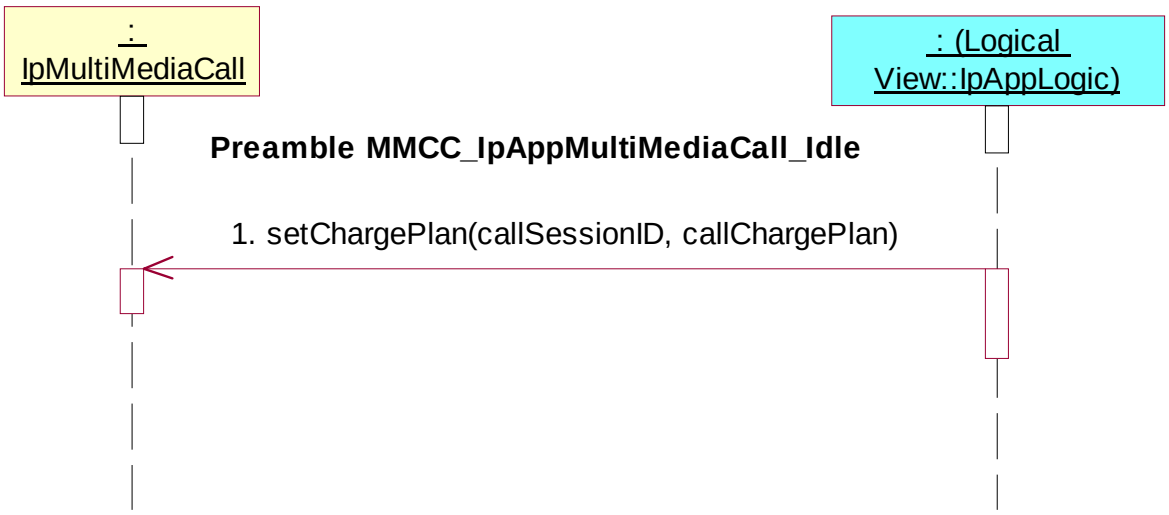
Reference: ES 202 915-4-3 [3], clause 7.2.1

Precondition: IUT capable of invoking **setChargePlan()**

Preamble: **MMCC_IpAppMultiMediaCall_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **setChargePlan()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID, callChargePlan



Test MMCC_IpAppMultiMediaCall_07

Summary: allow advice of charge information

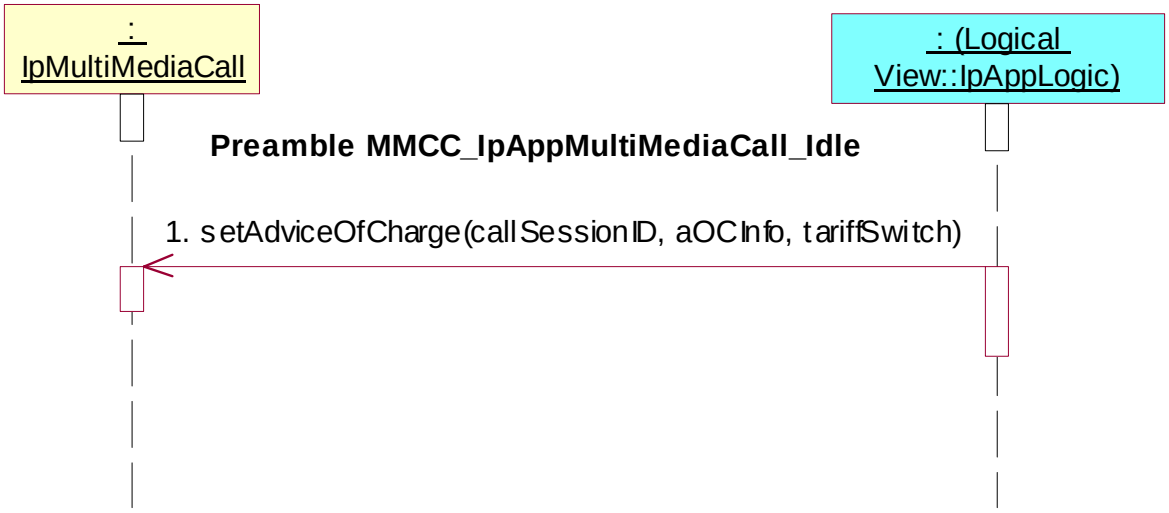
Reference: ES 202 915-4-3 [3], clause 7.2.1

Precondition: IUT capable of invoking **setAdviceOfCharge()**

Preamble: **MMCC_IpAppMultiMediaCall_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **setAdviceOfCharge()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID, aOCInfo, tariffSwitch



Test MMCC_IpAppMultiMediaCall_08

Summary: supervise call with granted volume

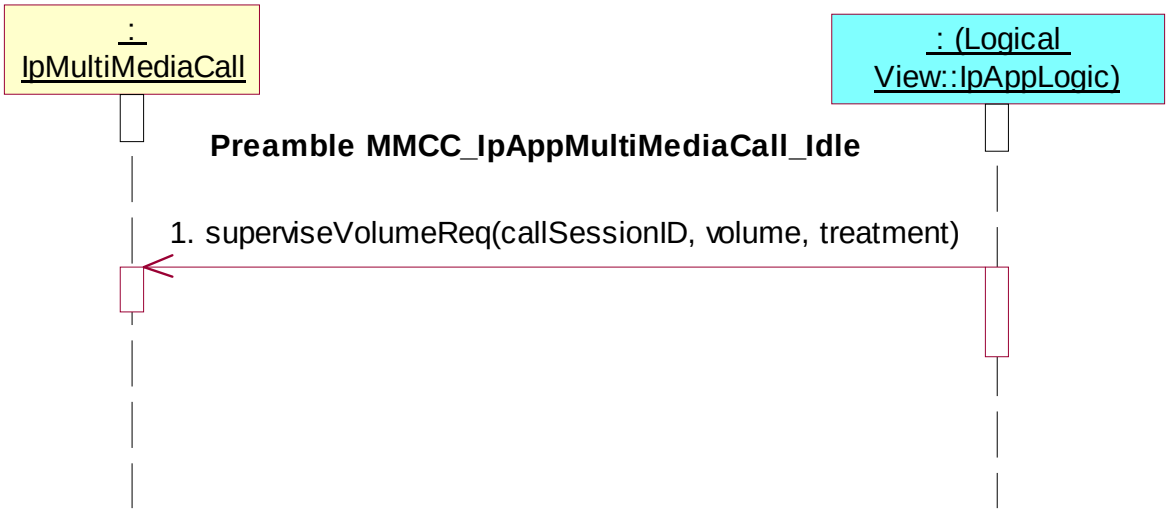
Reference: ES 202 915-4-4 [4], clause 6.3

Precondition: IUT capable of invoking **superviseVolumeReq()**

Preamble: **MMCC_IpAppMultiMediaCall_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseVolumeReq()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID, volume, treatment



7.2.3.2.2 Active state

Precondition: IUT capable of invoking **createCall()** and **createCallLeg()**

or IUT capable of invoking **createCall()** and **createAndRouteCallLegReq()**

or IUT capable of invoking **createNotification()**

Preamble MMCC_IpAppMultiMediaCall_Active

Reference: ES 202 915-4-3 [3], clause 7.2.2

Pre-preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiMediaCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiMediaCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Preamble Sequence:

1. Triggered Action: cause IUT to call **createCall()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCall

2. Triggered Action: cause IUT to call **createCallLeg()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID, appCallLeg

or

1. Triggered Action: cause IUT to call **createCall()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCall

2. Triggered Action: cause IUT to call **createAndRouteCallLegReq()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID, eventsRequested, targetAddress, originatingAddress, appInfo, appLegInterface

or

1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest
   notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   Check: valid value of TpAppMultiMediaCallBack is returned

or

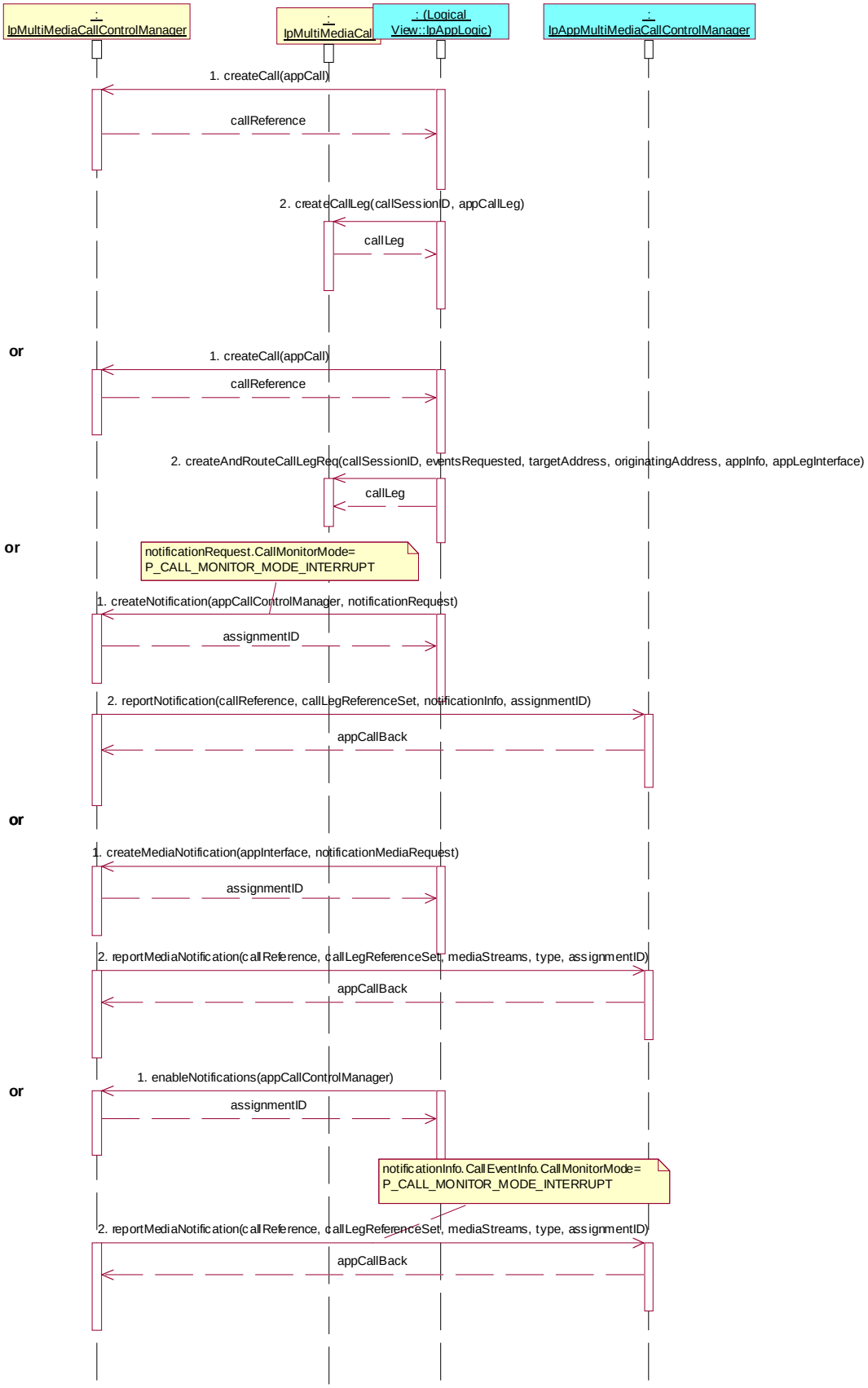
1. Triggered Action: cause IUT to call **createMediaNotification()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appInterface, notificationMediaRequest

2. Method call **reportMediaNotification()**
   Parameters: callReference, callLegReferenceSet, mediaStreams, type
   Check: valid value of TpAppMultiMediaCallBack is returned

or

1. Triggered Action: cause IUT to call **enableNotifications()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCallControlManager

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   notificationInfo.CallEventInfo.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   Check: valid value of TpAppMultiMediaCallBack is returned


: IpMultiMediaCallControlManager
: IpMultiMediaCall
: (Logical View::IpAppLogic)
: IpAppMultiMediaCallControlManager
1. createCall(appCall)
callReference
2. createCallLeg(callSessionID, appCallLeg)
callLeg
or
1. createCall(appCall)
callReference
2. createAndRouteCallLegReq(callSessionID, eventsRequested, targetAddress, originatingAddress, appInfo, appLegInterface)
callLeg
or
notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
1. createNotification(appCallControlManager, notificationRequest)
assignmentID
2. reportNotification(callReference, callLegReferenceSet, notificationInfo, assignmentID)
appCallBack
or
1. createMediaNotification(appInterface, notificationMediaRequest)
assignmentID
2. reportMediaNotification(callReference, callLegReferenceSet, mediaStreams, type, assignmentID)
appCallBack
or
1. enableNotifications(appCallControlManager)
assignmentID
notificationInfo.CallEventInfo.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
2. reportMediaNotification(callReference, callLegReferenceSet, mediaStreams, type, assignmentID)
appCallBack

Test MMCC_IpAppMultiMediaCall_09

Summary: create call leg

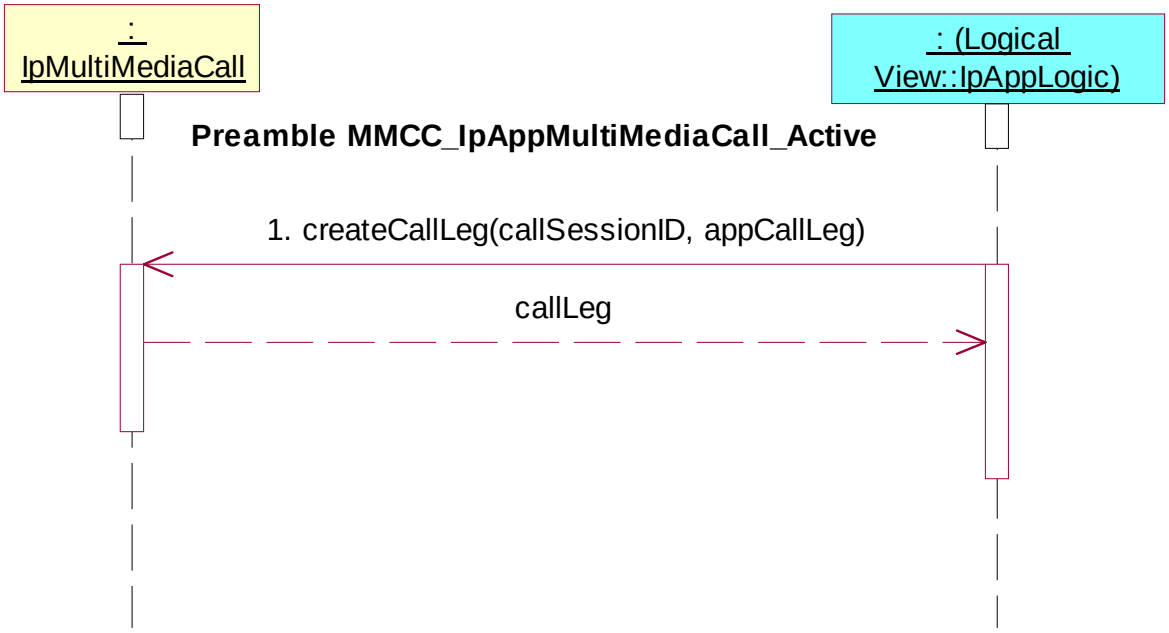
Reference: ES 202 915-4-3 [3], clause 7.2.2

Precondition: IUT capable of invoking **createCallLeg()**

Preamble: **MMCC_IpAppMultiMediaCall_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **createCallLeg()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID, appCallLeg



Test MMCC_IpAppMultiMediaCall_10

Summary: create and route call leg, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.2.2

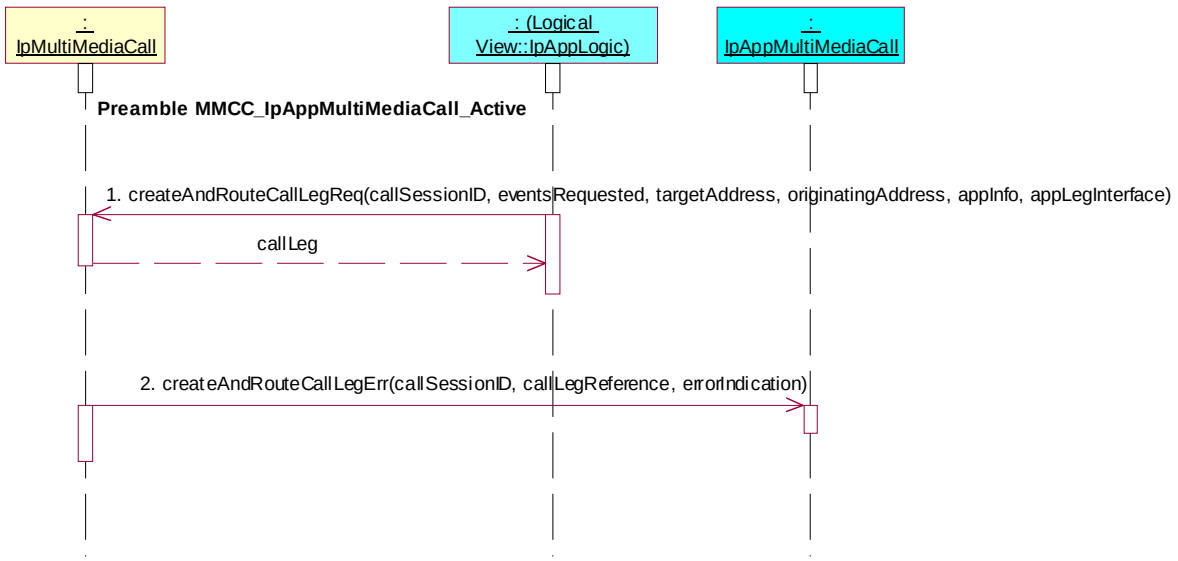
Precondition: IUT capable of invoking **createAndRouteCallLegReq()**

Preamble: **MMCC_IpAppMultiMediaCall_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **createAndRouteCallLegReq()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID, eventsRequested, targetAddress, originatingAddress, appInfo, appLegInterface

2. Method call **createAndRouteCallLegErr()**
   Parameters: callSessionID, appCallLegReference, errorIndication
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCall_11

Summary: supervise call, successful

Reference: ES 202 915-4-3 [3], clause 7.2.2

Precondition: IUT capable of invoking **superviseReq()**

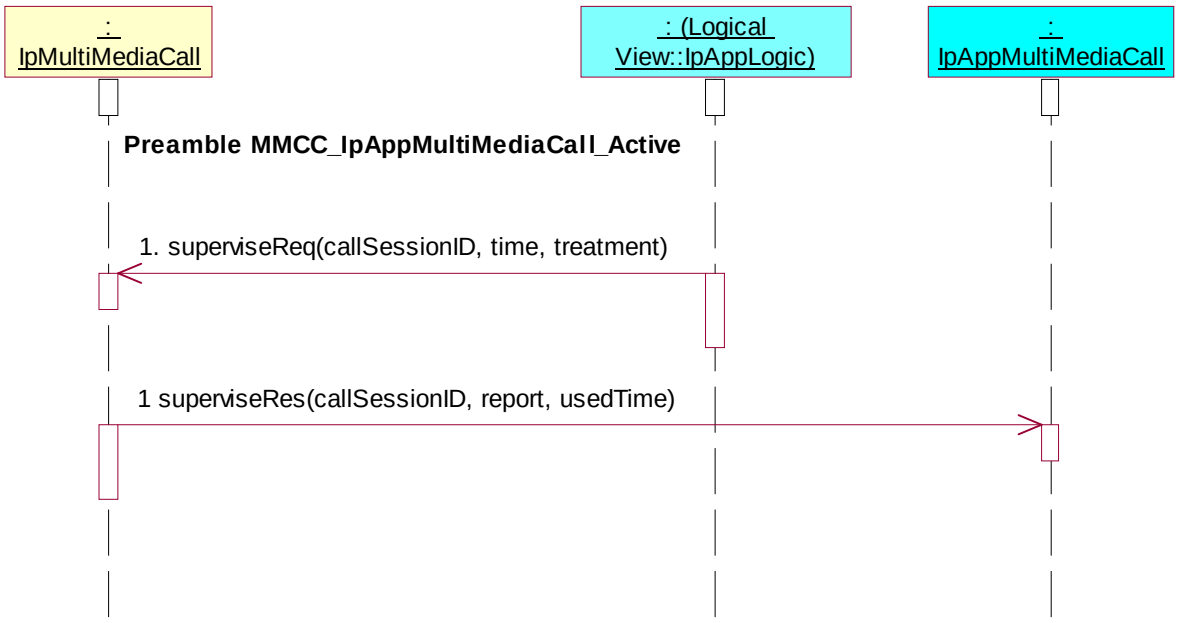
Preamble: **MMCC_IpAppMultiMediaCall_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID, time, treatment

NOTE: Between these two method sequences, the IUT may need to be triggered to complete establishment of the call to both parties, in order to justify the Tester's calling of a superviseRes() method.

2. Method call **superviseRes()**
   Parameters: callSessionID, report, usedTime
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCall_12

Summary: supervise call, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.2.2

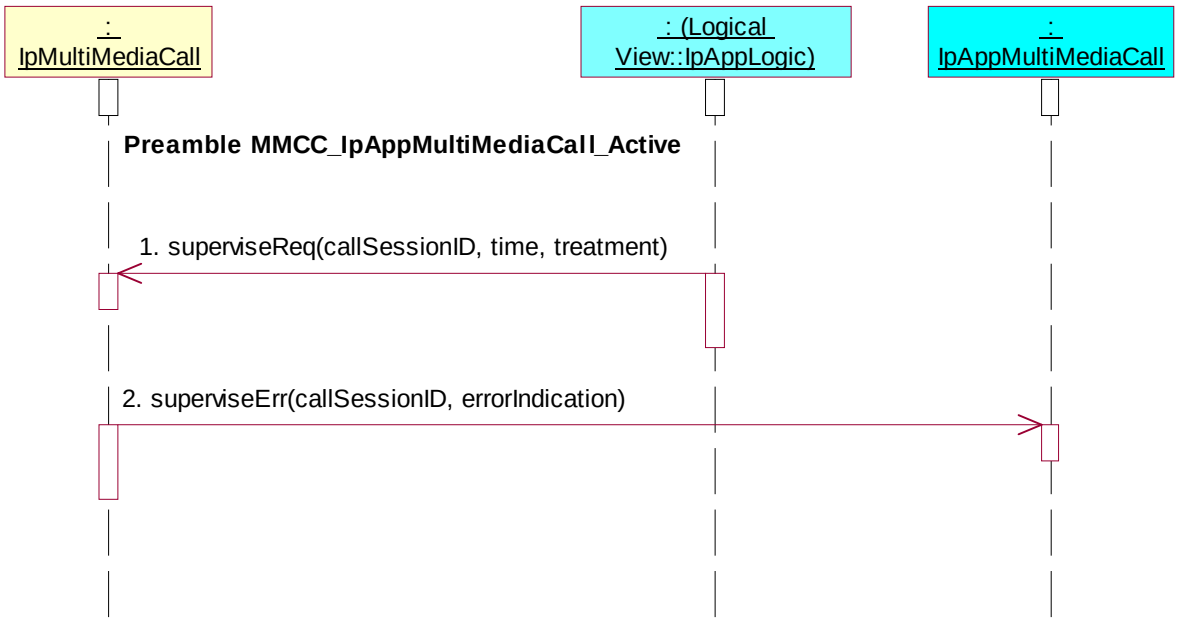
Precondition: IUT capable of invoking **superviseReq()**

Preamble: **MMCC_IpAppMultiMediaCall_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID, time, treatment

2. Method call **superviseErr()**
   Parameters: callSessionID, errorIndication
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCall_13

Summary: request call leg information

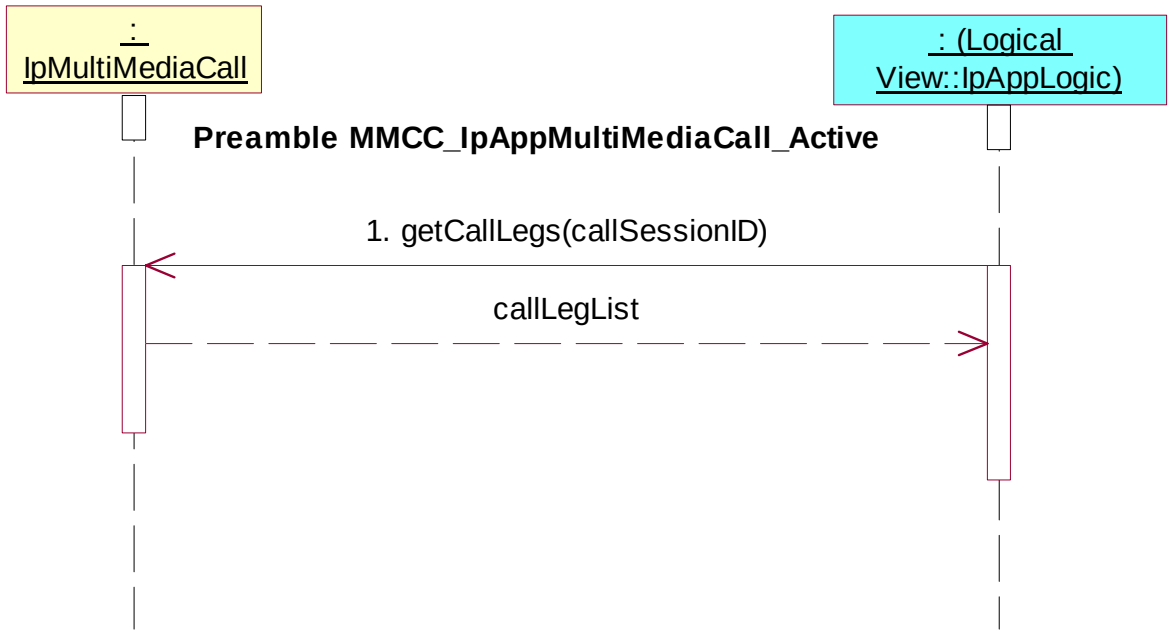
Reference: ES 202 915-4-3 [3], clause 7.2.2

Precondition: IUT capable of invoking **getCallLegs()**

Preamble: **MMCC_IpAppMultiMediaCall_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **getCallLegs()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID



Test MMCC_IpAppMultiMediaCall_14

Summary: release call

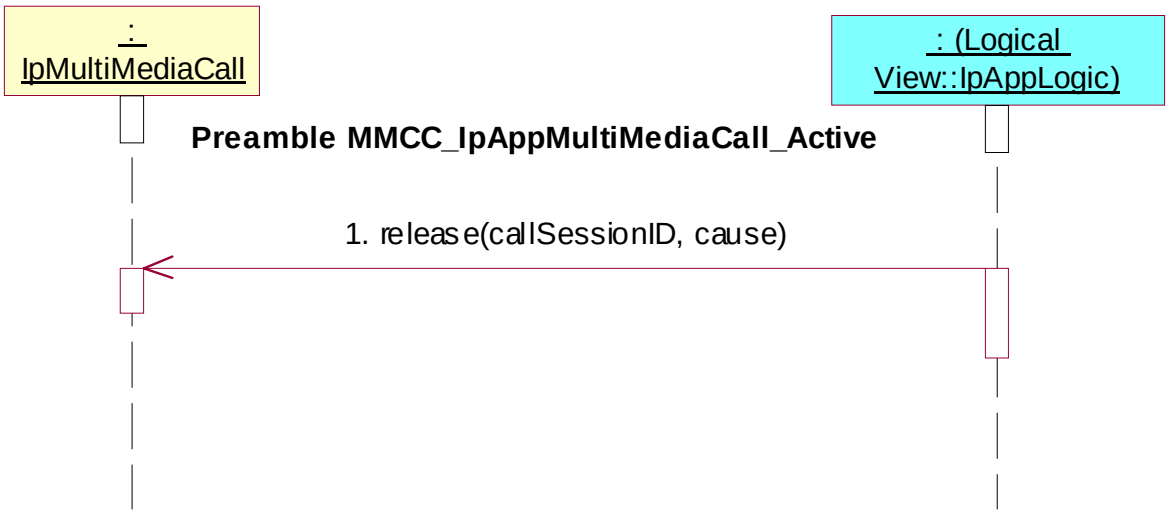
Reference: ES 202 915-4-3 [3], clause 7.2.2

Precondition: IUT capable of invoking **release()**

Preamble: **MMCC_IpAppMultiMediaCall_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID, cause



Test MMCC_IpAppMultiMediaCall_15

Summary: deassign call

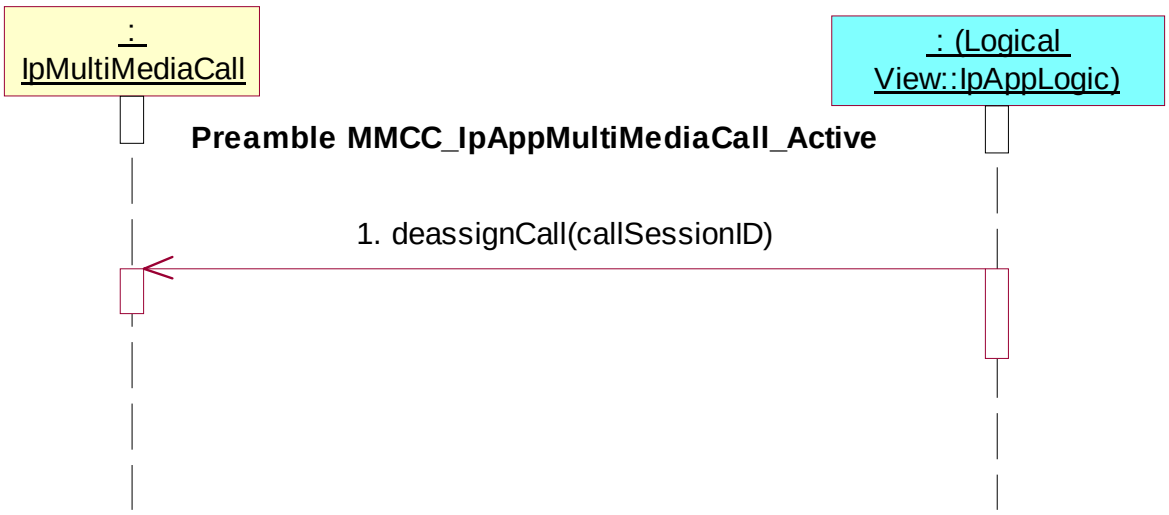
Reference: ES 202 915-4-3 [3], clause 7.2.2

Precondition: IUT capable of invoking **deassignCall()**

Preamble: **MMCC_IpAppMultiMediaCall_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **deassignCall()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID



Test MMCC_IpAppMultiMediaCall_16

Summary: indication of termination of call

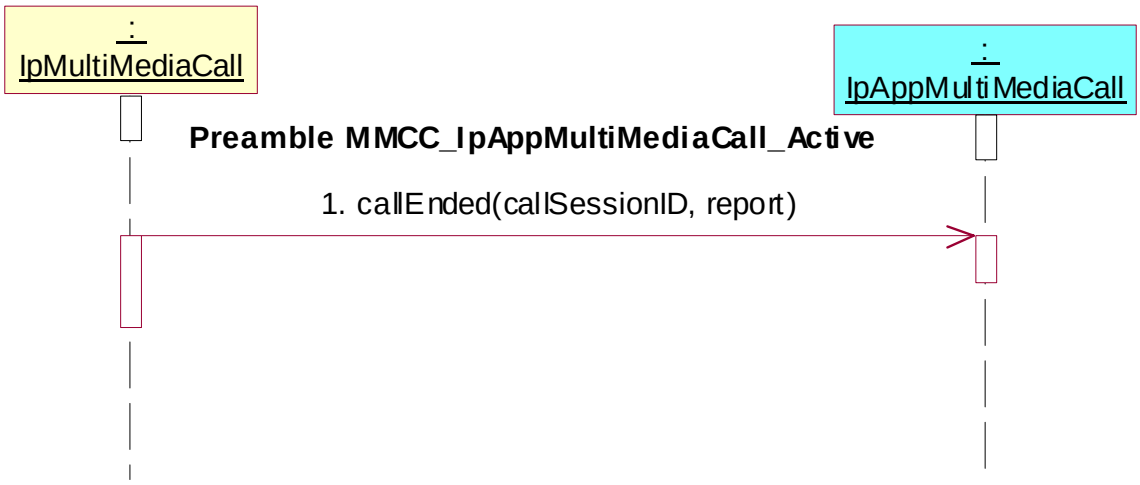
Reference: ES 202 915-4-3 [3], clause 7.2.2

Precondition: **callEnded()** implemented

Preamble: **MMCC_IpAppMultiMediaCall_Active**

Test Sequence:

1. Method call **callEnded()**
   Parameters: callSessionID, report
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCall_17

Summary: supervise call with granted volume, successful

Reference: ES 202 915-4-4 [4], clauses 6.3 and6.4

Precondition: IUT capable of invoking **superviseVolumeReq()**

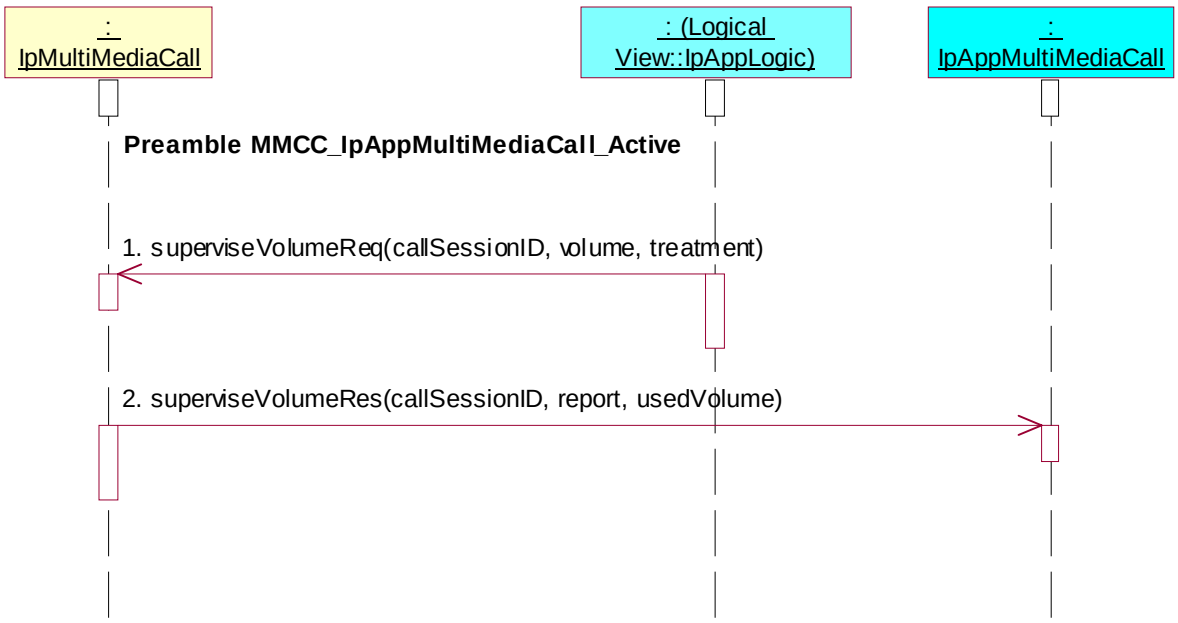
Preamble: **MMCC_IpAppMultiMediaCall_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseVolumeReq()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID, volume, treatment

NOTE: Between these two method sequences, the IUT may need to be triggered to complete establishment of the call to both parties, in order to justify the Tester's calling of a superviseVolumeRes() method.

2. Method call **superviseVolumeRes()**
   Parameters: callSessionID, report, usedVolume
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCall_18

Summary: supervise call with granted volume, unsuccessful

Reference: ES 202 915-4-4 [4], clauses 6.3 and 6.4

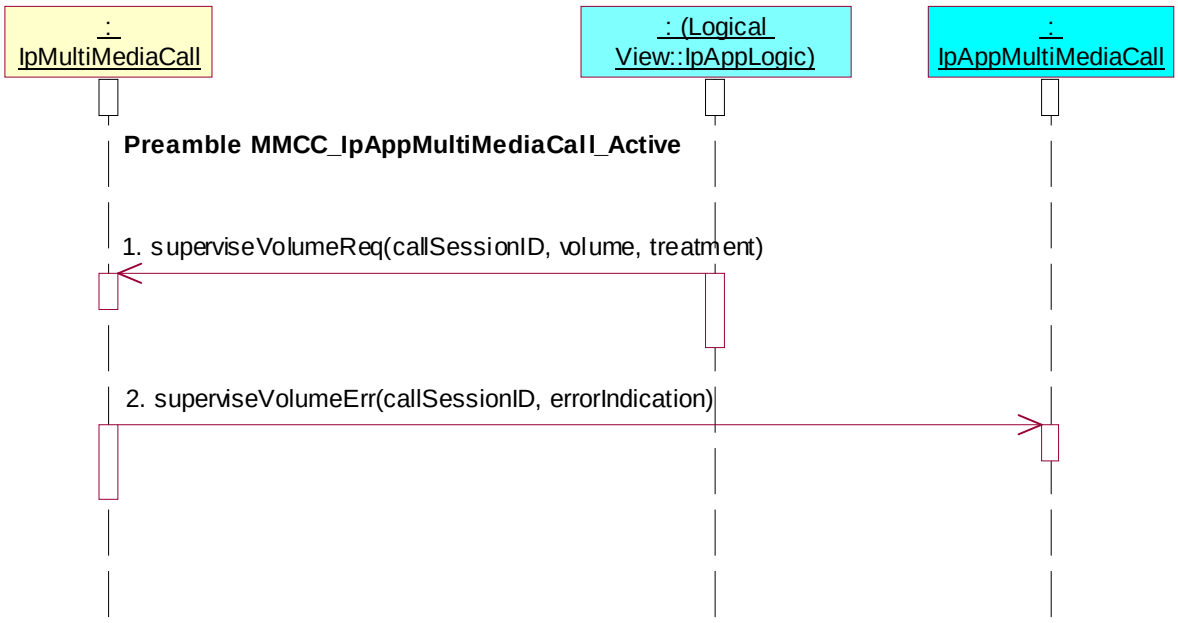
Precondition: IUT capable of invoking **superviseVolumeReq()**

Preamble: **MMCC_IpAppMultiMediaCall_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseVolumeReq()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID, volume, treatment

2. Method call **superviseVolumeErr()**
   Parameters: callSessionID, errorIndication
   Check: no exception is returned



7.2.3.2.3 Released state

Precondition: IUT capable of invoking **createCall()** and **createCallLeg()**

or IUT capable of invoking **createCall()** and **createAndRouteCallLegReq()**

or IUT capable of invoking **createNotification()**

and IUT capable of invoking **release()**

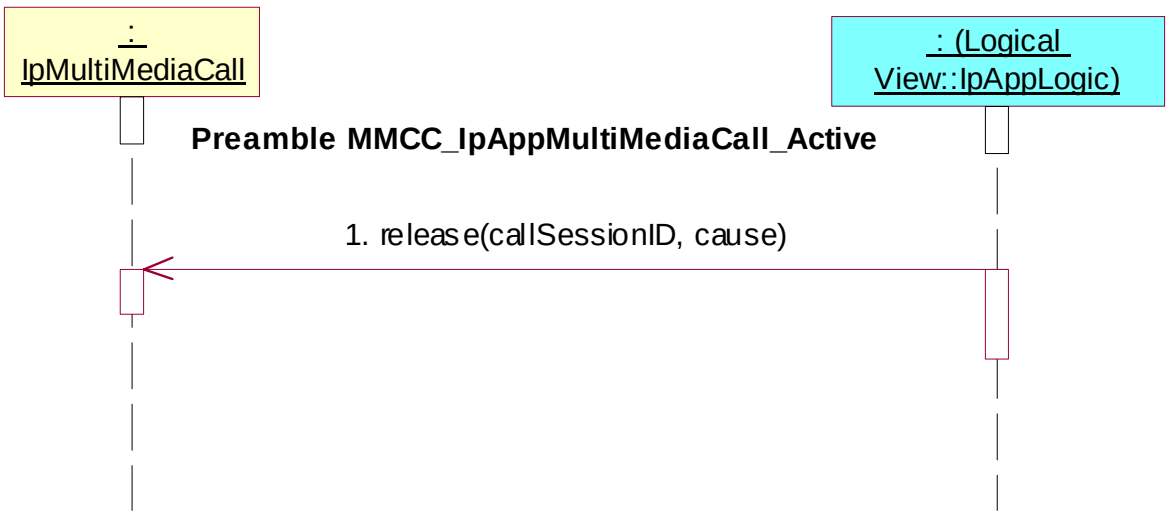
Preamble MMCC_IpAppMultiMediaCall_Released

Reference: ES 202 915-4-3 [3], clause 7.2.3

Pre-preamble: **MMCC_IpAppMultiMediaCall_Active**

Preamble Sequence:

1. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID, cause



Test MMCC_IpAppMultiMediaCall_19

Summary: request call leg information

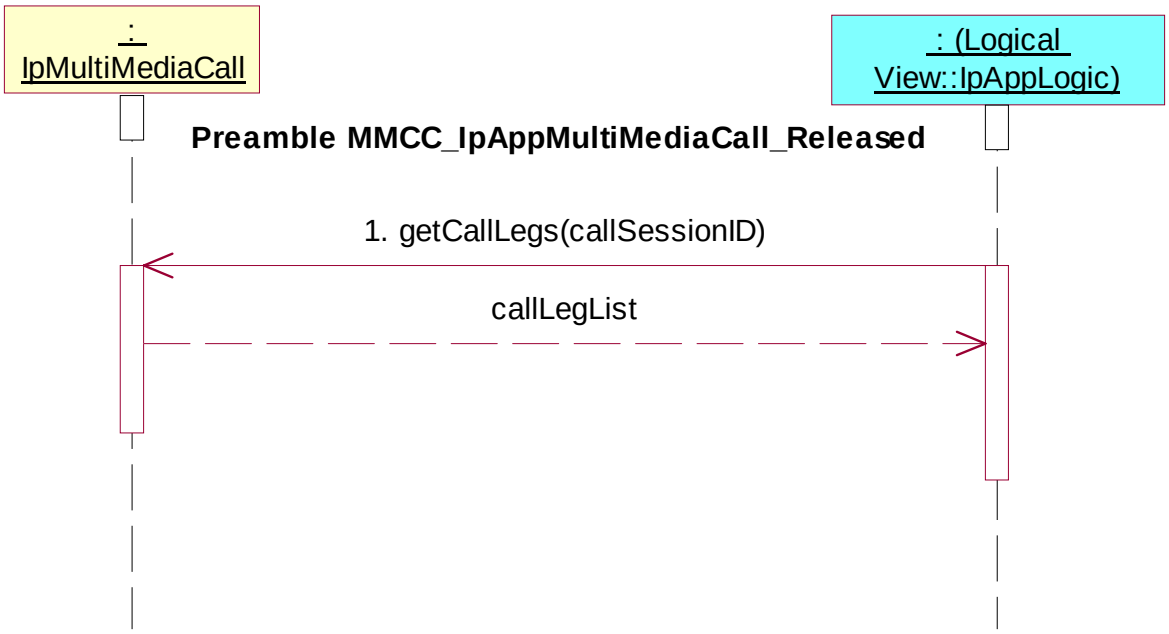
Reference: ES 202 915-4-3 [3], clause 7.2.3

Precondition: IUT capable of invoking **getCallLegs()**

Preamble: **MMCC_IpAppMultiMediaCall_Released**

Test Sequence:

1. Triggered Action: cause IUT to call **getCallLegs()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID



Test MMCC_IpAppMultiMediaCall_20

Summary: indication of termination of call

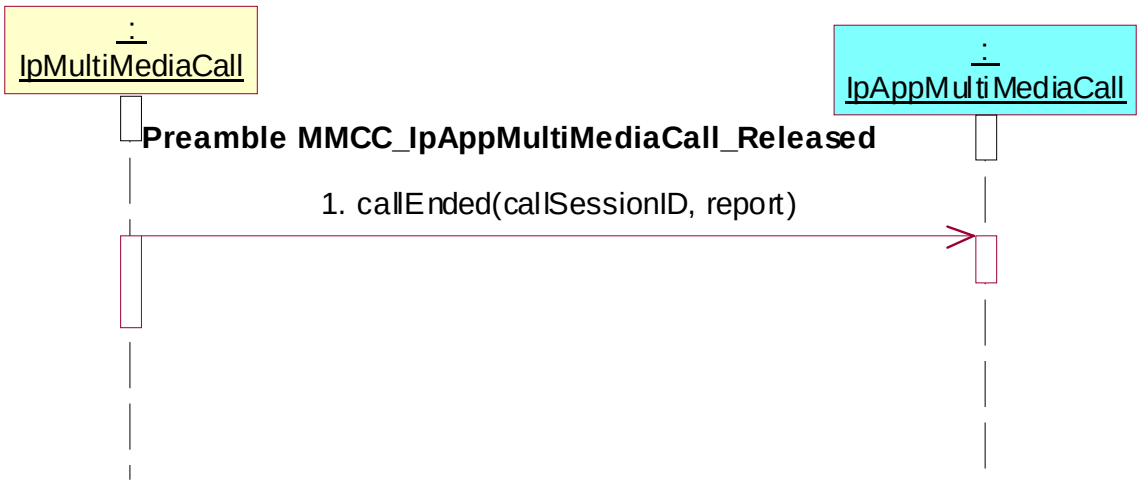
Reference: ES 202 915-4-3 [3], clause 7.2.3

Precondition: **callEnded()** implemented

Preamble: **MMCC_IpAppMultiMediaCall_Released**

Test Sequence:

1. Method call **callEnded()**
   Parameters: callSessionID, report
   Check: no exception is returned



7.2.3.3 IpAppMultiMediaCallLeg

Applications need not be capable of performing each of the sequences below, even if they support the methods indicated below.

Reference: ES 202 915-4-3 [3], clause 7.3, ES 202 915-4-4 [4] clauses 6.5 and 6.6

7.2.3.3.1 Originating Leg

Reference: ES 202 915-4-3 [3], clause 7.3.1

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

7.2.3.3.1.1 Initiating state

Preamble MMCC_IpAppMultiMediaCallLeg_Initiating

Reference: ES 202 915-4-3 [3], clause 7.3.1.1

Pre-preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiMediaCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiMediaCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Preamble Sequence:

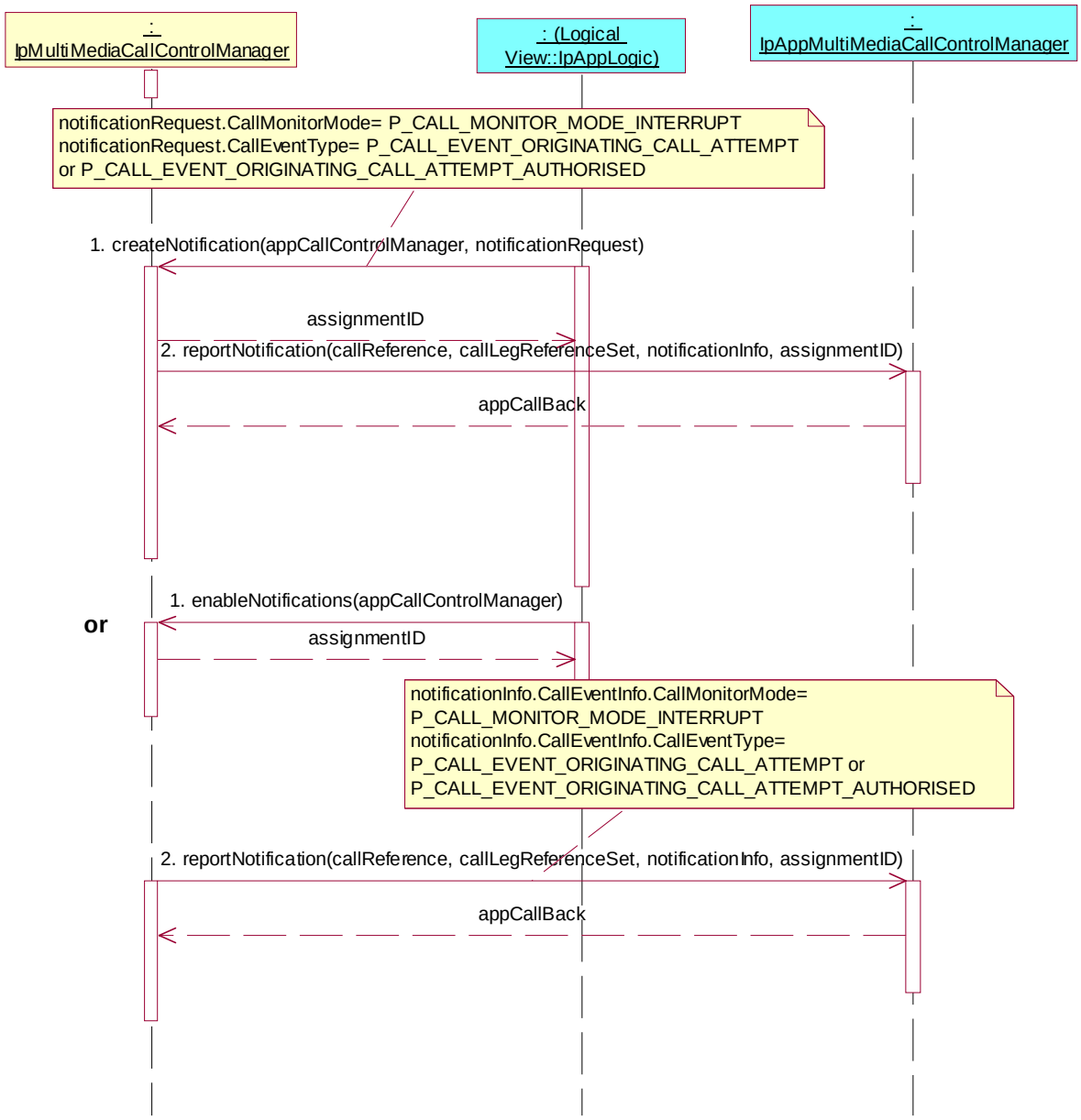
1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest
   notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   notificationRequest.CallEventType= P_CALL_EVENT_ORIGINATING_CALL_ATTEMPT or P_CALL_EVENT_ORIGINATING_CALL_ATTEMPT_AUTHORISED

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   Check: valid value of TpAppMultiPartyCallBack is returned

or

1. Triggered Action: cause IUT to call **enableNotifications()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCallControlManager

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   notificationInfo.CallEventInfo.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   notificationInfo.CallEventInfo.CallEventType=
   P_CALL_EVENT_ORIGINATING_CALL_ATTEMPT or
   P_CALL_EVENT_ORIGINATING_CALL_ATTEMPT_AUTHORISED
   Check: valid value of TpAppMultiMediaCallBack is returned



Test MMCC_IpAppMultiMediaCallLeg_01

Summary: request reference of call related to call leg

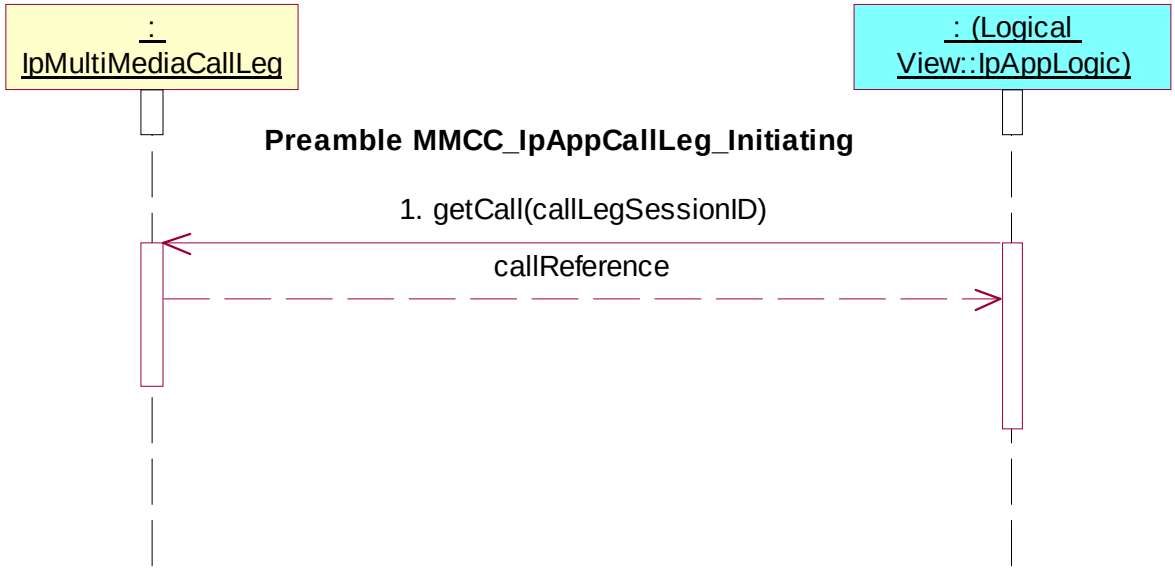
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **getCall()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Initiating**

Test Sequence:

1. Triggered Action: cause IUT to call **getCall()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID



Test MMCC_IpAppMultiMediaCallLeg_02

Summary: continue processing of call leg

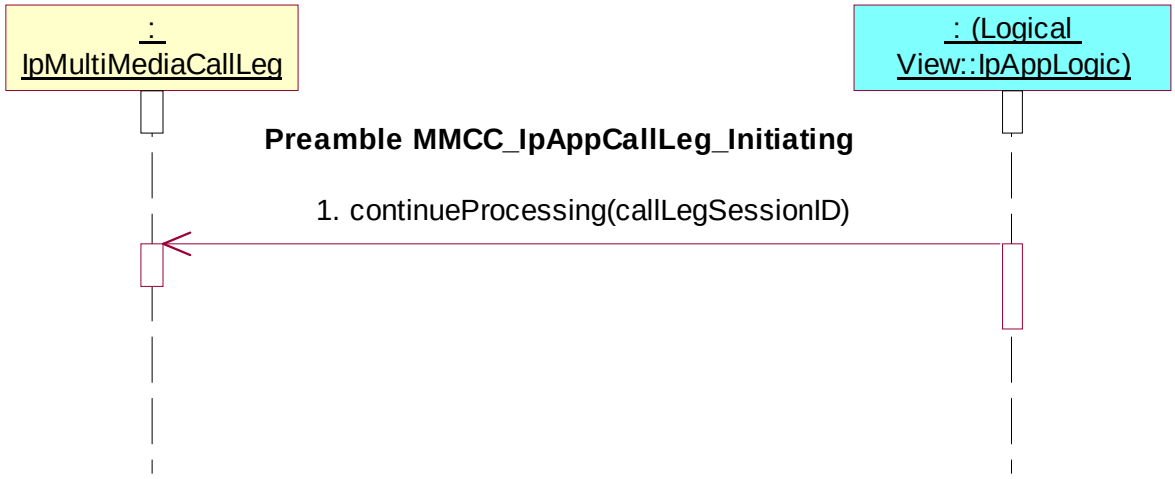
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **continueProcessing()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Initiating**

Test Sequence:

1. Triggered Action: cause IUT to call **continueProcessing()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID



Test MMCC_IpAppMultiMediaCallLeg_03

Summary: release call leg

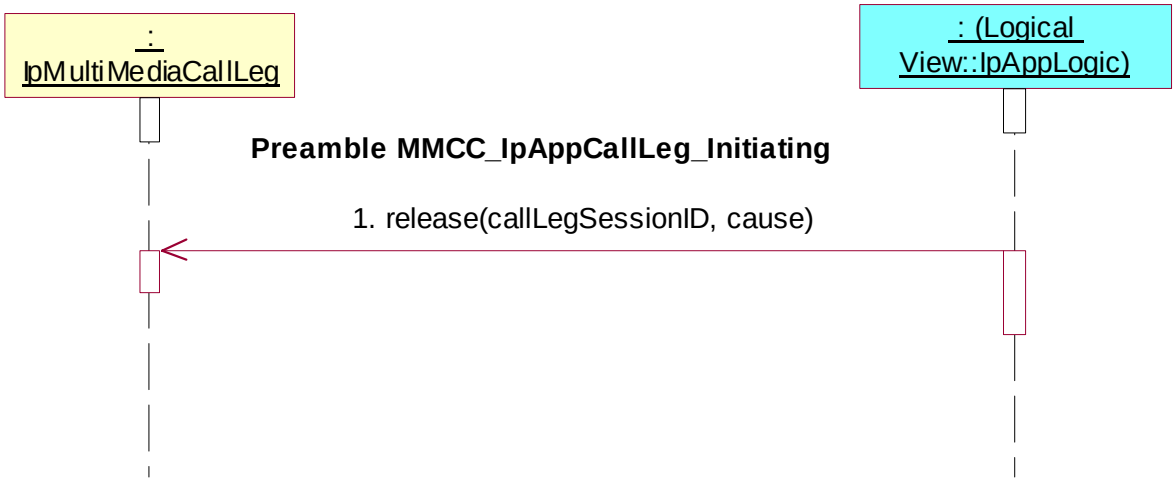
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **release()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Initiating**

Test Sequence:

1. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, cause



Test MMCC_IpAppMultiMediaCallLeg_04

Summary: de-assign call leg

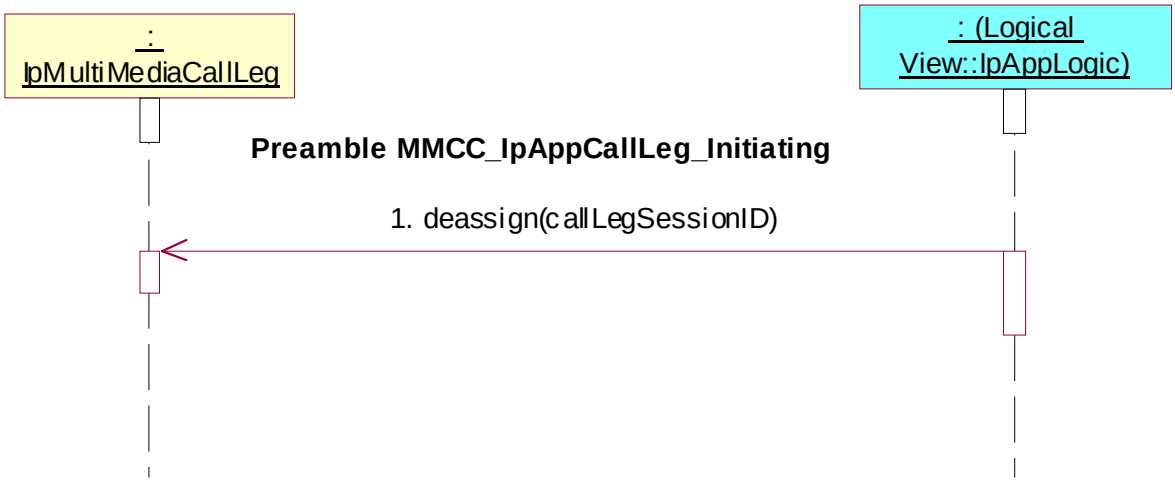
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **deassign()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Initiating**

Test Sequence:

1. Triggered Action: cause IUT to call **deassign()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID



Test MMCC_IpAppMultiMediaCallLeg_05

Summary: change or clear event criteria

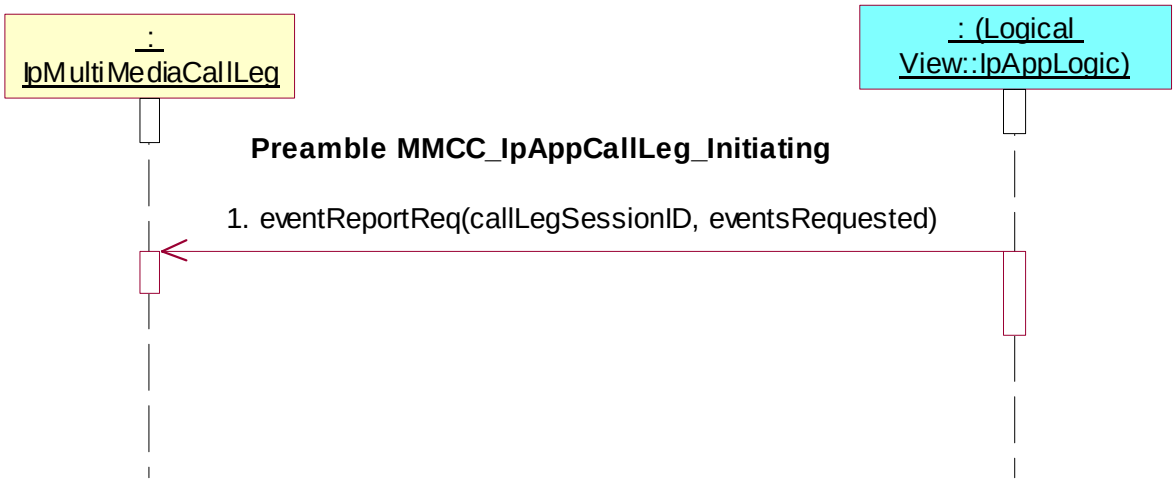
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **eventReportReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Initiating**

Test Sequence:

1. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, eventsRequested



Test MMCC_IpAppMultiMediaCallLeg_06

Summary: change or clear event criteria, unsuccessful

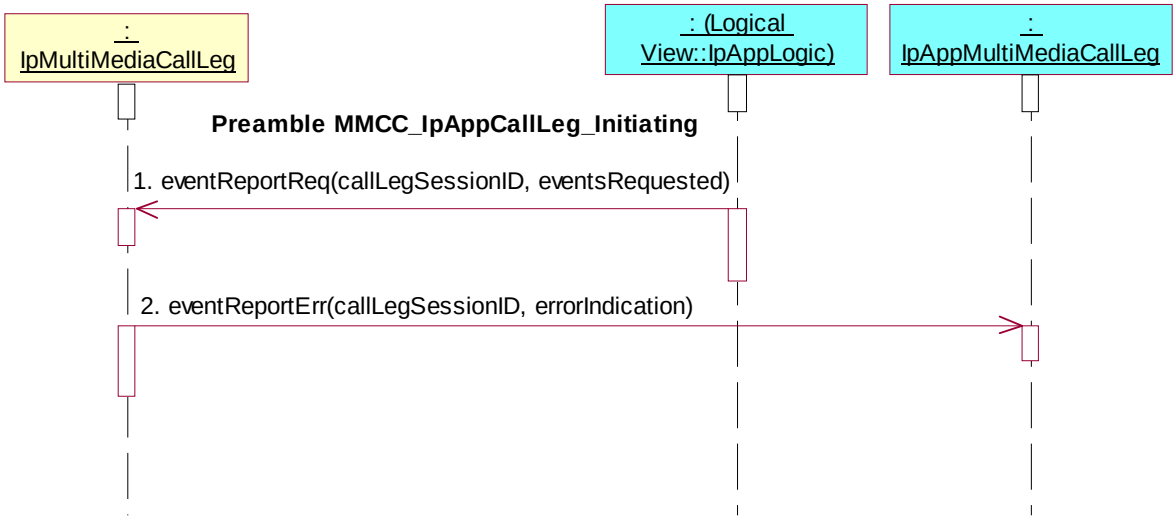
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **eventReportReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Initiating**

Test Sequence:

1. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, eventsRequested
2. Method call **eventReportErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallLeg_07

Summary: get information about call leg

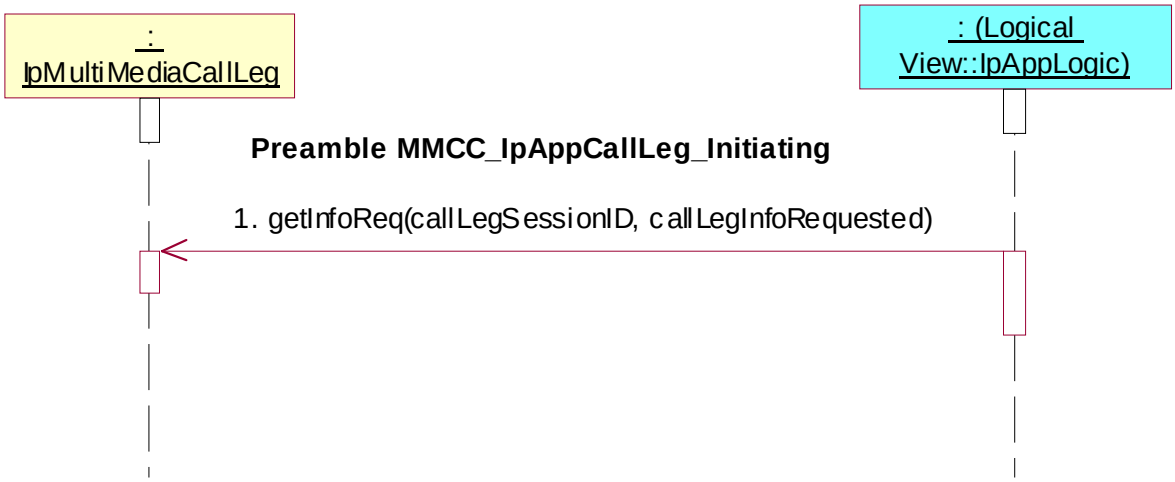
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Initiating**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, callLegInfoRequested



Test MMCC_IpAppMultiMediaCallLeg_08

Summary: get information about call leg, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.1

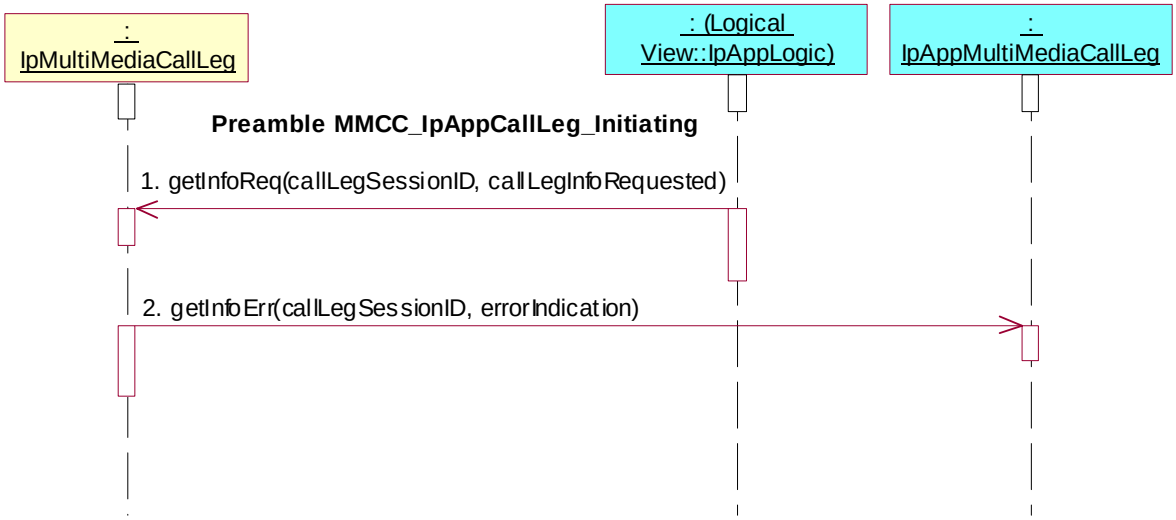
Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Initiating**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, callLegInfoRequested

2. Method call **getInfoErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallLeg_09

Summary: set charge plan for call leg

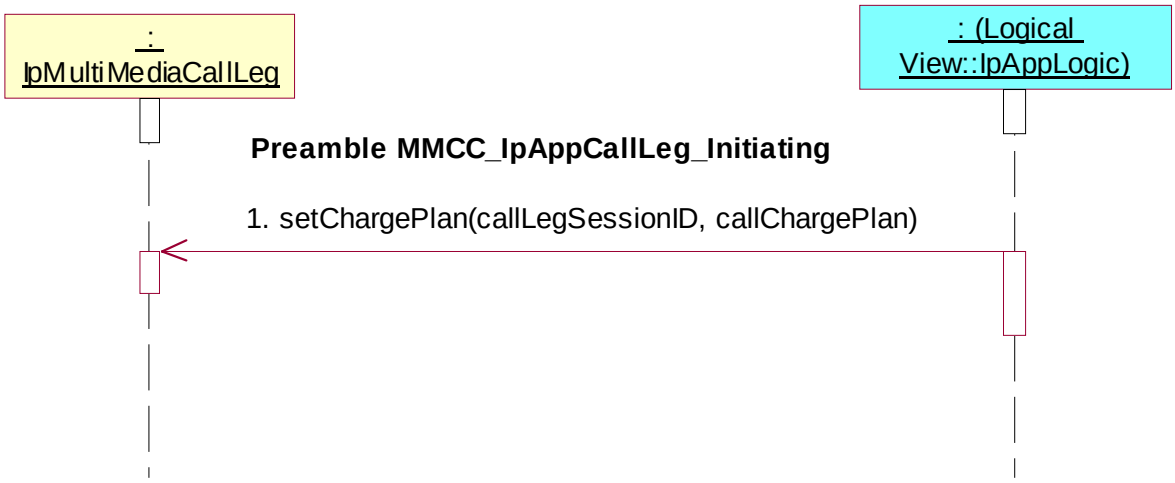
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **setChargePlan()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Initiating**

Test Sequence:

1. Triggered Action: cause IUT to call **setChargePlan()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, callChargePlan



Test MMCC_IpAppMultiMediaCallLeg_10

Summary: allow advice of charge information

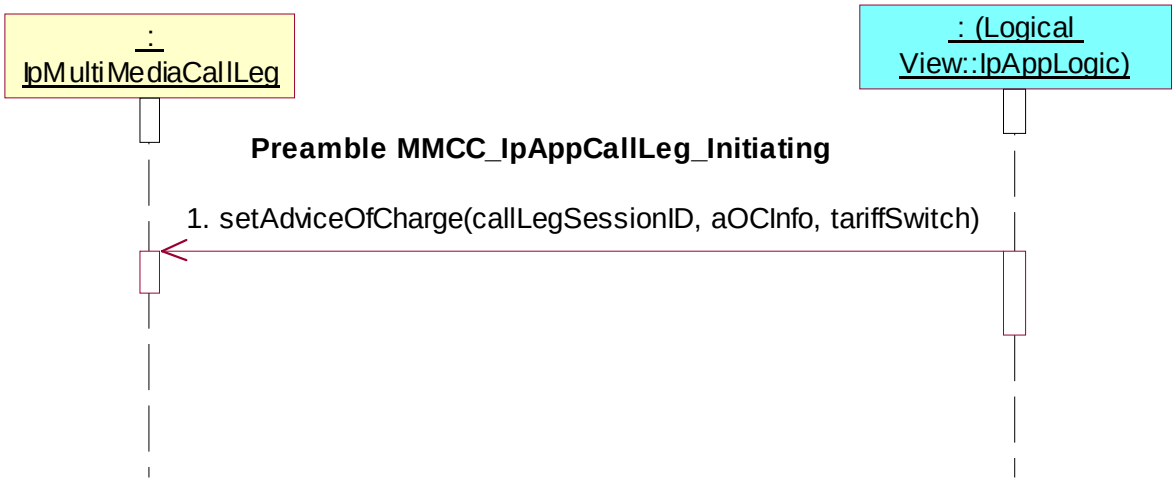
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **setAdviceOfCharge()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Initiating**

Test Sequence:

1. Triggered Action: cause IUT to call **setAdviceOfCharge()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, aOCInfo, tariffSwitch



Test MMCC_IpAppMultiMediaCallLeg_11

Summary: supervise call leg

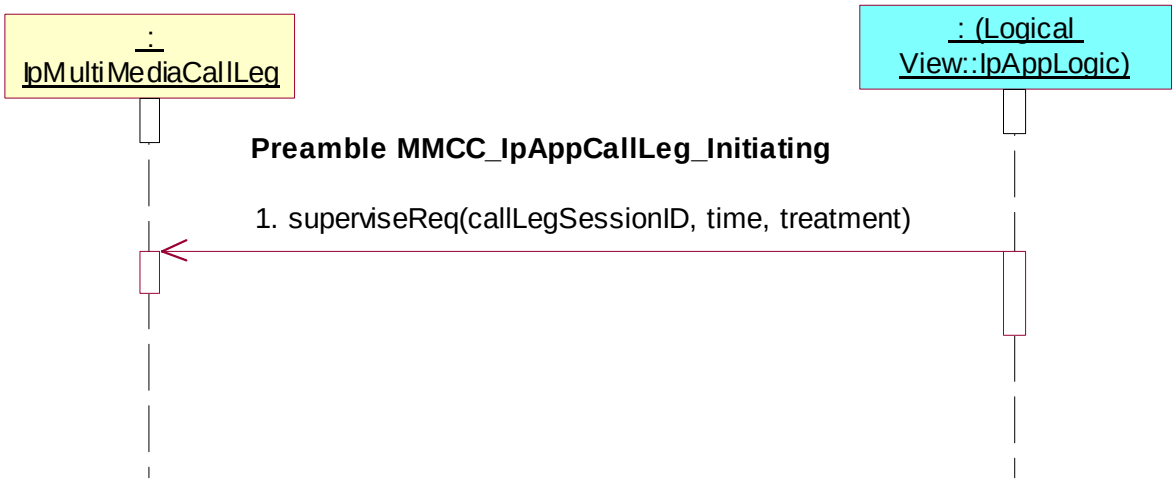
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **superviseReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Initiating**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, time, treatment



Test MMCC_IpAppMultiMediaCallLeg_12

Summary: supervise call leg, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.1

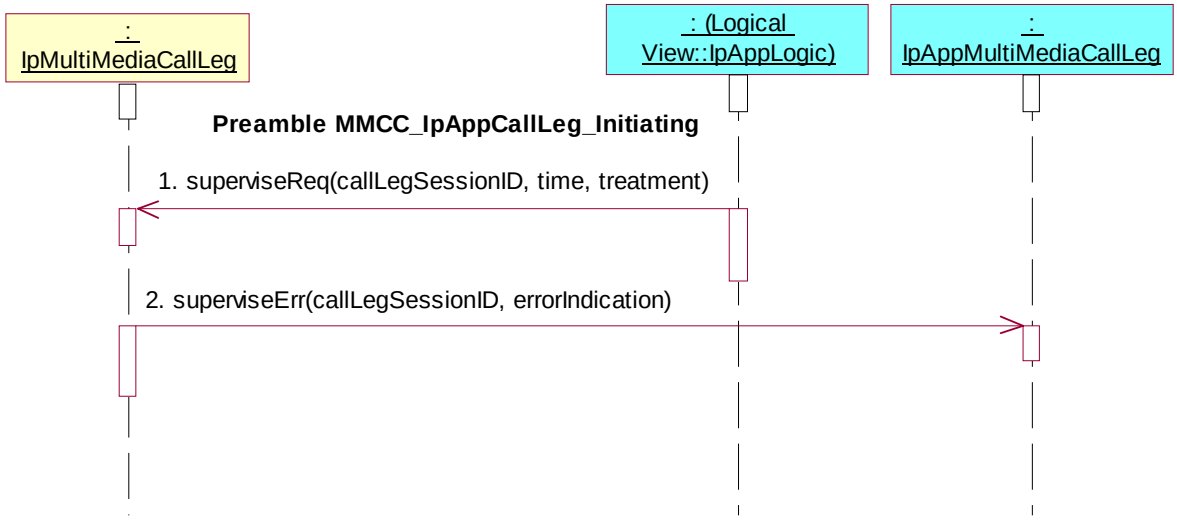
Precondition: IUT capable of invoking **superviseReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Initiating**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, time, treatment

2. Method call **superviseErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



7.2.3.3.1.2 Analysing state

Preamble MMCC_IpAppMultiMediaCallLeg_Analysing

Reference: ES 202 915-4-3 [3], clause 7.3.1.2

Pre-preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiMediaCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiMediaCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Preamble Sequence:

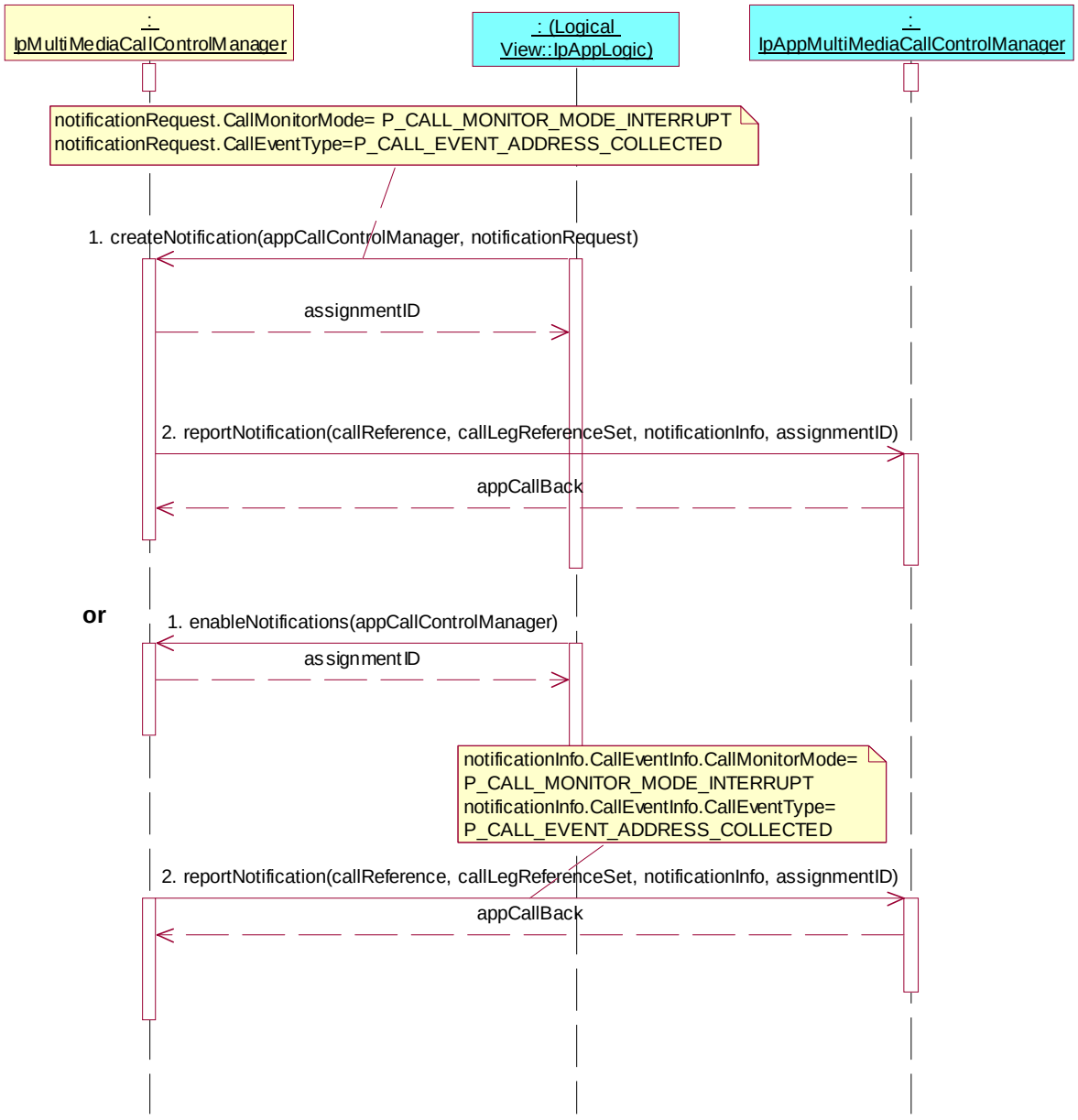
1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest
   notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   notificationRequest.CallEventType= P_CALL_EVENT_ADDRESS_COLLECTED

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   Check: valid value of TpAppMultiPartyCallBack is returned

or

1. Triggered Action: cause IUT to call **enableNotifications()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCallControlManager

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   notificationInfo.CallEventInfo.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   notificationInfo.CallEventInfo.CallEventType= P_CALL_EVENT_ADDRESS_COLLECTED
   Check: valid value of TpAppMultiMediaCallBack is returned



Test MMCC_IpAppMultiMediaCallLeg_13

Summary: attach media, successful

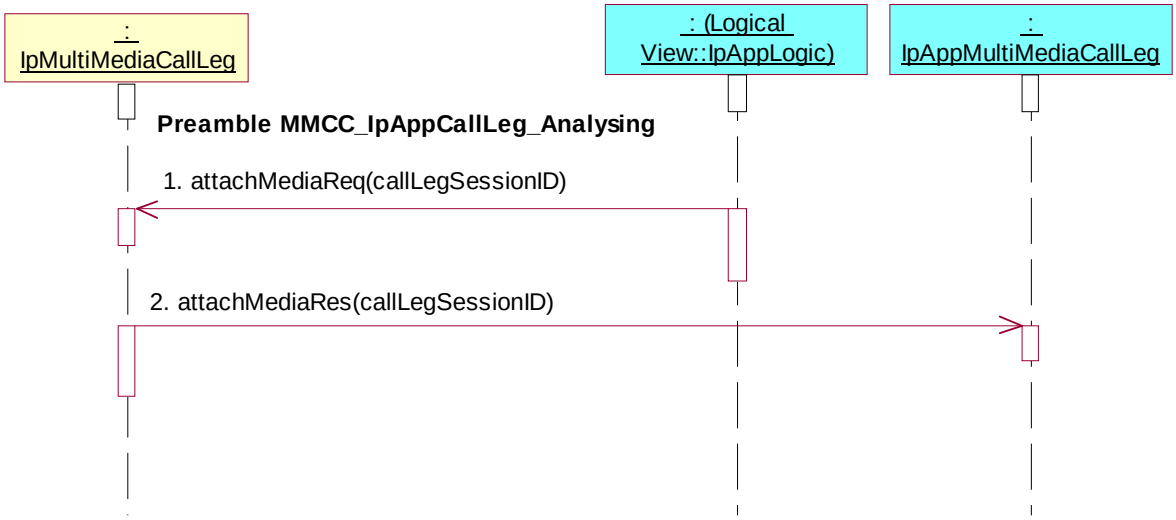
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **attachMediaReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Analysing**

Test Sequence:

1. Triggered Action: cause IUT to call **attachMediaReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID
2. Method call **attachMediaRes()**
   Parameters: callLegSessionID
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallLeg_14

Summary: attach media, unsuccessful

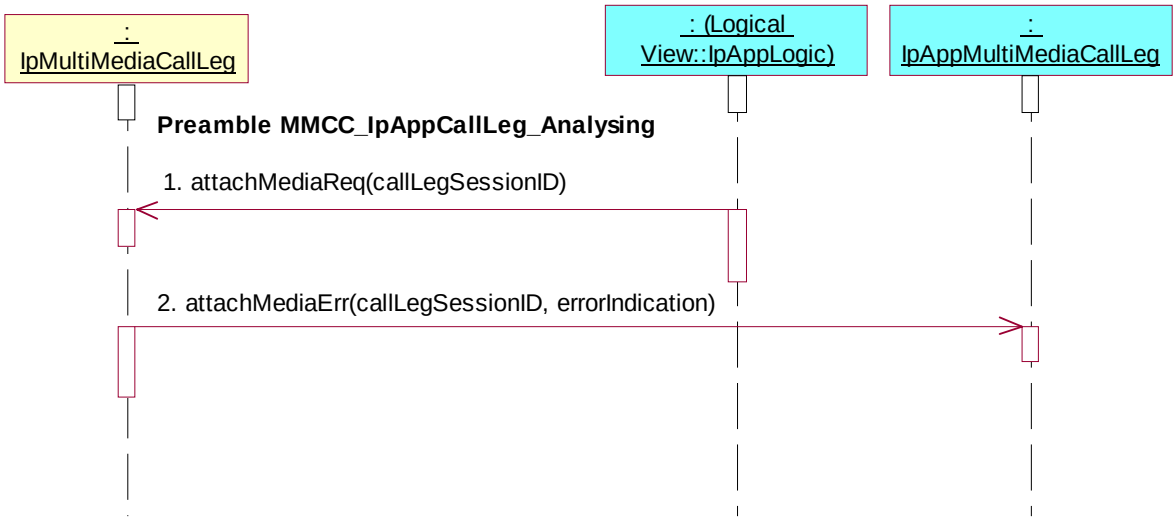
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **attachMediaReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Analysing**

Test Sequence:

1. Triggered Action: cause IUT to call **attachMediaReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID
2. Method call **attachMediaErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallLeg_15

Summary: detach media, successful

Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **detachMediaReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Analysing**

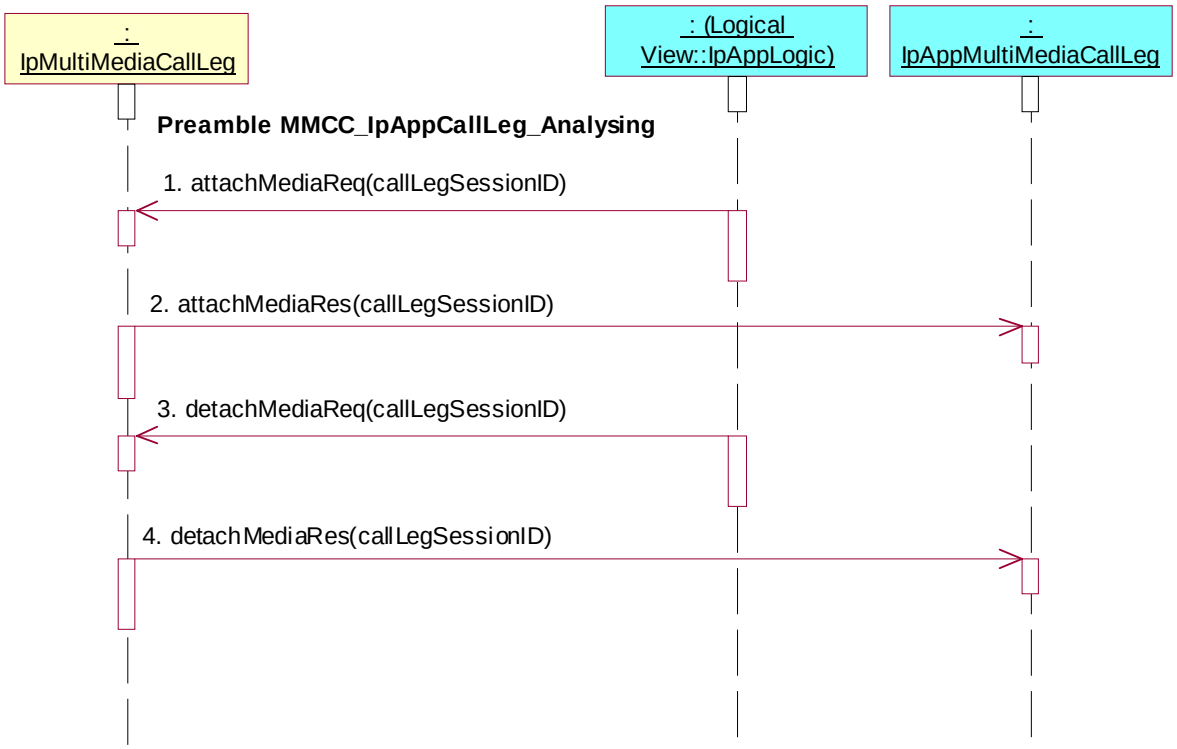
Test Sequence:

1. Triggered Action: cause IUT to call **attachMediaReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID

2. Method call **attachMediaRes()**
   Parameters: callLegSessionID
   Check: no exception is returned

3. Triggered Action: cause IUT to call **detachMediaReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID

4. Method call **detachMediaRes()**
   Parameters: callLegSessionID
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallLeg_16

Summary: detach media, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **detachMediaReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Analysing**

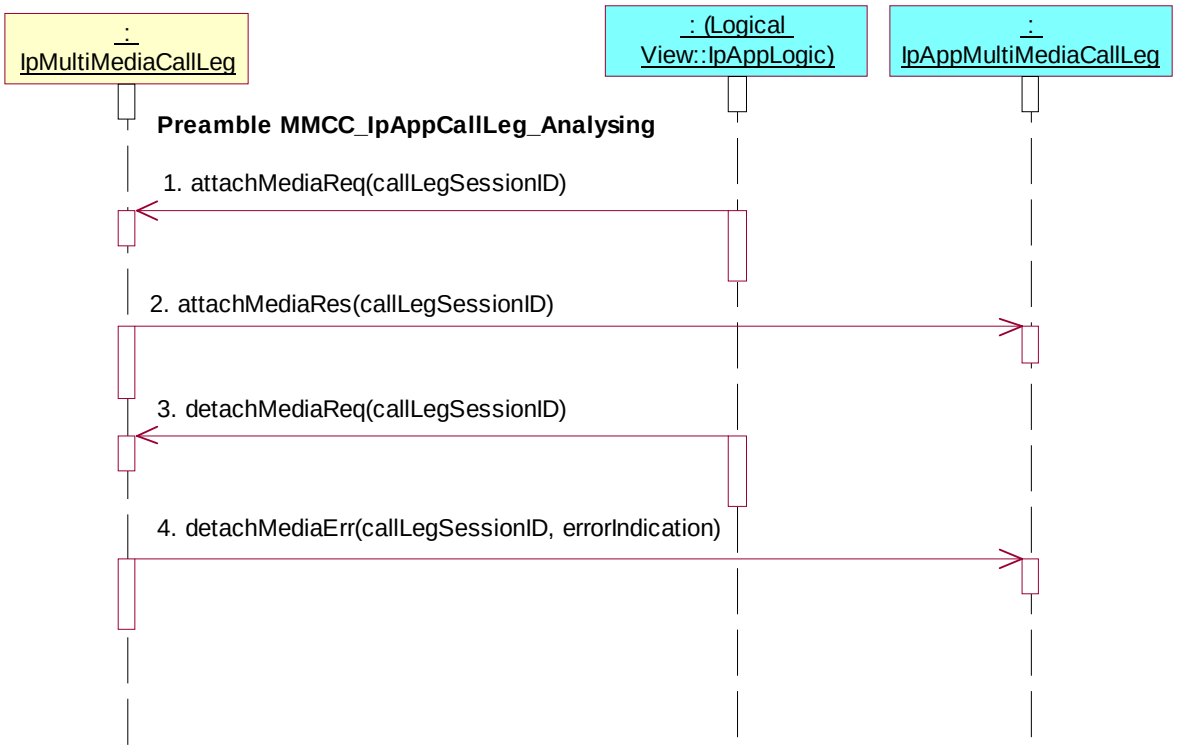
Test Sequence:

1. Triggered Action: cause IUT to call **attachMediaReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID

2. Method call **attachMediaRes()**
   Parameters: callLegSessionID
   Check: no exception is returned

3. Triggered Action: cause IUT to call **detachMediaReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID

4. Method call **detachMediaErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallLeg_17

Summary: request reference of call related to call leg

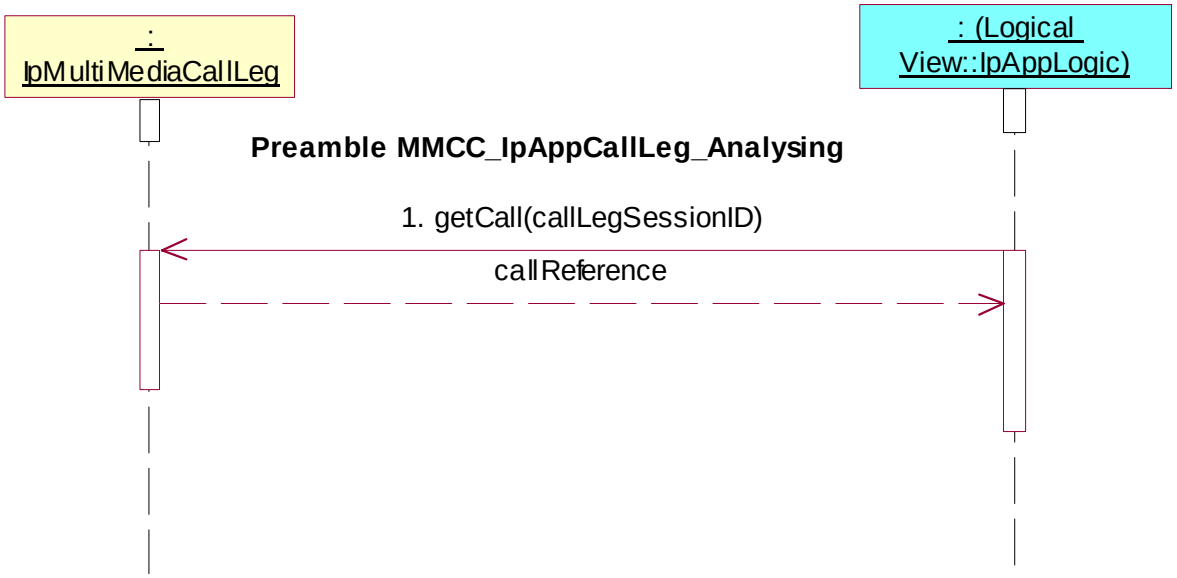
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **getCall()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Analysing**

Test Sequence:

1. Triggered Action: cause IUT to call **getCall()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID



Test MMCC_IpAppMultiMediaCallLeg_18

Summary: continue processing of call leg

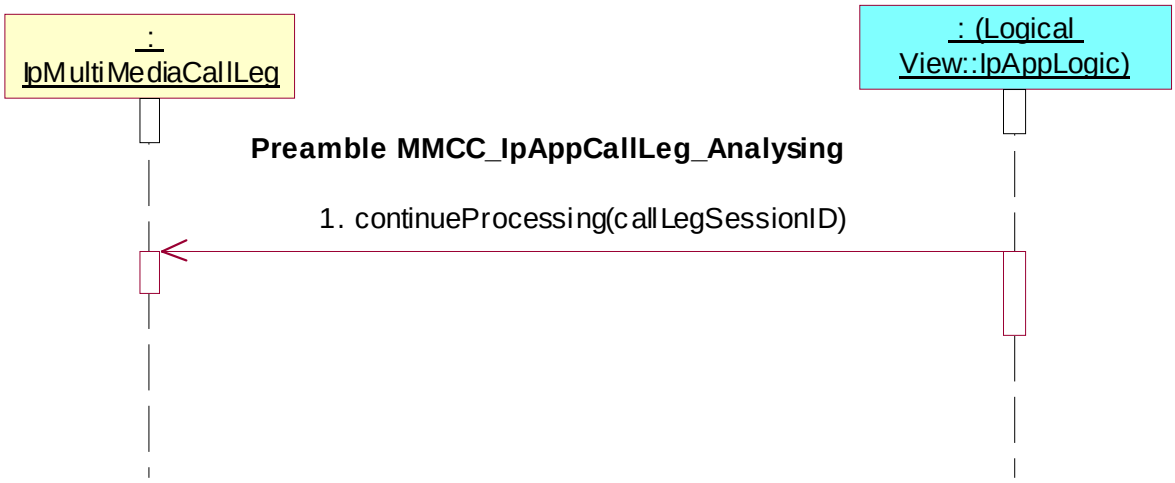
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **continueProcessing()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Analysing**

Test Sequence:

1. Triggered Action: cause IUT to call **continueProcessing()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID



Test MMCC_IpAppMultiMediaCallLeg_19

Summary: release call leg

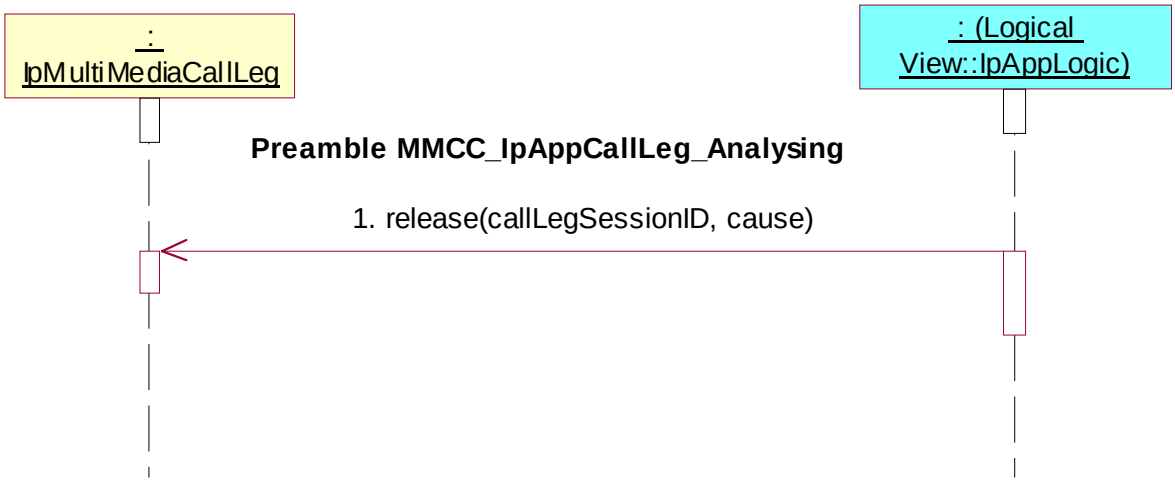
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **release()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Analysing**

Test Sequence:

1. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, cause



Test MMCC_IpAppMultiMediaCallLeg_20

Summary: de-assign call leg

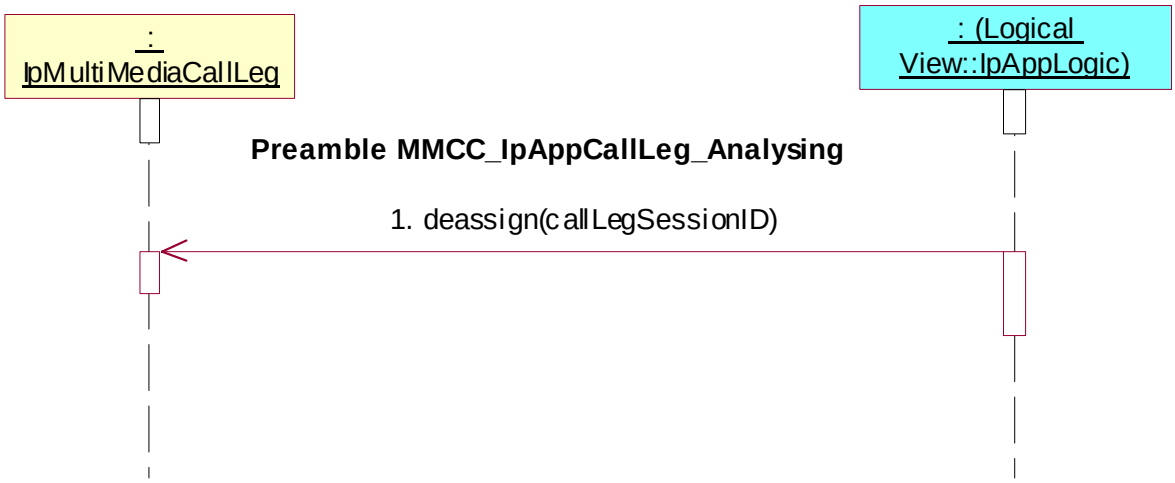
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **deassign()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Analysing**

Test Sequence:

1. Triggered Action: cause IUT to call **deassign()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID



Test MMCC_IpAppMultiMediaCallLeg_21

Summary: change or clear event criteria

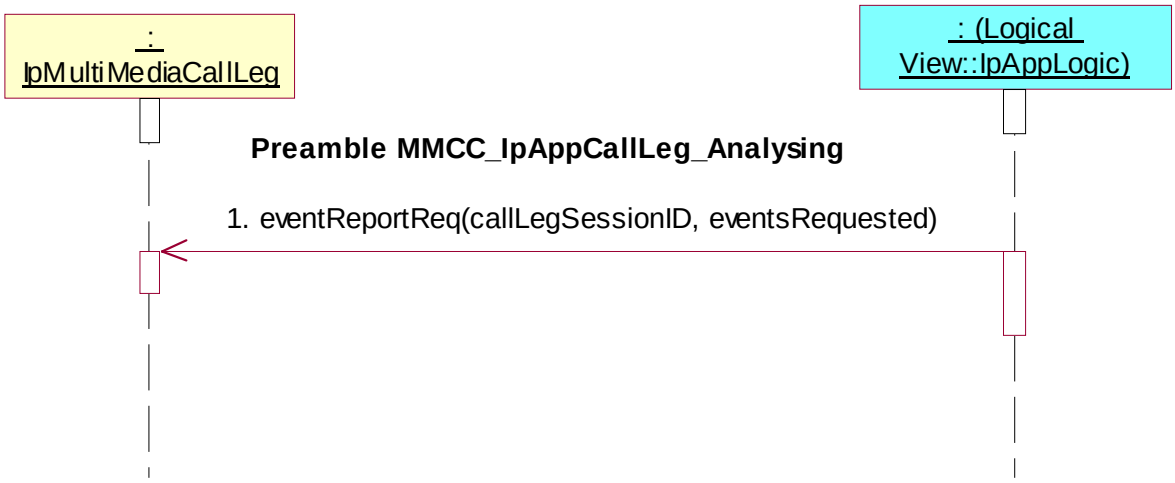
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **eventReportReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Analysing**

Test Sequence:

1. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, eventsRequested



Test MMCC_IpAppMultiMediaCallLeg_22

Summary: change or clear event criteria, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.1

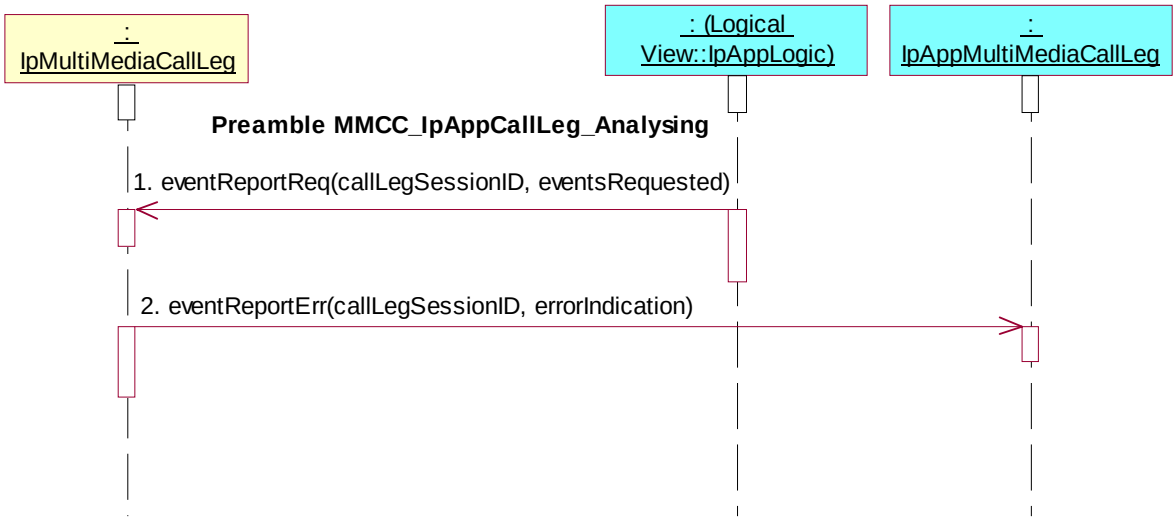
Precondition: IUT capable of invoking **eventReportReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Analysing**

Test Sequence:

1. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, eventsRequested

2. Method call **eventReportErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallLeg_23

Summary: get information about call leg

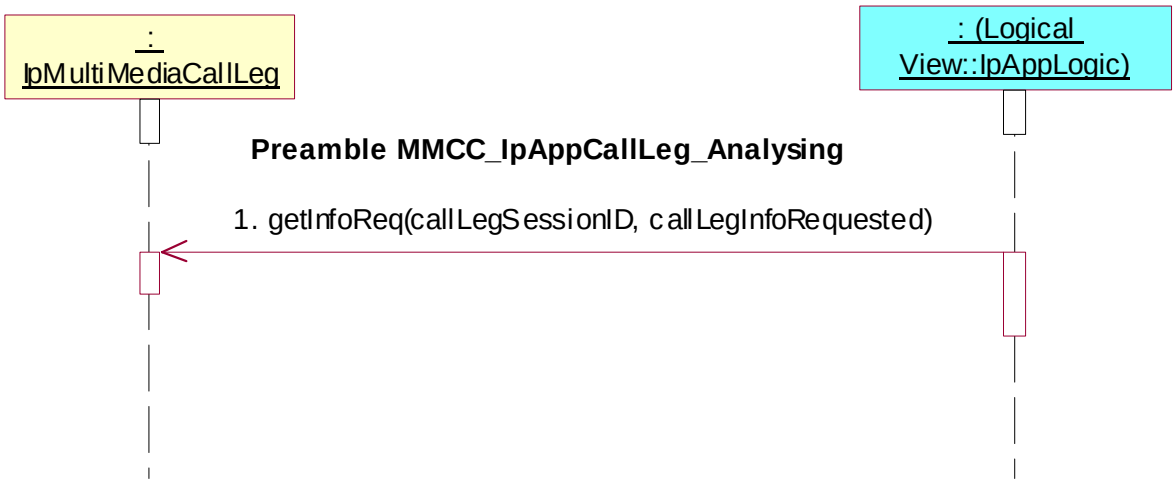
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Analysing**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, callLegInfoRequested



Test MMCC_IpAppMultiMediaCallLeg_24

Summary: get information about call leg, unsuccessful

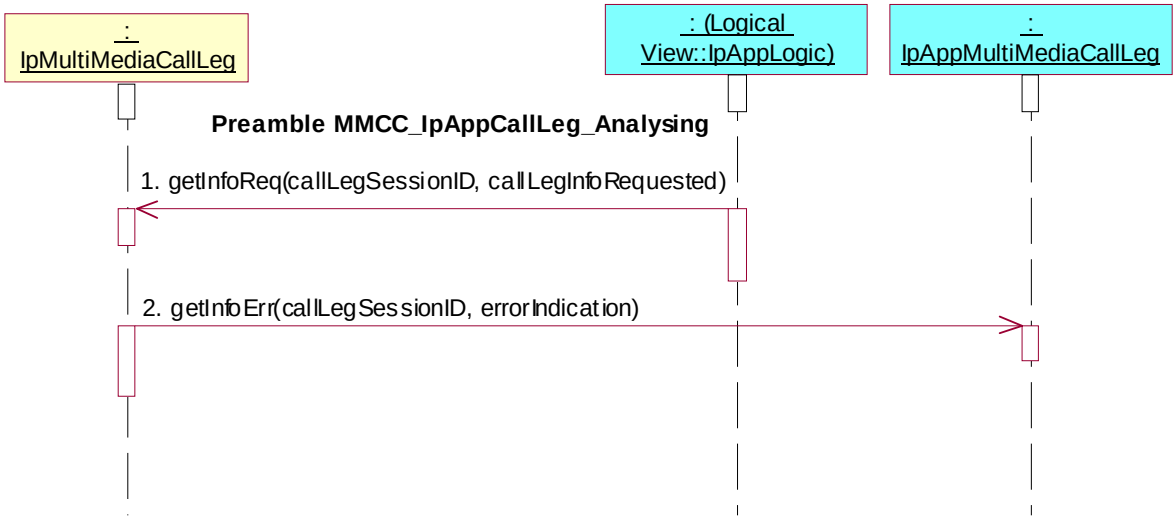
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Analysing**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, callLegInfoRequested
2. Method call **getInfoErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallLeg_25

Summary: set charge plan for call leg

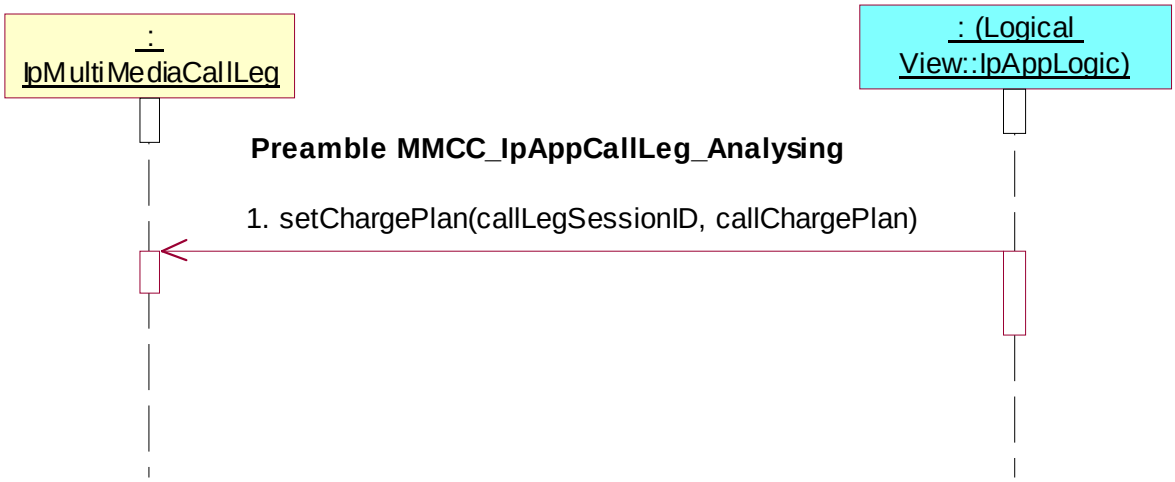
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **setChargePlan()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Analysing**

Test Sequence:

1. Triggered Action: cause IUT to call **setChargePlan()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, callChargePlan



Test MMCC_IpAppMultiMediaCallLeg_26

Summary: allow advice of charge information

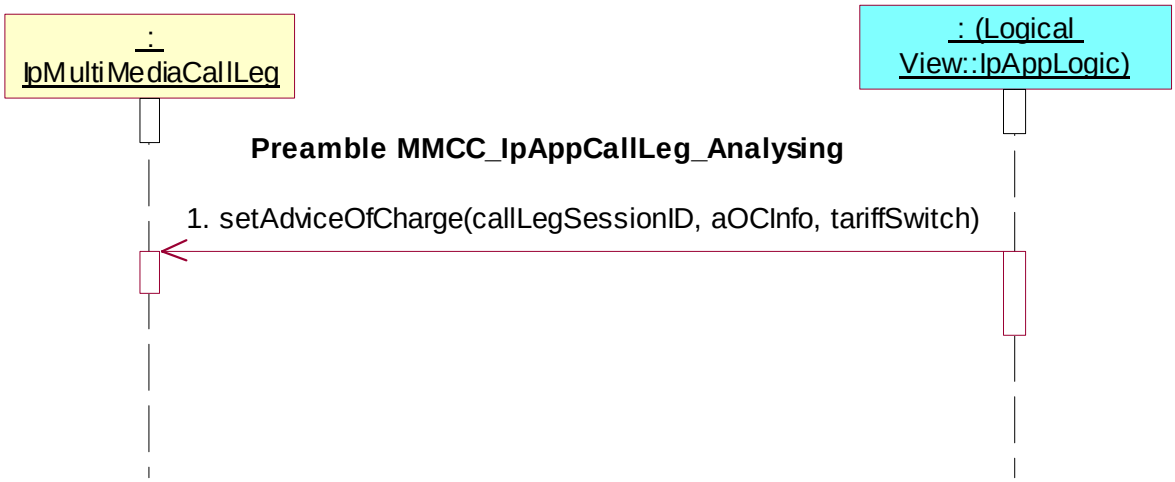
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **setAdviceOfCharge()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Analysing**

Test Sequence:

1. Triggered Action: cause IUT to call **setAdviceOfCharge()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, aOCInfo, tariffSwitch



Test MMCC_IpAppMultiMediaCallLeg_27

Summary: supervise call leg

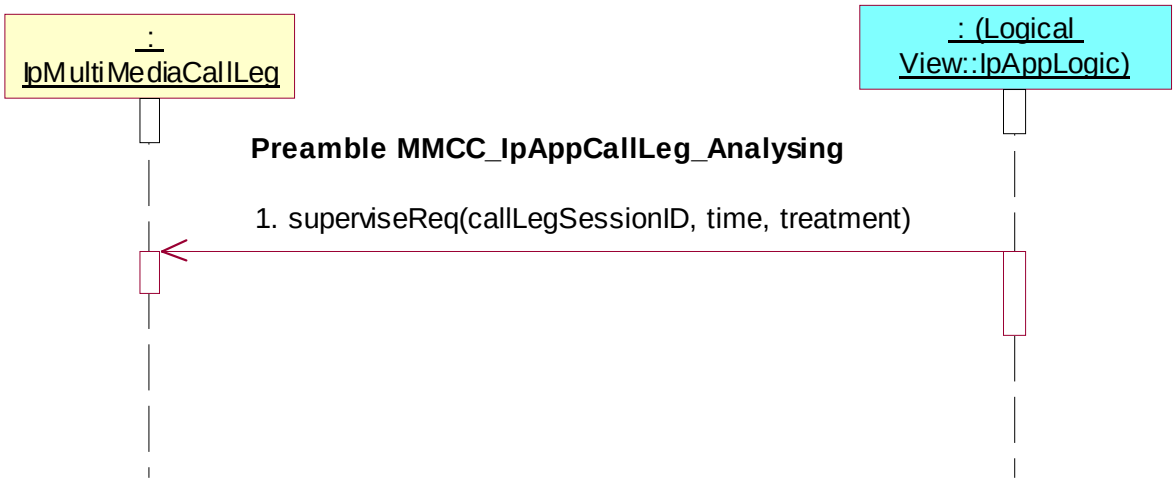
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **superviseReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Analysing**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, time, treatment



Test MMCC_IpAppMultiMediaCallLeg_28

Summary: supervise call leg, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.1

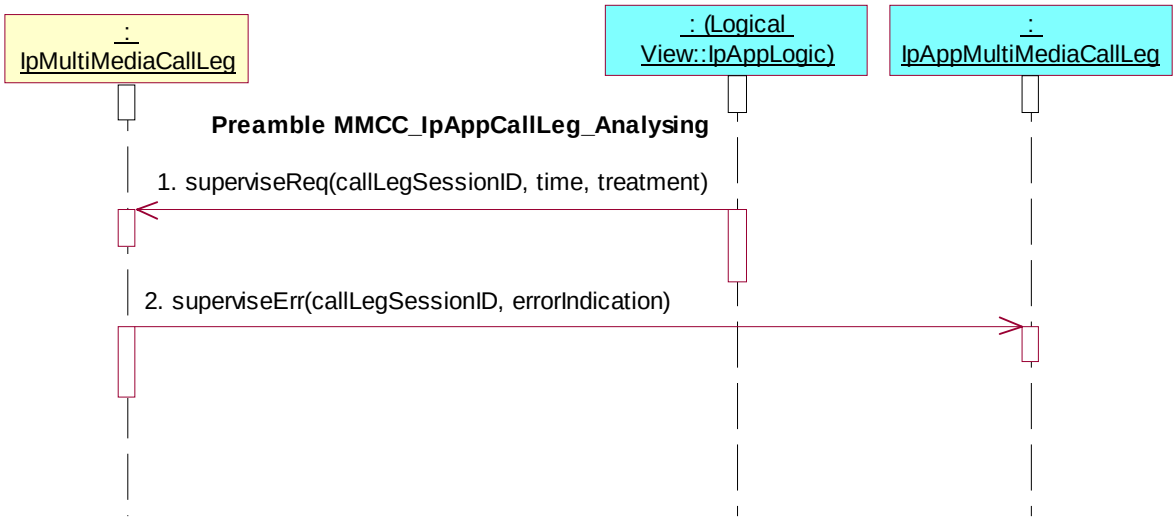
Precondition: IUT capable of invoking **superviseReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Analysing**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, time, treatment

2. Method call **superviseErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



7.2.3.3.1.3 Active state

Preamble MMCC_IpAppMultiMediaCallLeg_Active

Reference: ES 202 915-4-3 [3], clause 7.3.1.3

Pre-preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiMediaCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiMediaCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Preamble Sequence:

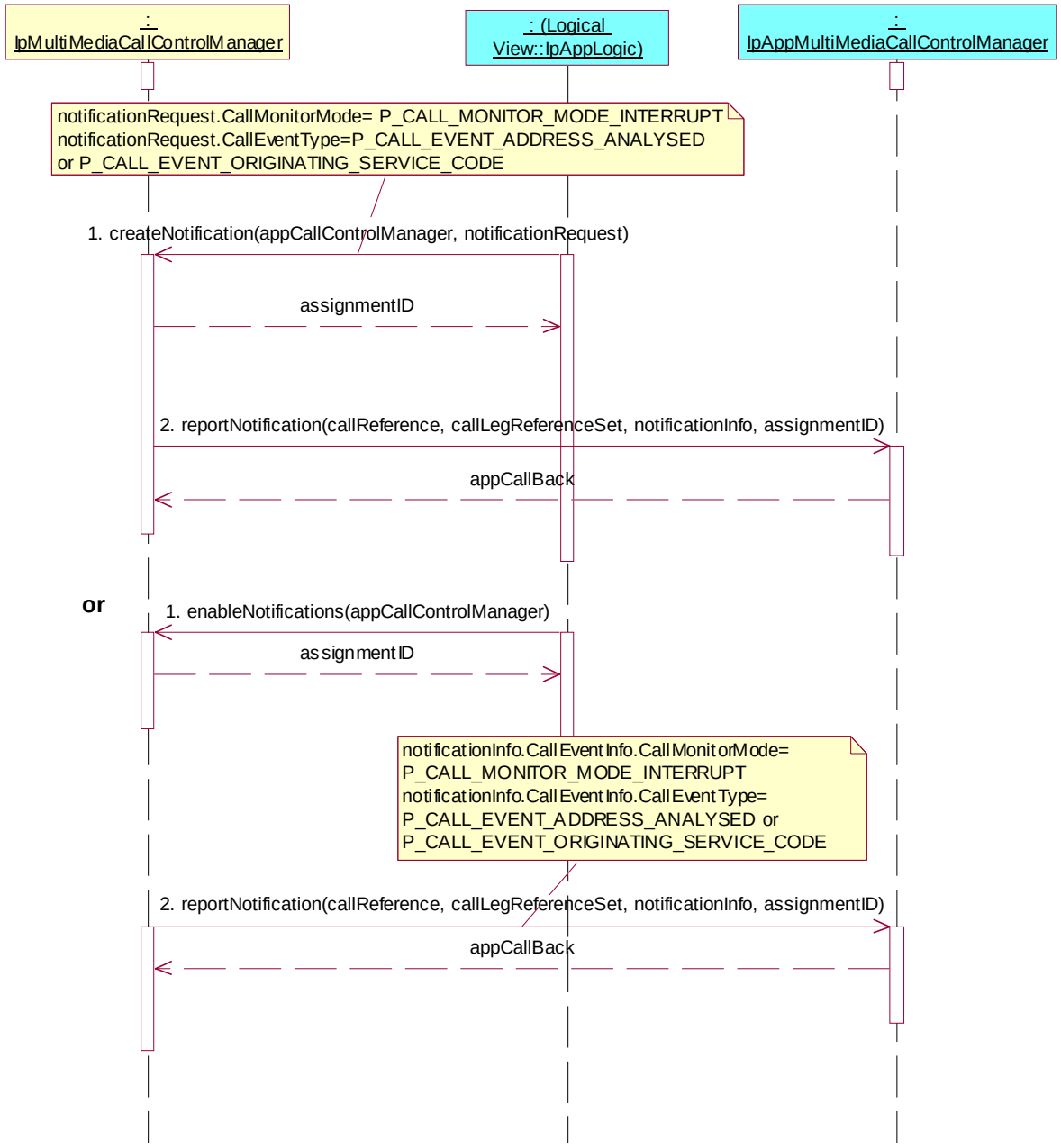
1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest
   notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   notificationRequest.CallEventType= P_CALL_EVENT_ADDRESS_ANALYSED or P_CALL_EVENT_ORIGINATING_SERVICE_CODE

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   Check: valid value of TpAppMultiPartyCallBack is returned

or

1. Triggered Action: cause IUT to call **enableNotifications()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCallControlManager

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   notificationInfo.CallEventInfo.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   notificationInfo.CallEventInfo.CallEventType= P_CALL_EVENT_ADDRESS_ANALYSED or P_CALL_EVENT_ORIGINATING_SERVICE_CODE
   Check: valid value of TpAppMultiMediaCallBack is returned


: IpMultiMediaCallControlManager
: (Logical View::IpAppLogic)
: IpAppMultiMediaCallControlManager
notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
notificationRequest.CallEventType=P_CALL_EVENT_ADDRESS_ANALYSED
or P_CALL_EVENT_ORIGINATING_SERVICE_CODE
1. createNotification(appCallControlManager, notificationRequest)
assignmentID
2. reportNotification(callReference, callLegReferenceSet, notificationInfo, assignmentID)
appCallBack
or
1. enableNotifications(appCallControlManager)
assignmentID
notificationInfo.CallEventInfo.CallMonitorMode=
P_CALL_MONITOR_MODE_INTERRUPT
notificationInfo.CallEventInfo.CallEventType=
P_CALL_EVENT_ADDRESS_ANALYSED or
P_CALL_EVENT_ORIGINATING_SERVICE_CODE
2. reportNotification(callReference, callLegReferenceSet, notificationInfo, assignmentID)
appCallBack

Test MMCC_IpAppMultiMediaCallLeg_29

Summary: attach media, successful

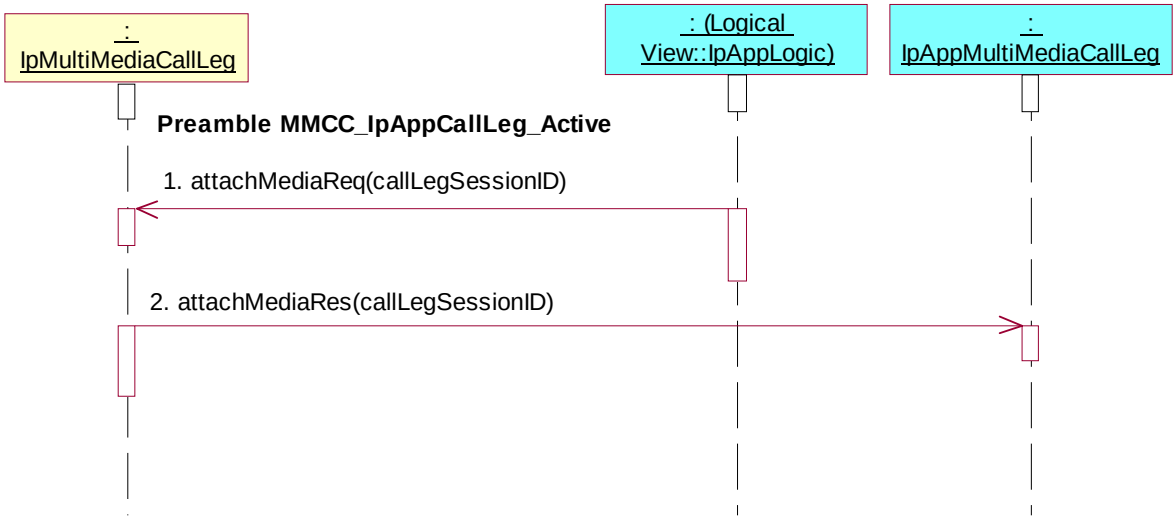
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **attachMediaReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **attachMediaReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID
2. Method call **attachMediaRes()**
   Parameters: callLegSessionID
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallLeg_30

Summary: attach media, unsuccessful

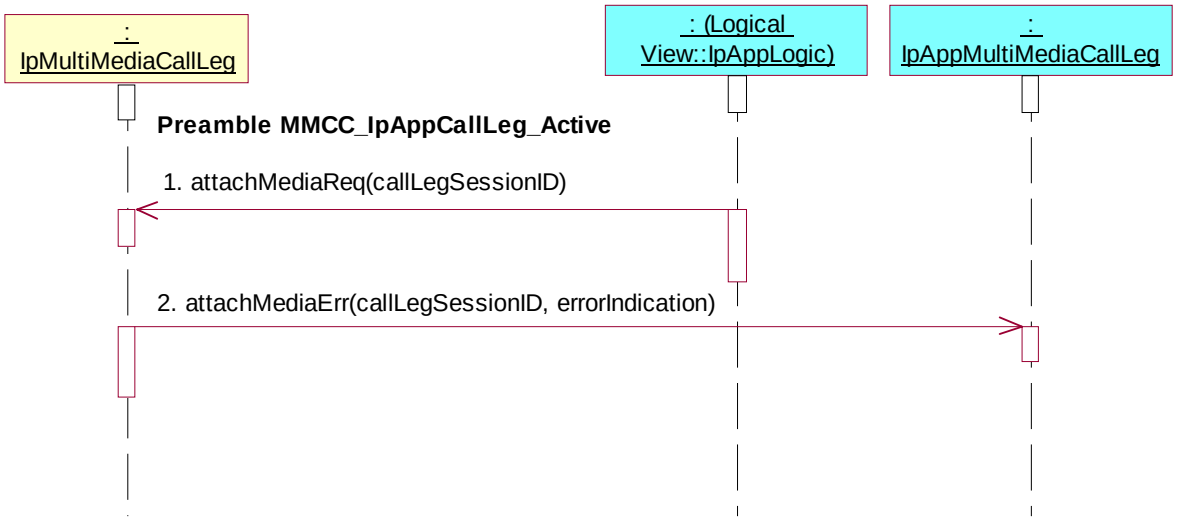
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **attachMediaReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **attachMediaReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID
2. Method call **attachMediaErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallLeg_31

Summary: detach media, successful

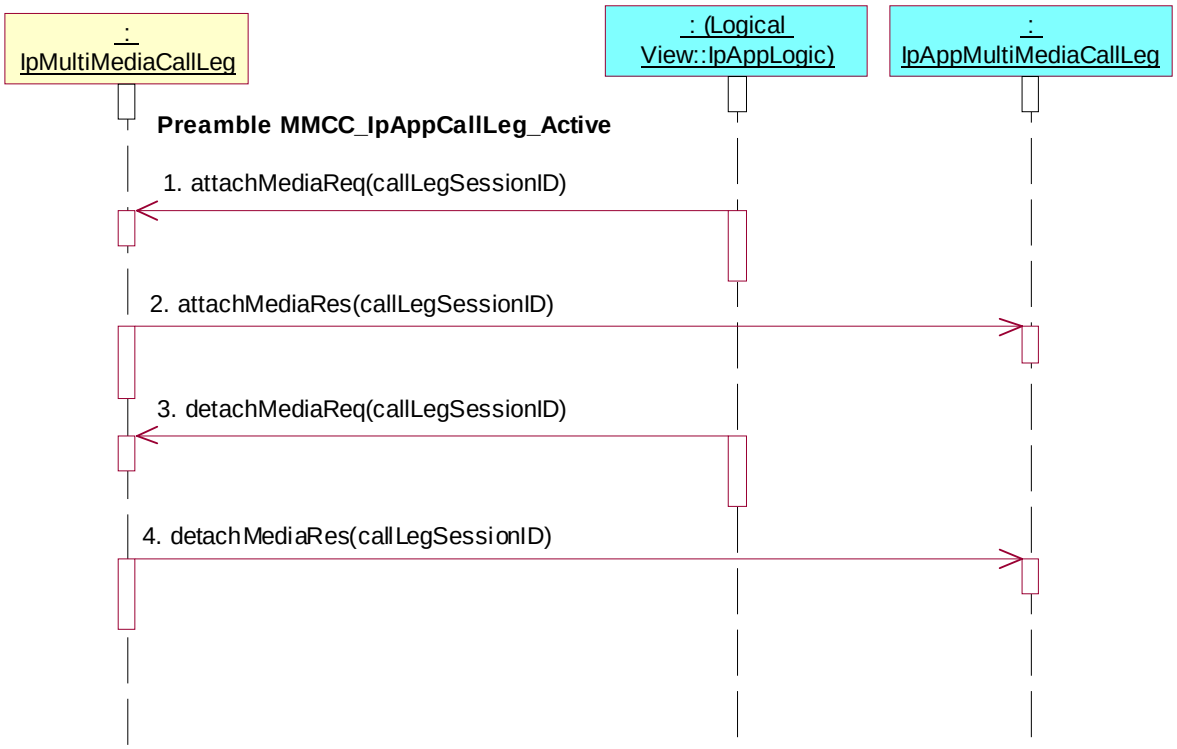
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **detachMediaReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **attachMediaReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID
2. Method call **attachMediaRes()**
   Parameters: callLegSessionID
   Check: no exception is returned
3. Triggered Action: cause IUT to call **detachMediaReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID
4. Method call **detachMediaRes()**
   Parameters: callLegSessionID
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallLeg_32

Summary: detach media, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **detachMediaReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active**

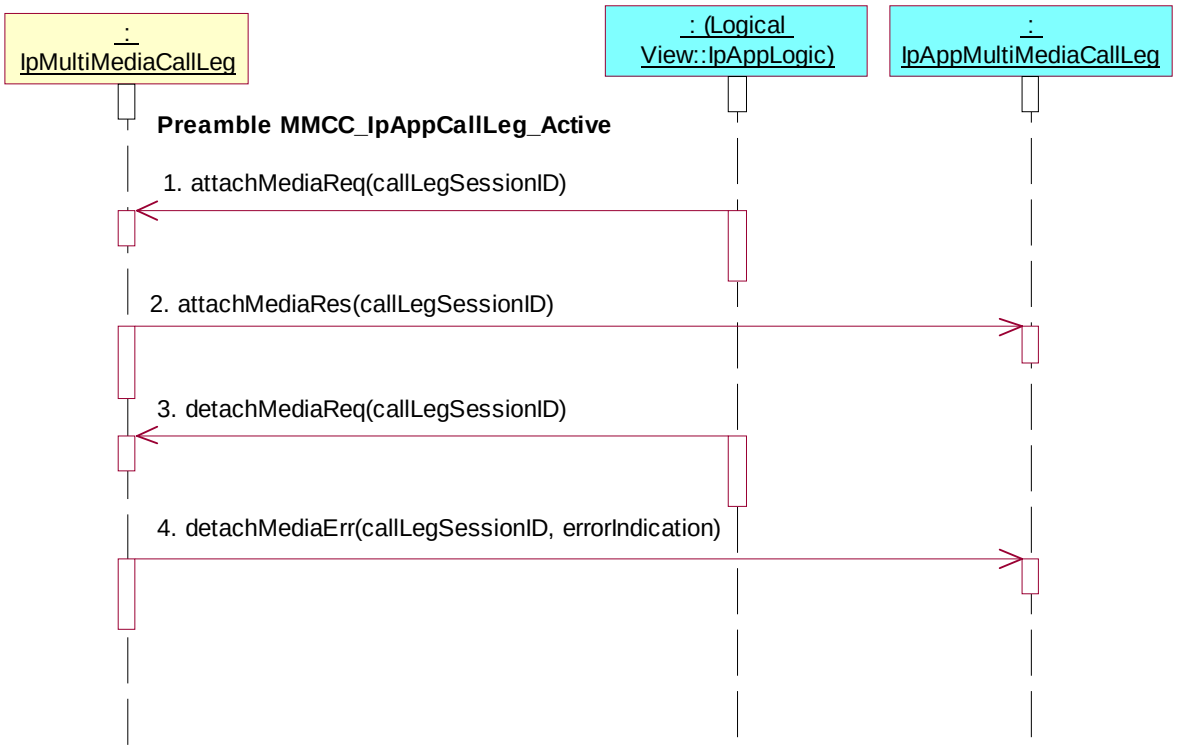
Test Sequence:

1. Triggered Action: cause IUT to call **attachMediaReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID

2. Method call **attachMediaRes()**
   Parameters: callLegSessionID
   Check: no exception is returned

3. Triggered Action: cause IUT to call **detachMediaReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID

4. Method call **detachMediaErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallLeg_33

Summary: request reference of call related to call leg

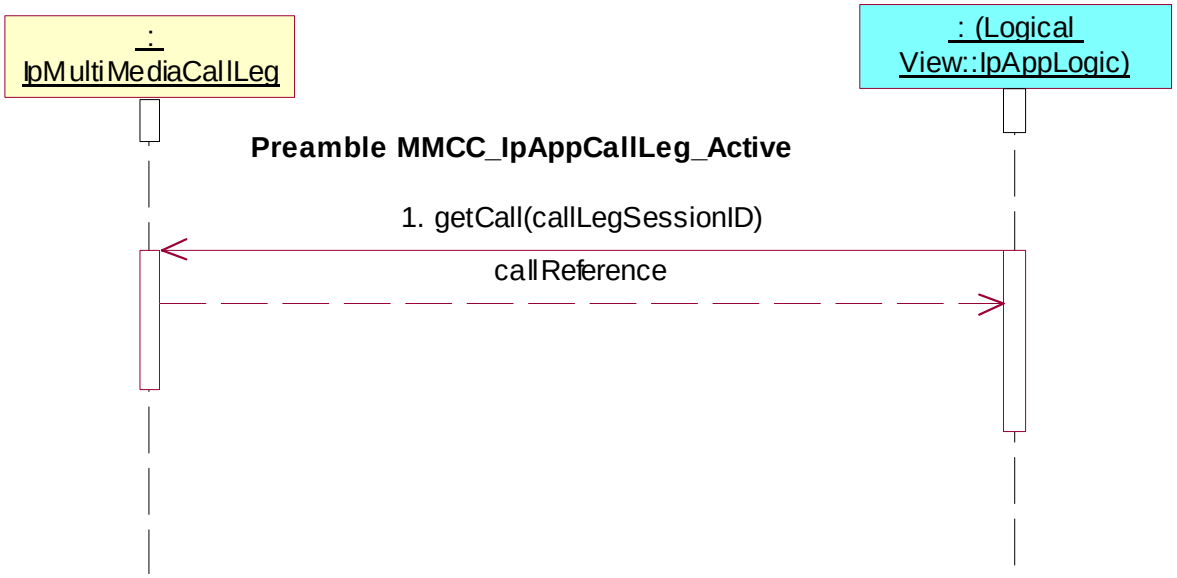
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **getCall()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **getCall()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID



Test MMCC_IpAppMultiMediaCallLeg_34

Summary: continue processing of call leg

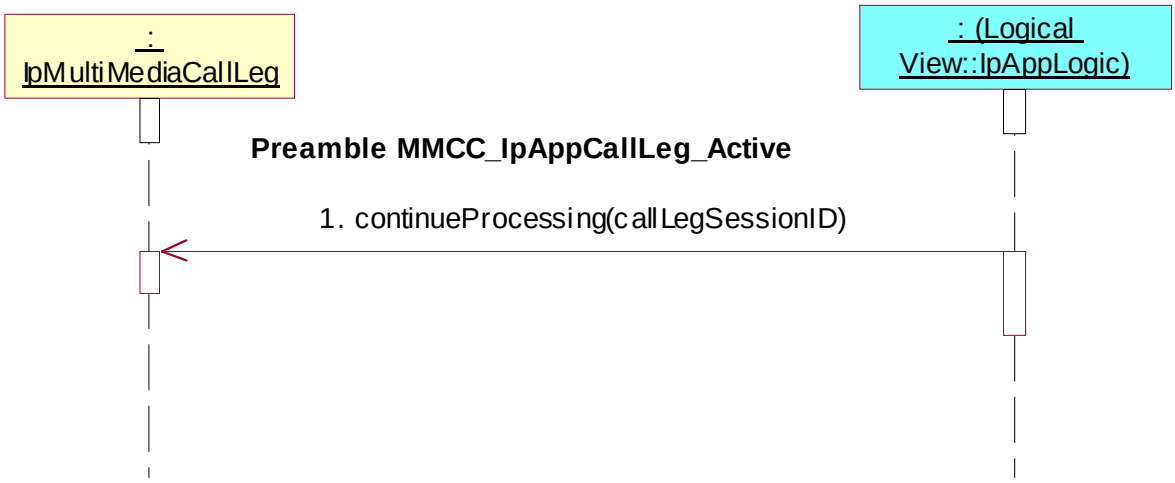
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **continueProcessing()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **continueProcessing()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID



Test MMCC_IpAppMultiMediaCallLeg_35

Summary: release call leg

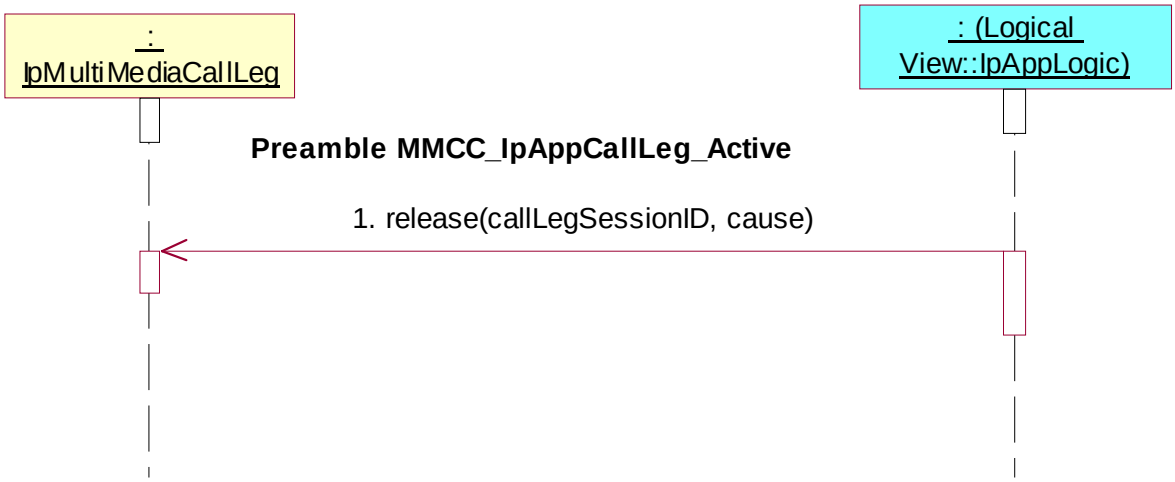
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **release()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, cause



Test MMCC_IpAppMultiMediaCallLeg_36

Summary: de-assign call leg

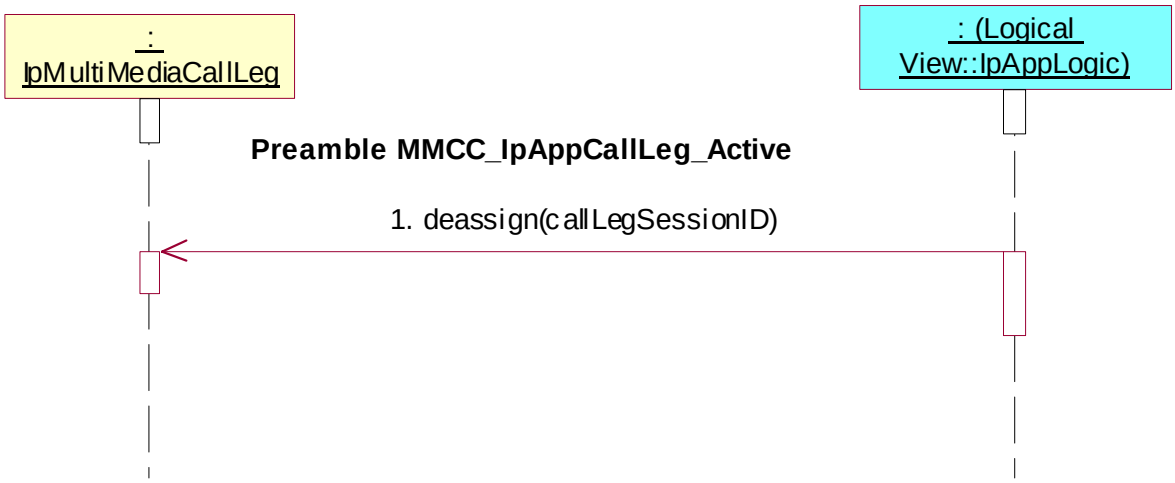
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **deassign()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **deassign()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID



Test MMCC_IpAppMultiMediaCallLeg_37

Summary: change or clear event criteria

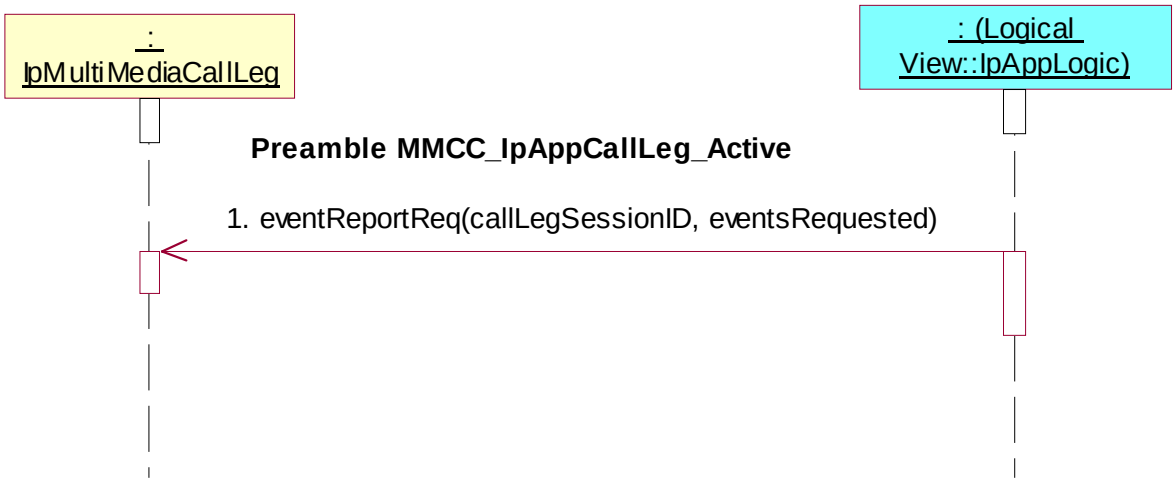
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **eventReportReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, eventsRequested



Test MMCC_IpAppMultiMediaCallLeg_38

Summary: change or clear event criteria, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.1

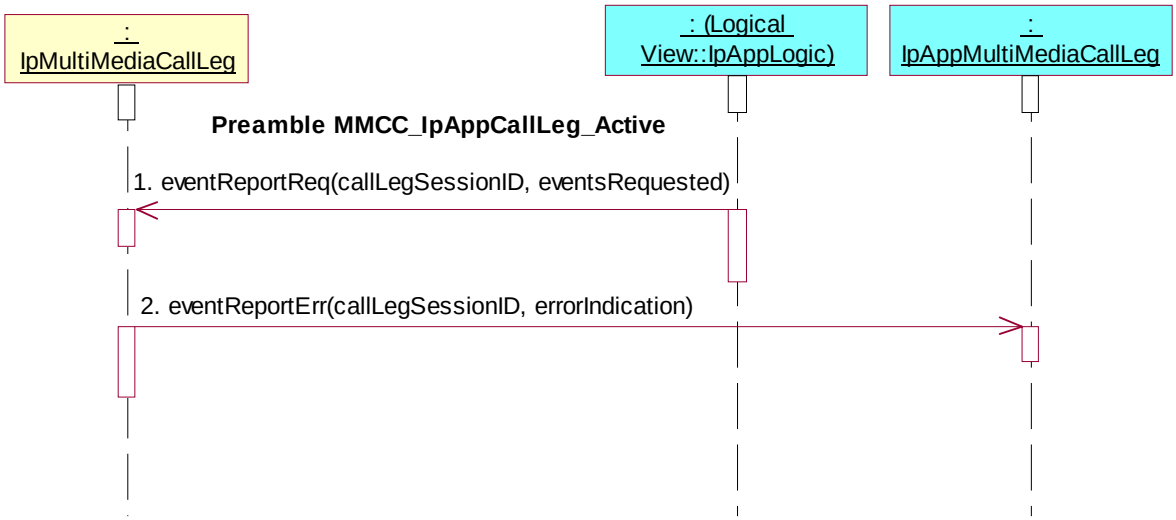
Precondition: IUT capable of invoking **eventReportReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, eventsRequested

2. Method call **eventReportErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallLeg_39

Summary: get information about call leg

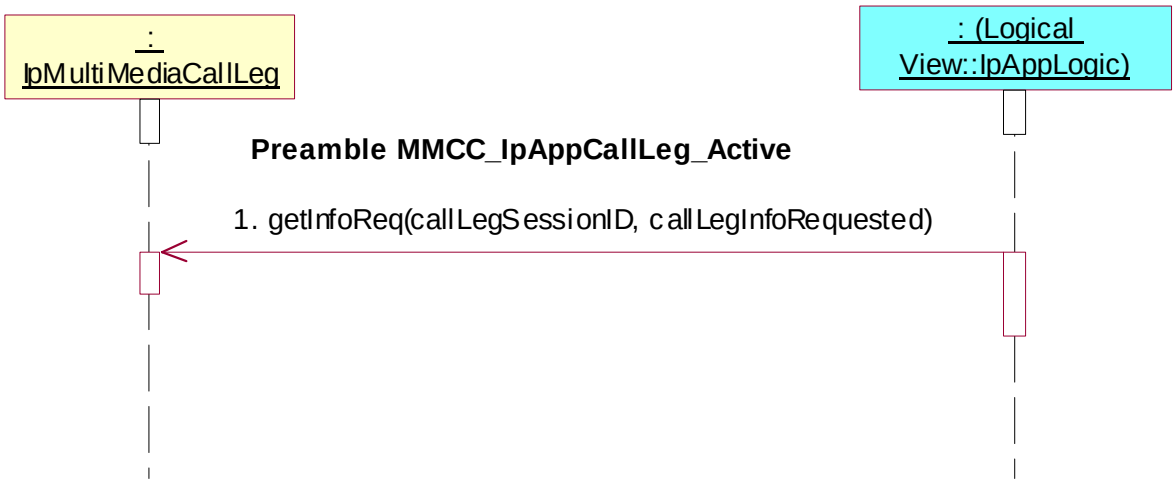
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, callLegInfoRequested



Test MMCC_IpAppMultiMediaCallLeg_40

Summary: get information about call leg, unsuccessful

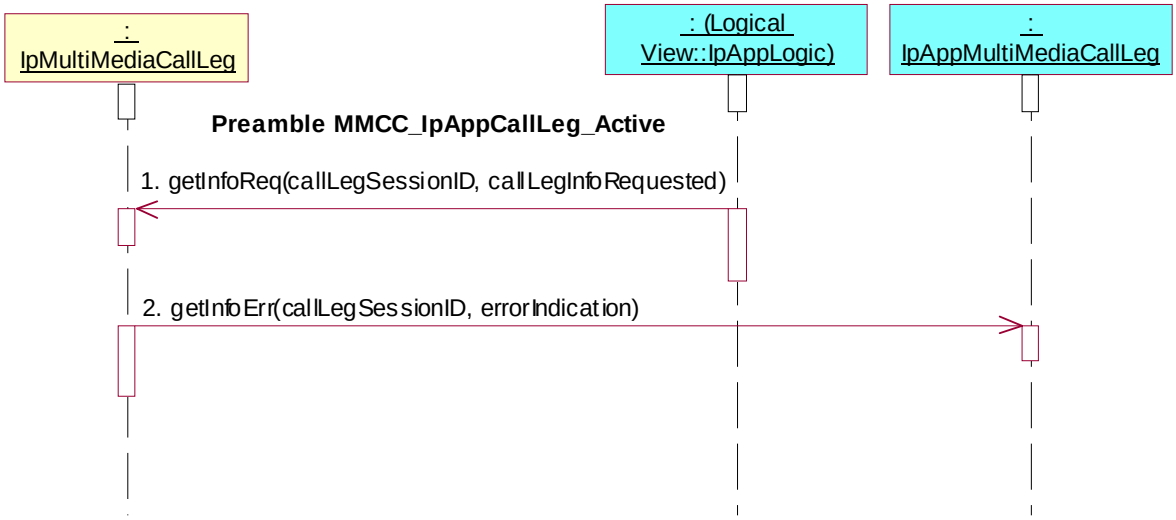
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, callLegInfoRequested
2. Method call **getInfoErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallLeg_41

Summary: set charge plan for call leg

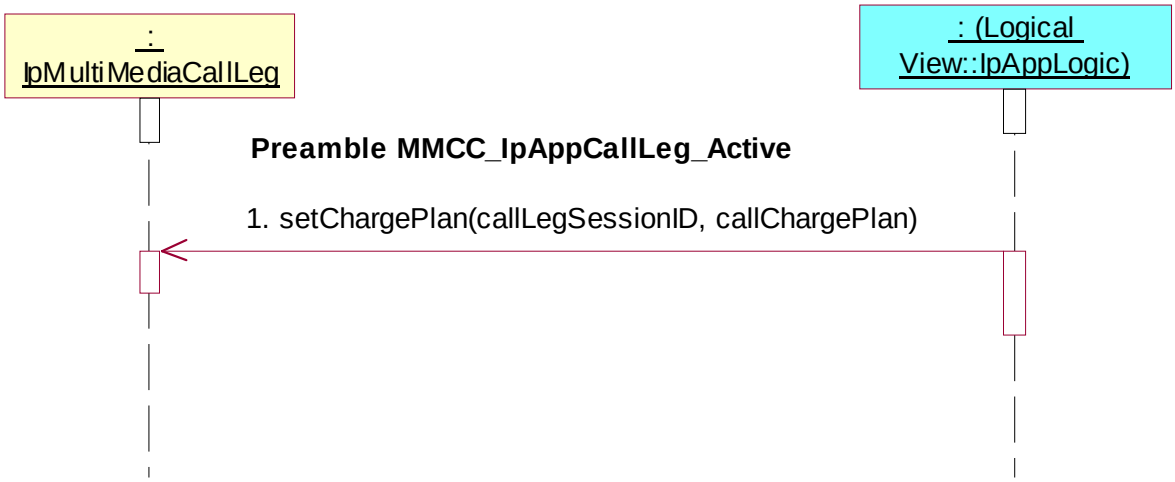
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **setChargePlan()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **setChargePlan()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, callChargePlan



Test MMCC_IpAppMultiMediaCallLeg_42

Summary: allow advice of charge information

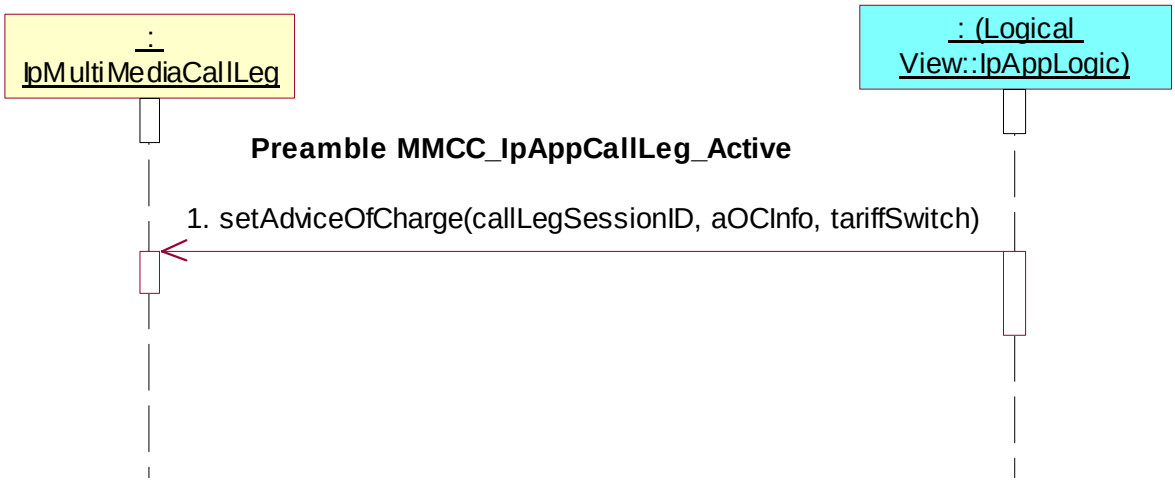
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **setAdviceOfCharge()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **setAdviceOfCharge()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, aOCInfo, tariffSwitch



Test MMCC_IpAppMultiMediaCallLeg_43

Summary: supervise call leg

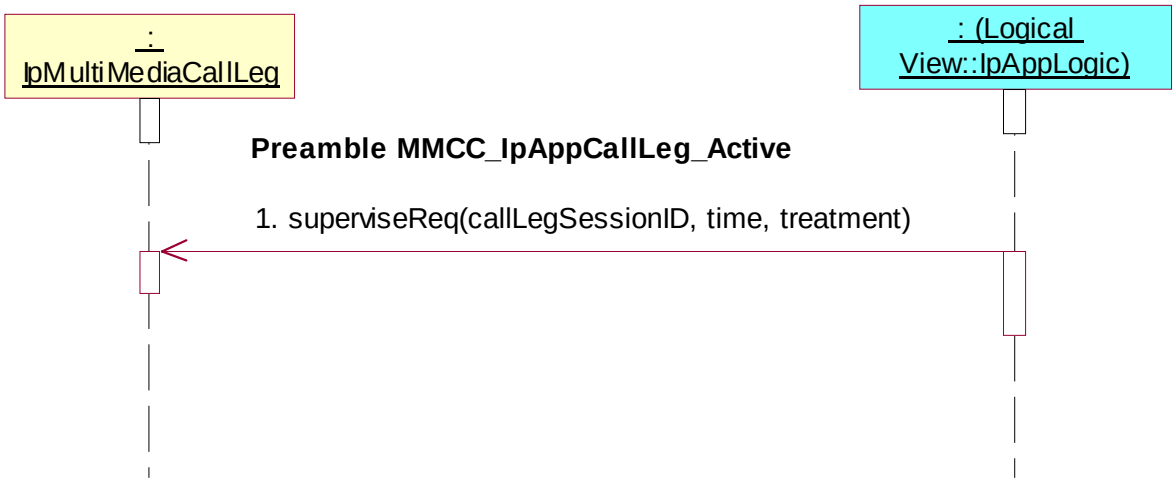
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **superviseReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, time, treatment



Test MMCC_IpAppMultiMediaCallLeg_44

Summary: supervise call leg, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.1

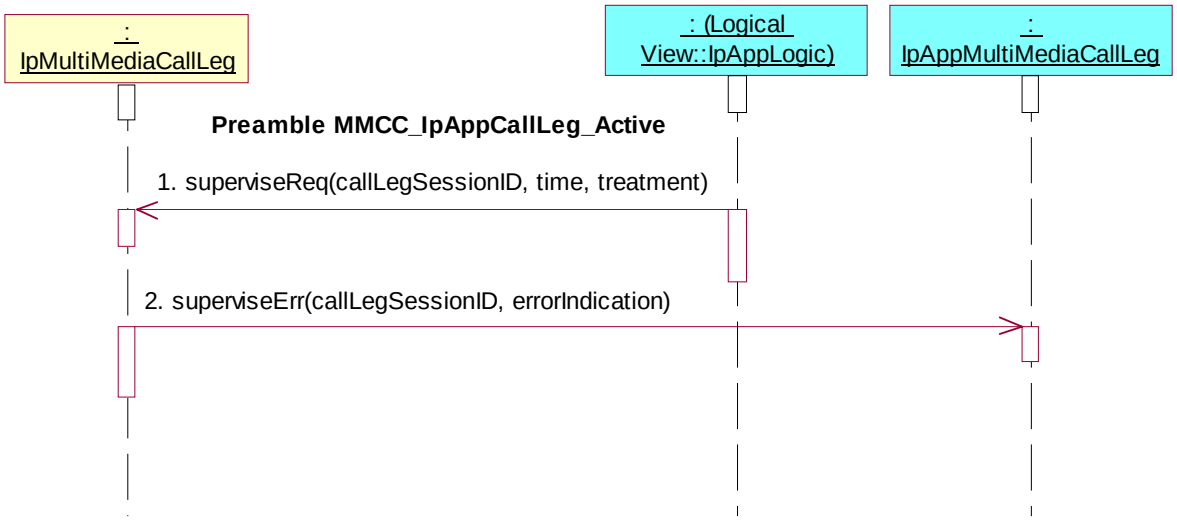
Precondition: IUT capable of invoking **superviseReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, time, treatment

2. Method call **superviseErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallLeg_45

Summary: set monitor on media streams

Reference: ES 202 915-4-3 [3], clause 7.3.1 and ES 202 915-4-4 [4], clauses 6.5 and 6.6

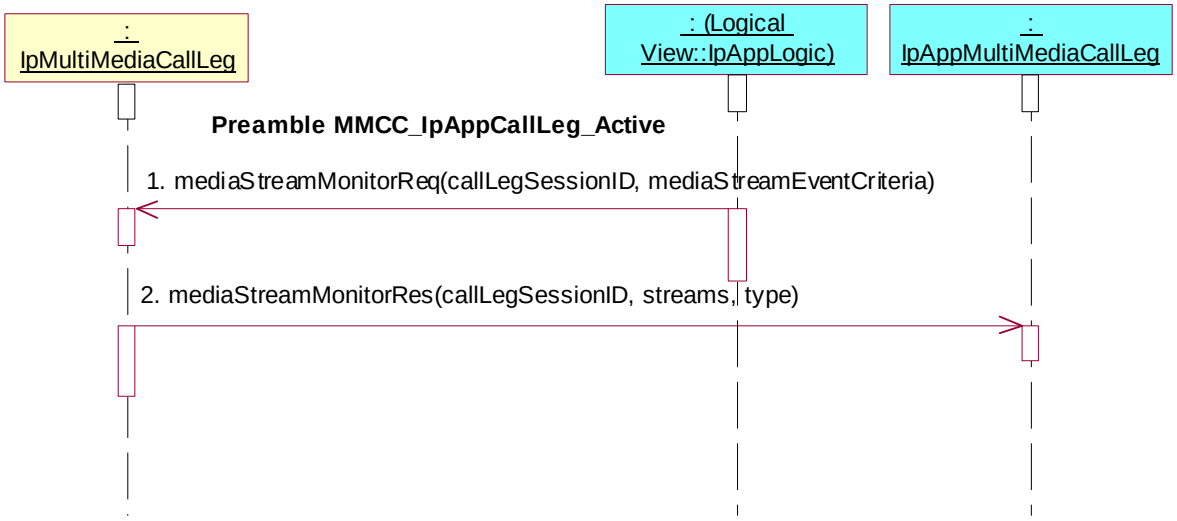
Precondition: IUT capable of invoking **mediaStreamMonitorReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **mediaStreamMonitorReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, mediaStreamEventCriteria

2. Method call **mediaStreamMonitorRes()**
   Parameters: callLegSessionID, streams, type
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallLeg_46

Summary: set monitor on media streams and allow setup

Reference: ES 202 915-4-3 [3], clause 7.3.1 and ES 202 915-4-4 [4], clauses 6.5 and 6.6

Precondition: IUT capable of invoking **mediaStreamMonitorReq()** and **mediaStreamAllow()**

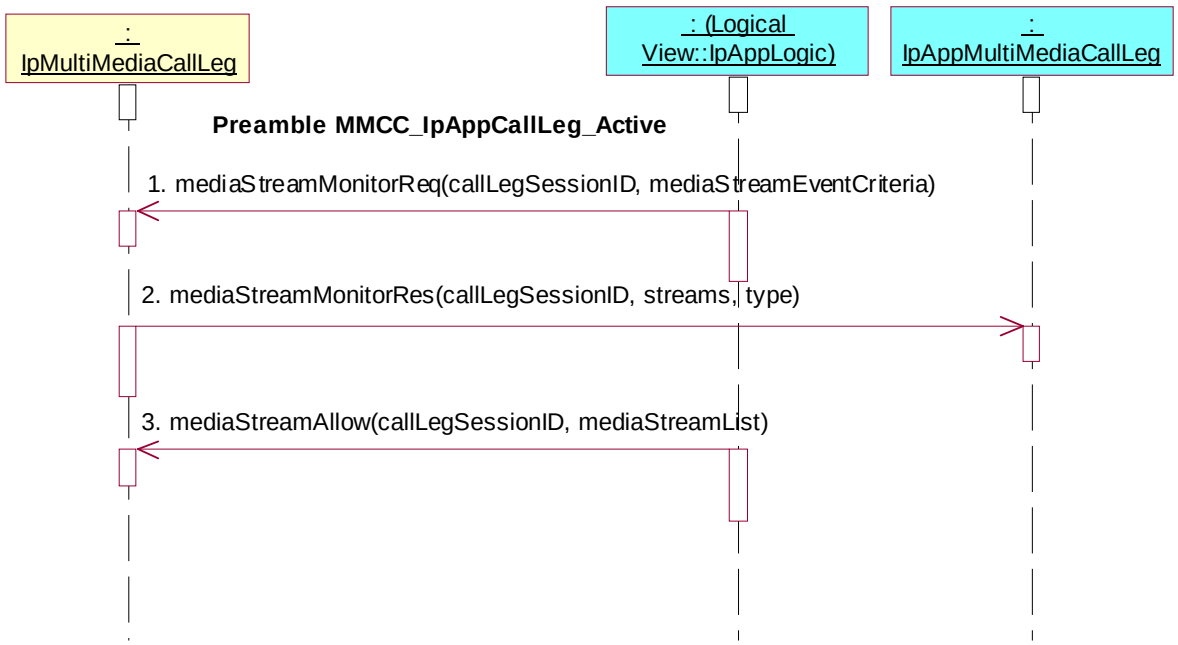
Preamble: **MMCC_IpAppMultiMediaCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **mediaStreamMonitorReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, mediaStreamEventCriteria
   mediaStreamEventCriteria.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT

2. Method call **mediaStreamMonitorRes()**
   Parameters: callLegSessionID, streams, type
   Check: no exception is returned

3. Triggered Action: cause IUT to call **mediaStreamAllow()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, mediaStreamList



Test MMCC_IpAppMultiMediaCallLeg_47

Summary: get media streams

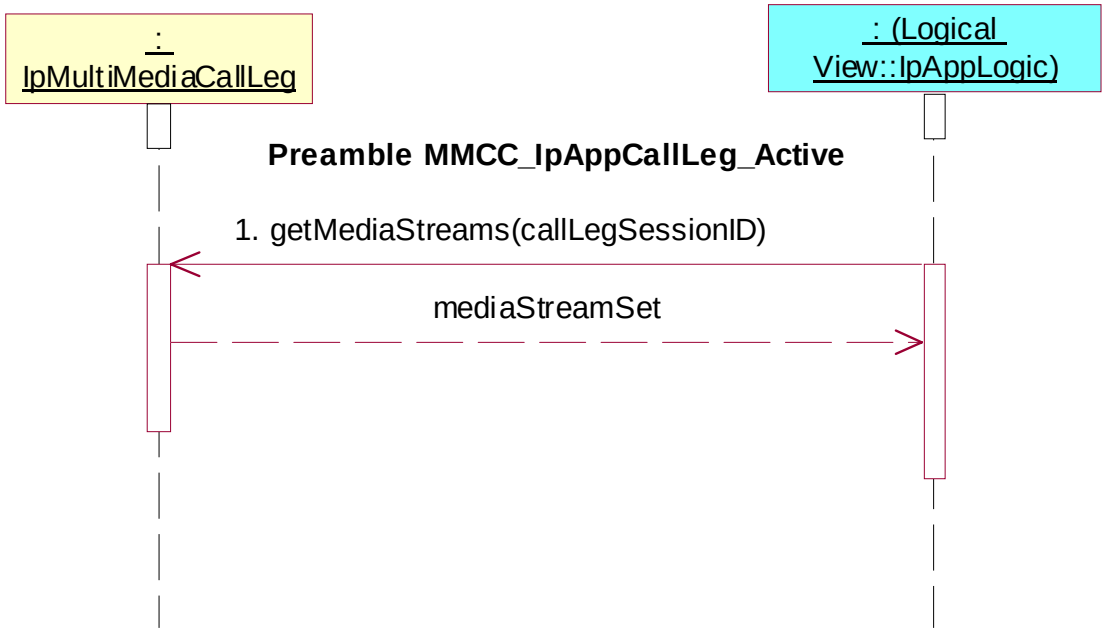
Reference: ES 202 915-4-3 [3], clause 7.3.1 and ES 202 915-4-4 [4], clauses 6.5 and 6.6

Precondition: IUT capable of invoking **getMediaStreams()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **getMediaStreams()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID



7.2.3.3.1.4 Releasing state

Preamble MMCC_IpAppMultiMediaCallLeg_Releasing

Reference: ES 202 915-4-3 [3], clause 7.3.1.4

Pre-preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiMediaCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiMediaCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Preamble Sequence:

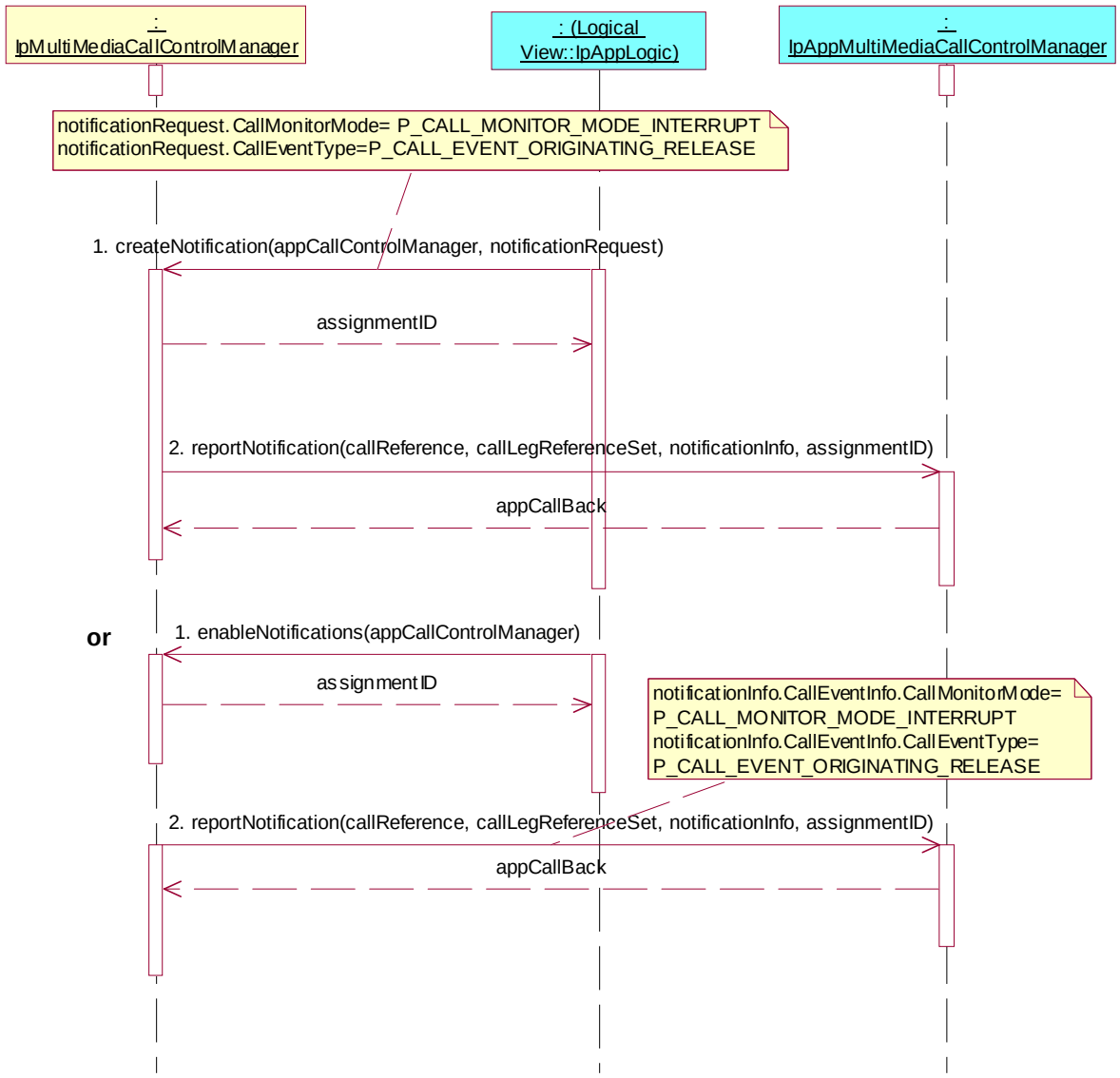
1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest
   notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   notificationRequest.CallEventType= P_CALL_EVENT_ORIGINATING_RELEASE

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   Check: valid value of TpAppMultiPartyCallBack is returned

or

1. Triggered Action: cause IUT to call **enableNotifications()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCallControlManager

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   notificationInfo.CallEventInfo.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   notificationInfo.CallEventInfo.CallEventType= P_CALL_EVENT_ORIGINATING_RELEASE
   Check: valid value of TpAppMultiMediaCallBack is returned



Test MMCC_IpAppMultiMediaCallLeg_48

Summary: request reference of call related to call leg

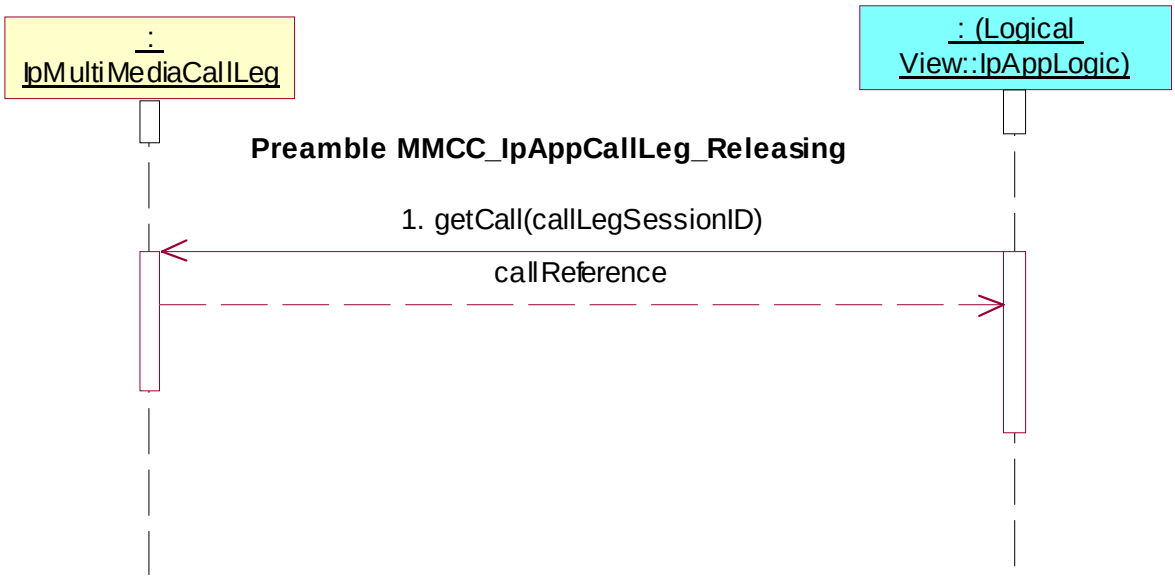
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **getCall()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Releasing**

Test Sequence:

1. Triggered Action: cause IUT to call **getCall()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID



Test MMCC_IpAppMultiMediaCallLeg_49

Summary: continue processing of call leg

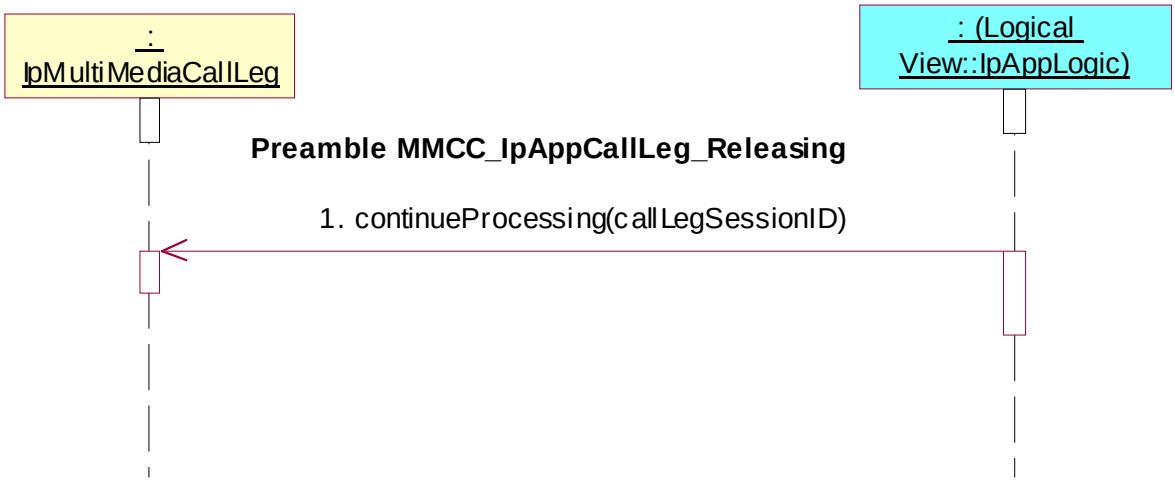
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **continueProcessing()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Releasing**

Test Sequence:

1. Triggered Action: cause IUT to call **continueProcessing()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID



Test MMCC_IpAppMultiMediaCallLeg_50

Summary: de-assign call leg

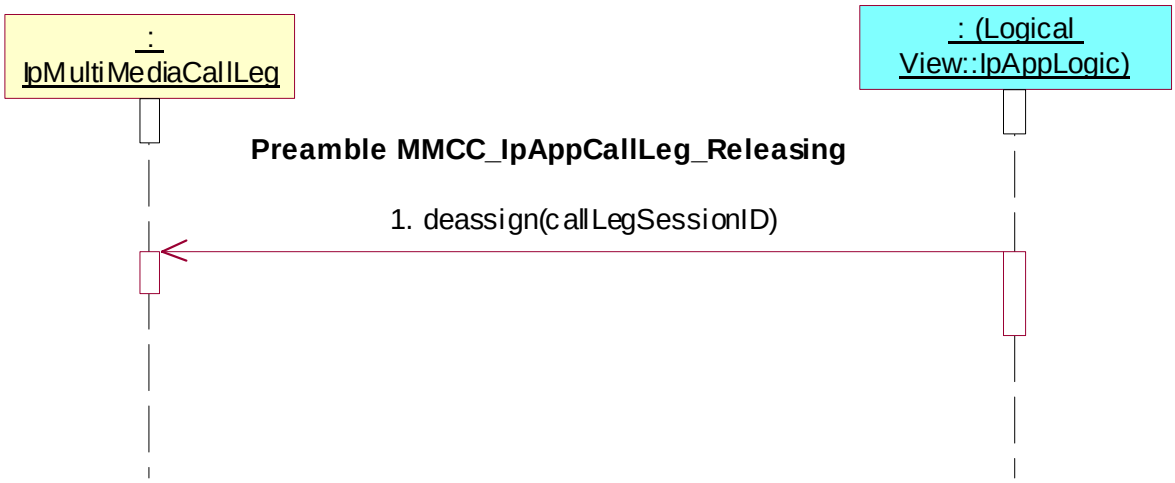
Reference: ES 202 915-4-3 [3], clause 7.3.1

Precondition: IUT capable of invoking **deassign()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Releasing**

Test Sequence:

1. Triggered Action: cause IUT to call **deassign()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID



7.2.3.3.2 Terminating Leg

Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **createNotification()** and **createCallLeg()**

7.2.3.3.2.1 Idle state

Preamble MMCC_IpAppMultiMediaCallLeg_Idle

Reference: ES 202 915-4-3 [3], clause 7.3.2.1

Pre-preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiMediaCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiMediaCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Preamble Sequence:

1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest
   notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   Check: valid value of TpAppMultiPartyCallBack is returned

3. Triggered Action: cause IUT to call **createCallLeg()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID, AppMultiMediaCallLeg

or

1. Triggered Action: cause IUT to call **createCall()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCall

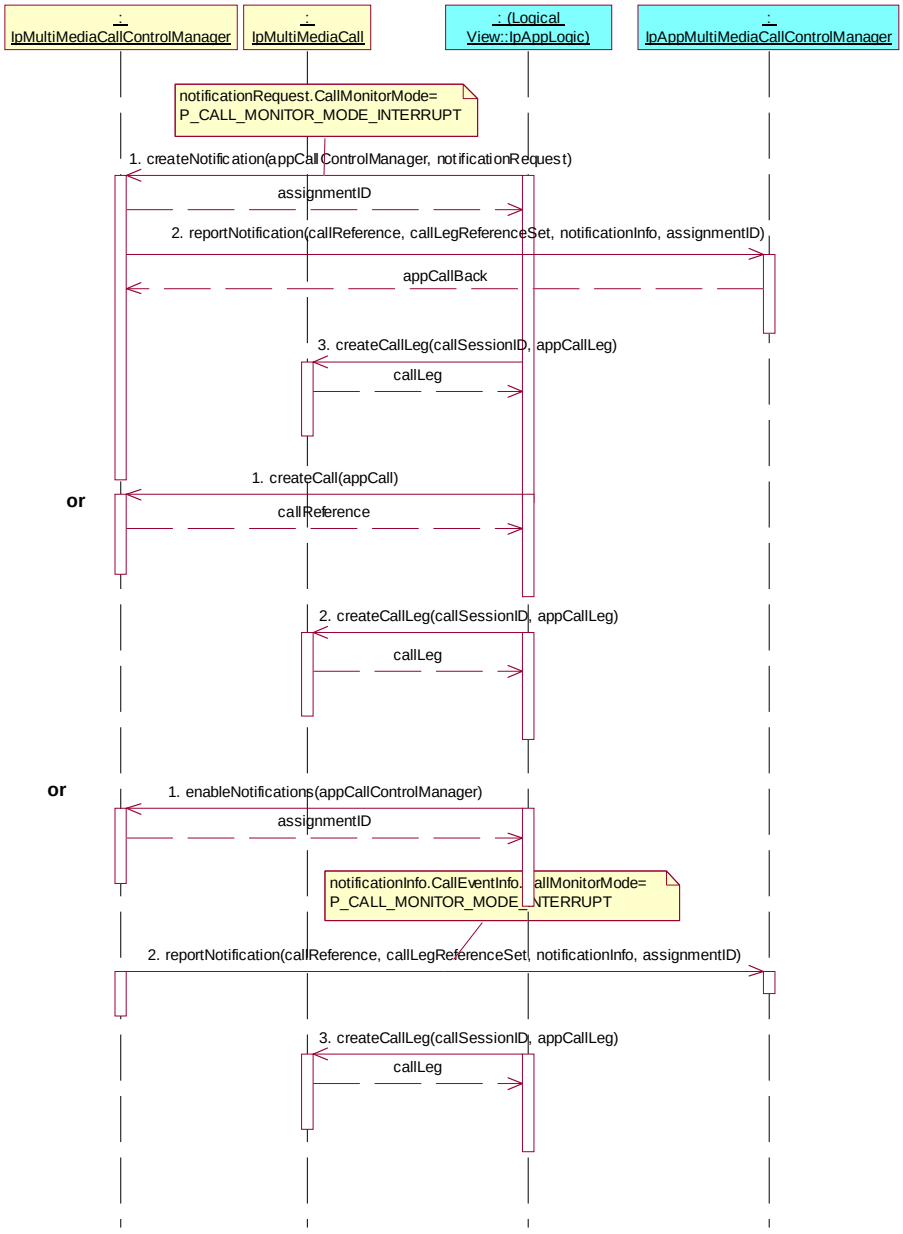
2. Triggered Action: cause IUT to call **createCallLeg()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID, appCallLeg

or

1. Triggered Action: cause IUT to call **enableNotifications()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCallControlManager

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   notificationInfo.CallEventInfo.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   Check: valid value of TpAppMultiMediaCallBack is returned

3. Triggered Action: cause IUT to call **createCallLeg()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID, AppMultiMediaCallLeg



Test MMCC_IpAppMultiMediaCallLeg_51

Summary: route call leg, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.2

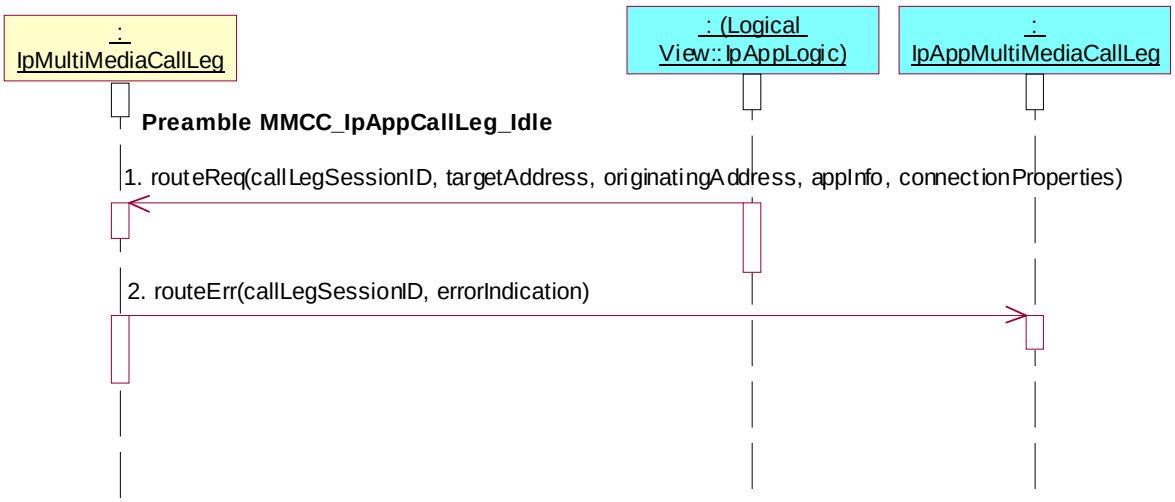
Precondition: IUT capable of invoking **routeReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **routeReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties

2. Method call **routeErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallLeg_52

Summary: request reference of call related to call leg

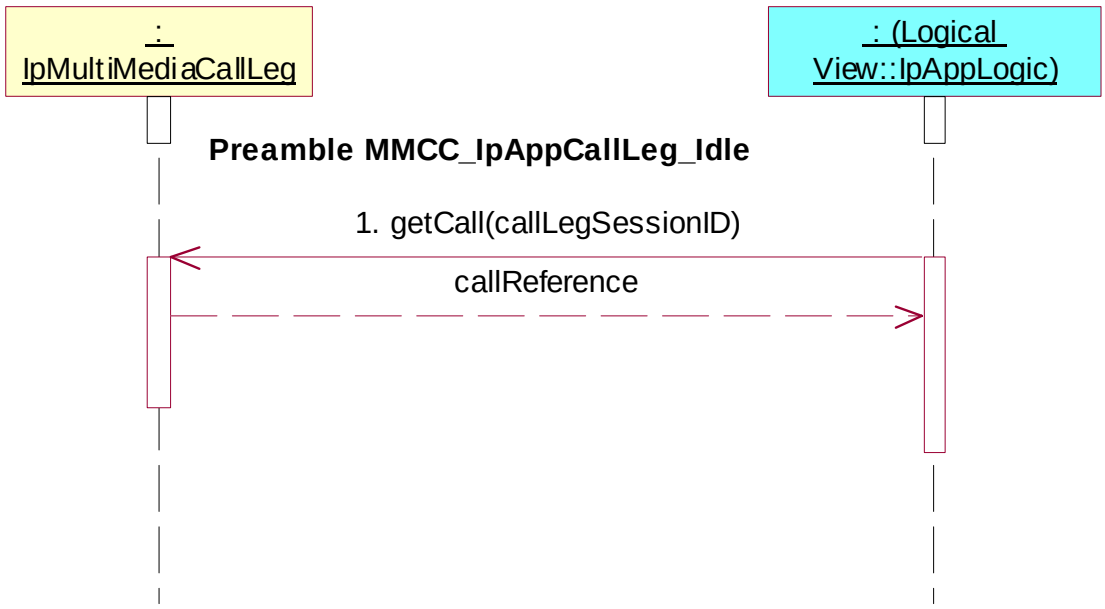
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **getCall()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **getCall()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID



Test MMCC_IpAppMultiMediaCallLeg_53

Summary: release call leg

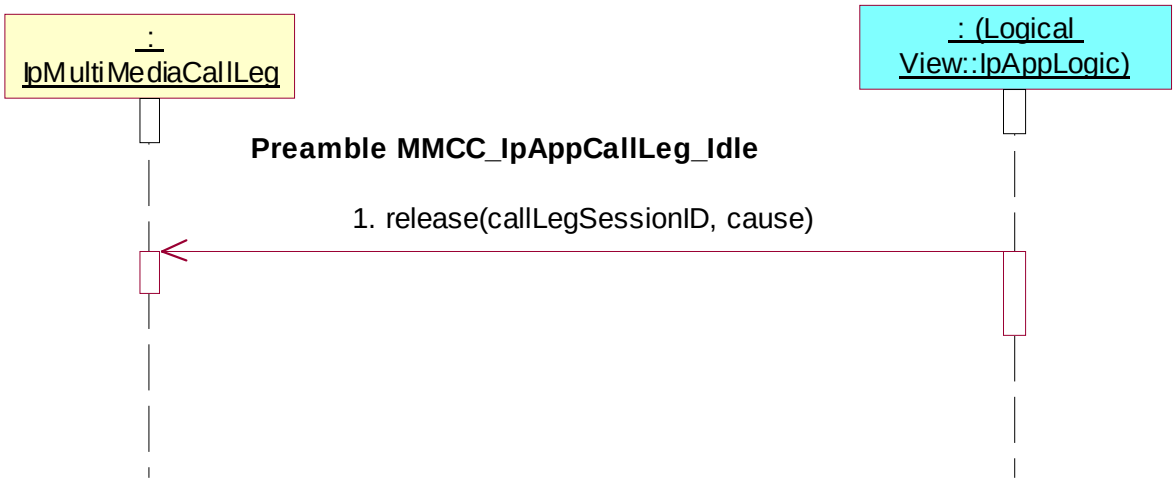
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **release()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, cause



Test MMCC_IpAppMultiMediaCallLeg_54

Summary: change or clear event criteria

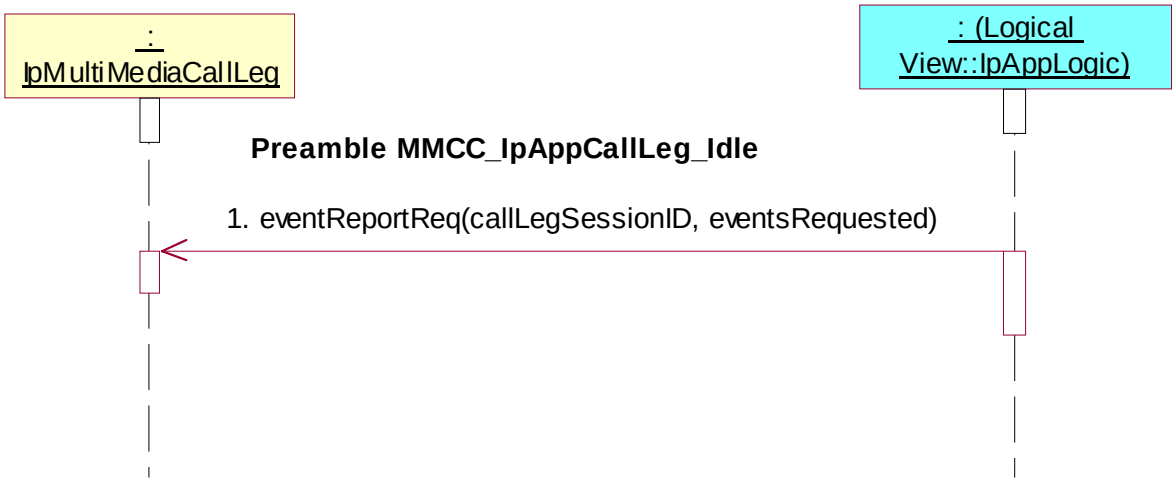
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **eventReportReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, eventsRequested



Test MMCC_IpAppMultiMediaCallLeg_55

Summary: change or clear event criteria, successful

Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **eventReportReq(), routeReq()**

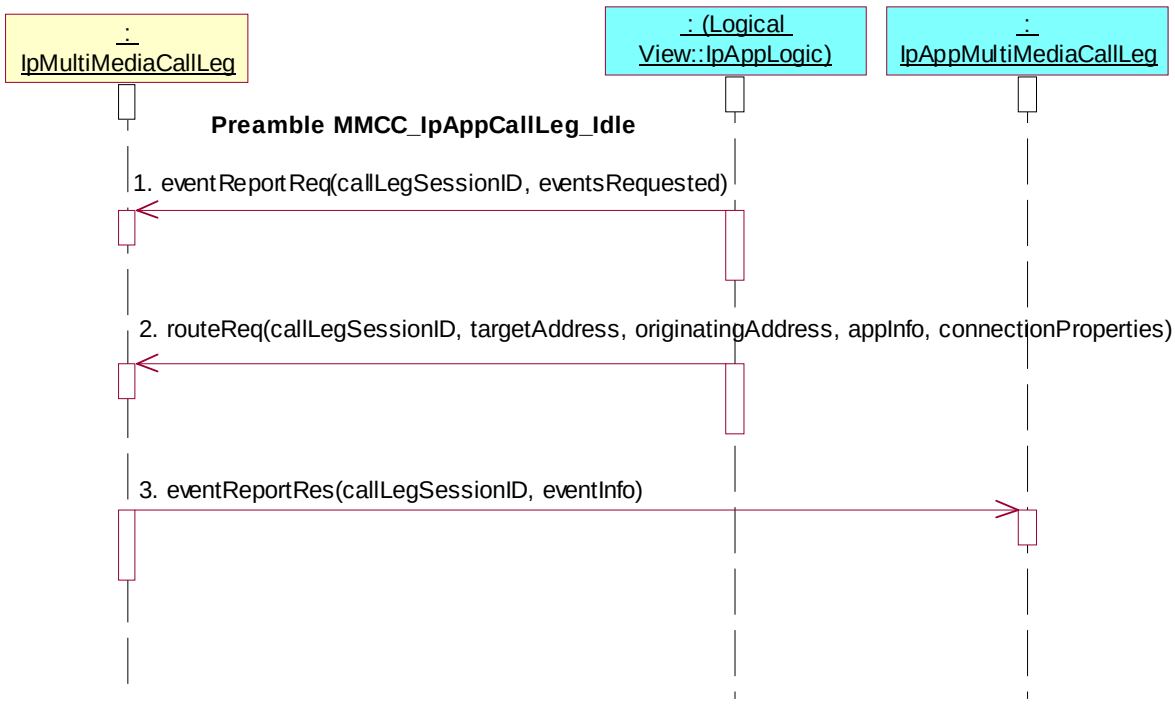
Preamble: **MMCC_IpAppCallLeg_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, eventsRequested

2. Triggered Action: cause IUT to call **routeReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties

3. Method call **eventReportRes()**
   Parameters: callLegSessionID, eventInfo
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallLeg_56

Summary: change or clear event criteria, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.2

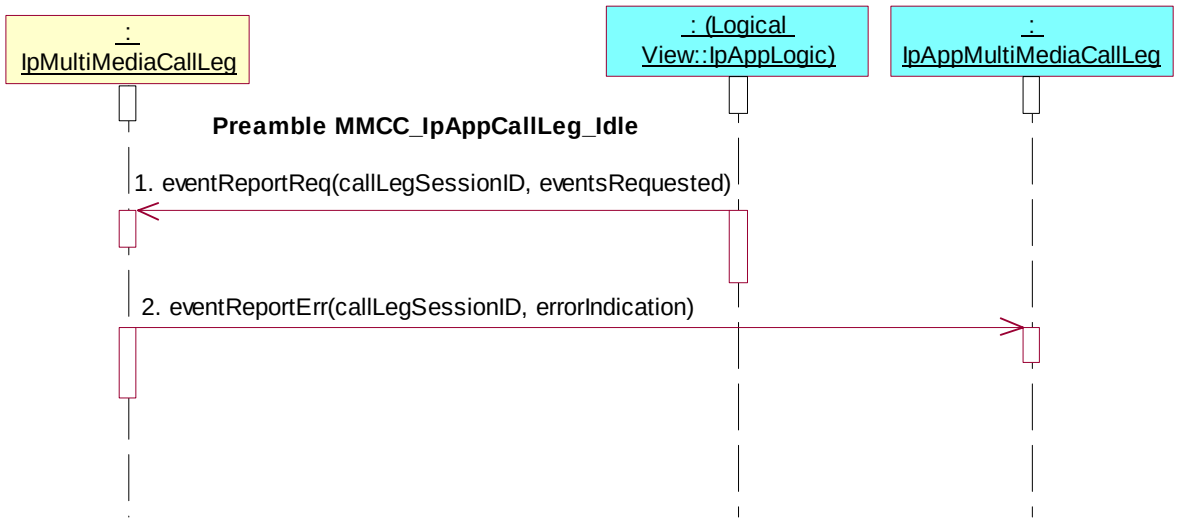
Precondition: IUT capable of invoking **eventReportReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, eventsRequested

2. Method call **eventReportErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallLeg_57

Summary: get information about call leg

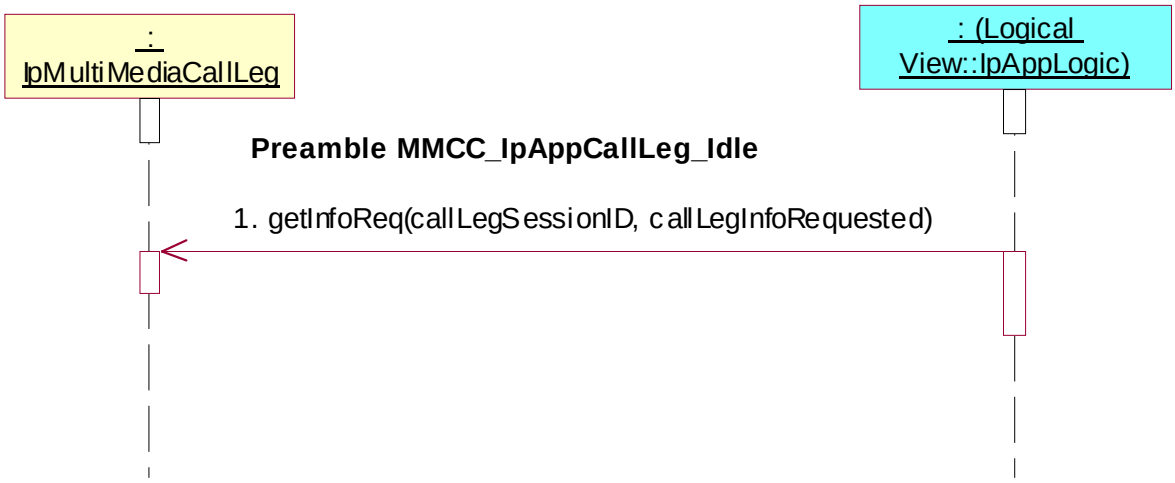
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, callLegInfoRequested



Test MMCC_IpAppMultiMediaCallLeg_58

Summary: get information about call leg, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.2

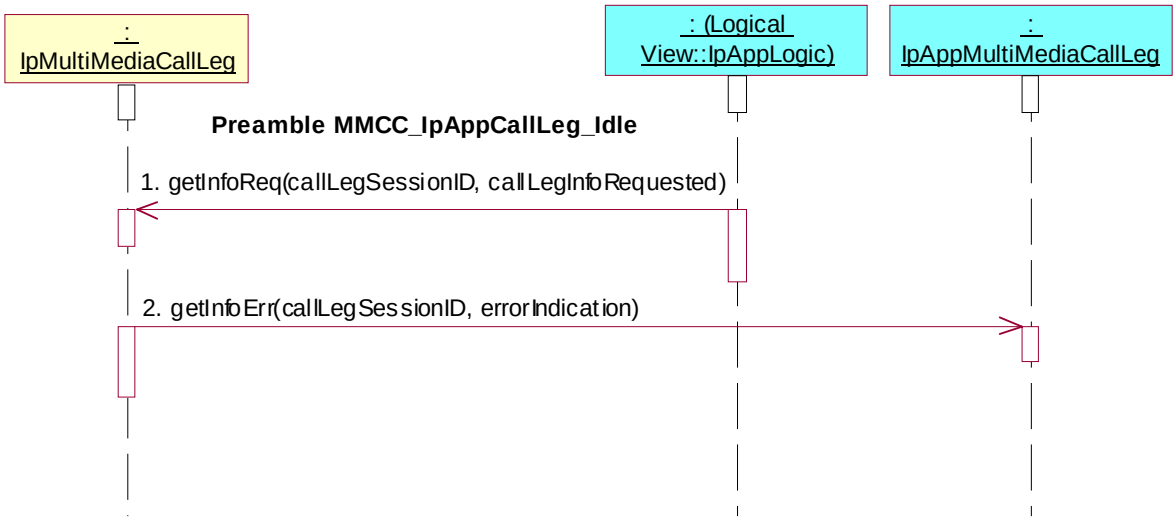
Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, callLegInfoRequested

2. Method call **getInfoErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallLeg_59

Summary: set charge plan for call leg

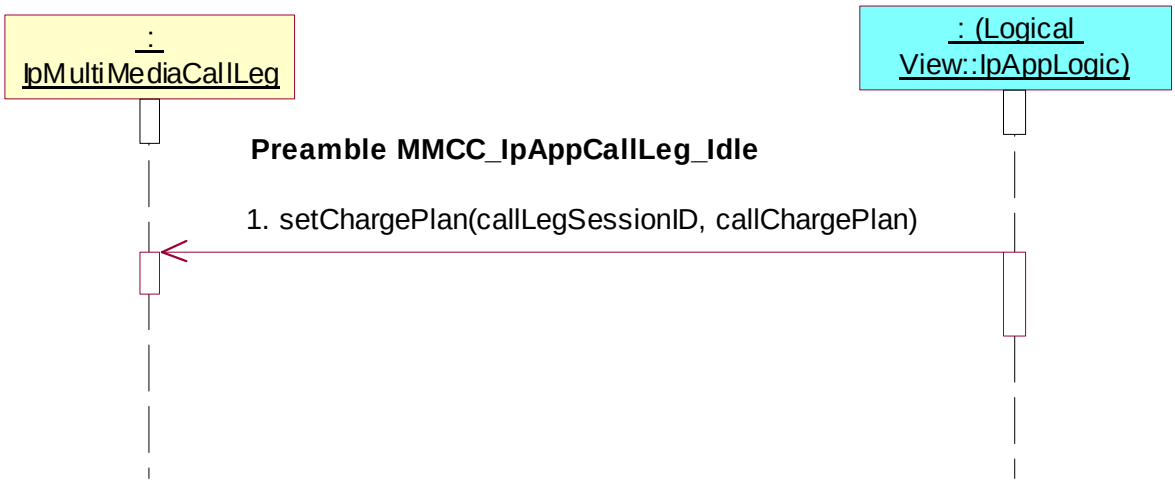
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **setChargePlan()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **setChargePlan()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, callChargePlan



Test MMCC_IpAppMultiMediaCallLeg_60

Summary: allow advice of charge information

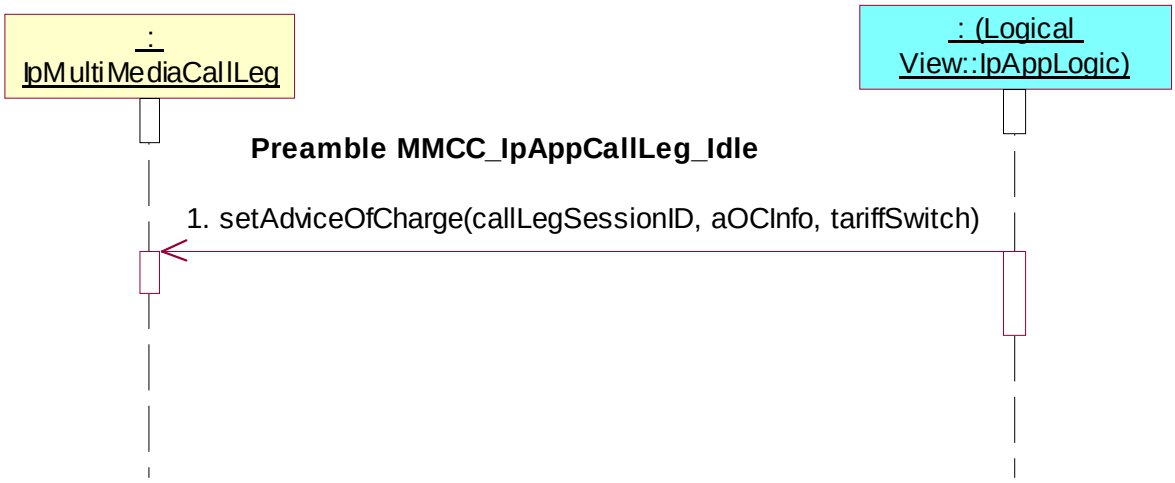
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **setAdviceOfCharge()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **setAdviceOfCharge()** method on the tester's (SCF's terminating) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, aOCInfo, tariffSwitch



Test MMCC_IpAppMultiMediaCallLeg_61

Summary: supervise call leg

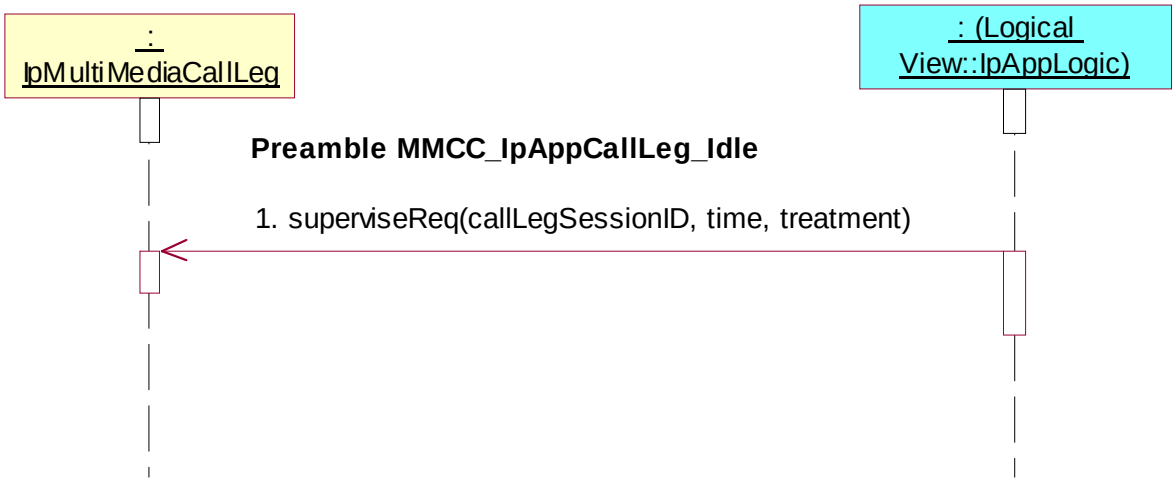
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **superviseReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, time, treatment



Test MMCC_IpAppMultiMediaCallLeg_62

Summary: supervise call leg, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.2

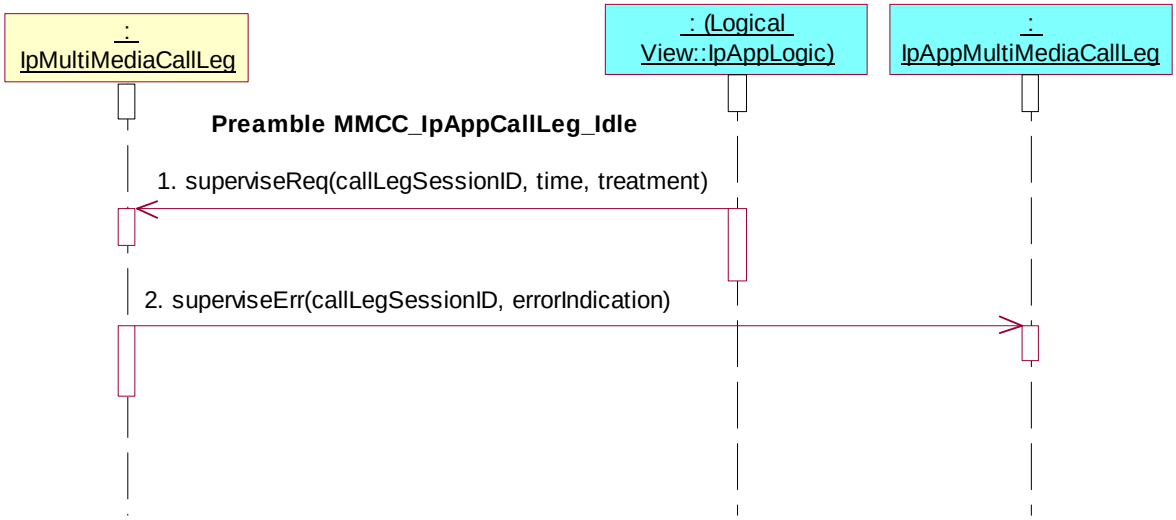
Precondition: IUT capable of invoking **superviseReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Idle**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, time, treatment

2. Method call **superviseErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



7.2.3.3.2.2 Active (Terminating) state

Precondition: IUT capable of invoking **eventReportReq()** and **routeReq()**

Preamble MMCC_IpAppMultiMediaCallLeg_Active_Terminating

Reference: ES 202 915-4-3 [3], clause 7.3.2.2

Pre-preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiMediaCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiMediaCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Preamble Sequence:

1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest
   notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   Check: valid value of TpAppMultiPartyCallBack is returned

3. Triggered Action: cause IUT to call **createCallLeg()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID, AppMultiMediaCallLeg

4. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, eventsRequested

5. Triggered Action: cause IUT to call **routeReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties

6. Method call **eventReportRes()**
   Parameters: callLegSessionID, eventInfo
   Check: no exception is returned

or

1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest
   notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   notificationRequest.CallEventType= P_CALL_EVENT_TERMINATING_CALL_ATTEMPT or P_CALL_EVENT_TERMINATING_CALL_ATTEMPT_AUTHORISED or P_CALL_EVENT_ALERTING or P_CALL_EVENT_ANSWER or P_CALL_EVENT_REDIRECTED or P_CALL_EVENT_QUEUED or P_CALL_EVENT_TERMINATING_SERVICE_CODE

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   Check: valid value of TpAppMultiPartyCallBack is returned

or

1. Triggered Action: cause IUT to call **createCall()** method on the tester's (SCF's) IpMultiPartyCallControlManager interface.
   Parameters: appCall

2. Triggered Action: cause IUT to call **createCallLeg()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID, AppMultiMediaCallLeg

3. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, eventsRequested

4. Triggered Action: cause IUT to call **routeReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties

5. Method call **eventReportRes()**
   Parameters: callLegSessionID, eventInfo
   Check: no exception is returned

or

1. Triggered Action: cause IUT to call **enableNotifications()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCallControlManager

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   notificationInfo.CallEventInfo.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   Check: valid value of TpAppMultiMediaCallBack is returned

3. Triggered Action: cause IUT to call **createCallLeg()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID, AppMultiMediaCallLeg

4. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, eventsRequested

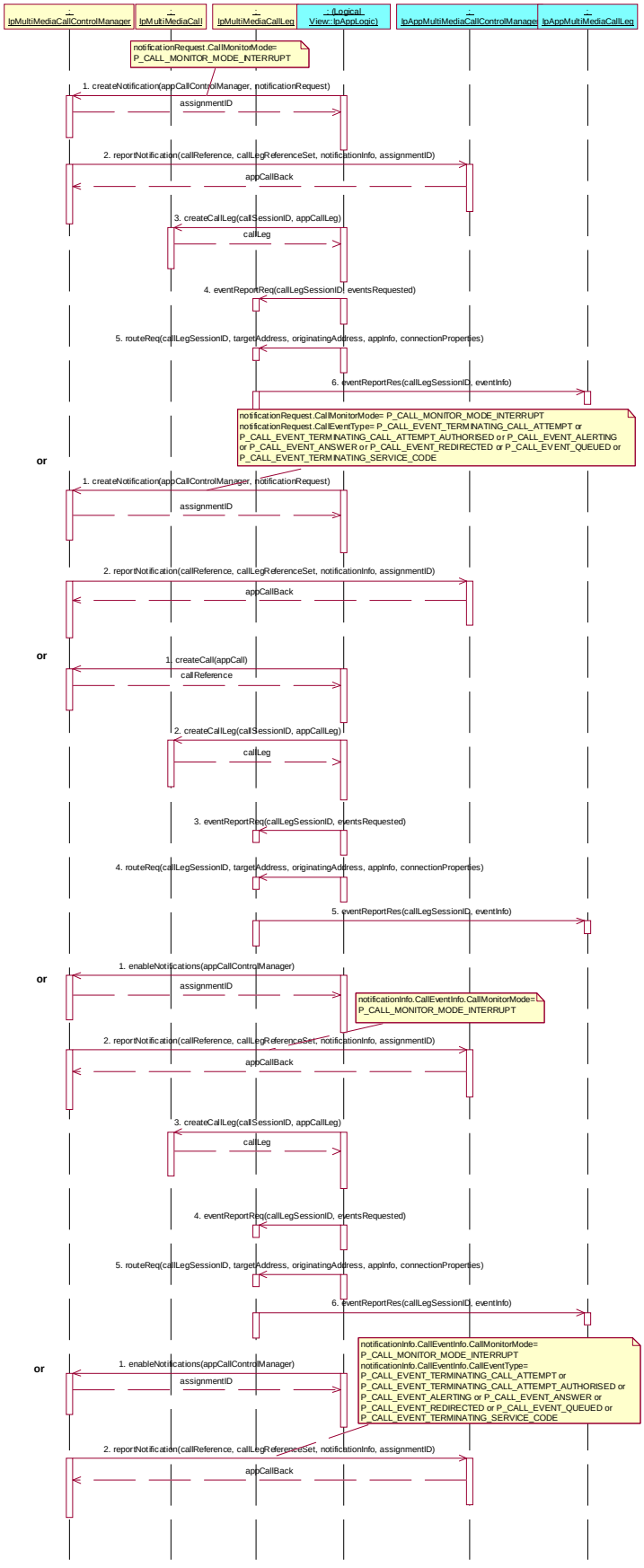
5. Triggered Action: cause IUT to call **routeReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties

6. Method call **eventReportRes()**
   Parameters: callLegSessionID, eventInfo
   Check: no exception is returned

or

1. Triggered Action: cause IUT to call **enableNotifications()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCallControlManager

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   notificationInfo.CallEventInfo.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   notificationInfo.CallEventInfo.CallEventType=
   P_CALL_EVENT_TERMINATING_CALL_ATTEMPT or
   P_CALL_EVENT_TERMINATING_CALL_ATTEMPT_AUTHORISED or
   P_CALL_EVENT_ALERTING or P_CALL_EVENT_ANSWER or
   P_CALL_EVENT_REDIRECTED or P_CALL_EVENT_QUEUED or
   P_CALL_EVENT_TERMINATING_SERVICE_CODE
   Check: valid value of TpAppMultiMediaCallBack is returned


IpMultiMediaCallControlManager
IpMultiMediaCall
IpMultiMediaCallLeg
(Logical View::IpAppLogic)
IpAppMultiMediaCallControlManager
IpAppMultiMediaCallLeg
notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
1. createNotification(appCallControlManager, notificationRequest)
assignmentID
2. reportNotification(callReference, callLegReferenceSet, notificationInfo, assignmentID)
appCallBack
3. createCallLeg(callSessionID, appCallLeg)
callLeg
4. eventReportReq(callLegSessionID, eventsRequested)
5. routeReq(callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties)
6. eventReportRes(callLegSessionID, eventInfo)
notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
notificationRequest.CallEventType= P_CALL_EVENT_TERMINATING_CALL_ATTEMPT or P_CALL_EVENT_TERMINATING_CALL_ATTEMPT_AUTHORISED or P_CALL_EVENT_ALERTING or P_CALL_EVENT_ANSWER or P_CALL_EVENT_REDIRECTED or P_CALL_EVENT_QUEUED or P_CALL_EVENT_TERMINATING_SERVICE_CODE
or
1. createNotification(appCallControlManager, notificationRequest)
assignmentID
2. reportNotification(callReference, callLegReferenceSet, notificationInfo, assignmentID)
appCallBack
or
1. createCall(appCall)
callReference
2. createCallLeg(callSessionID, appCallLeg)
callLeg
3. eventReportReq(callLegSessionID, eventsRequested)
4. routeReq(callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties)
5. eventReportRes(callLegSessionID, eventInfo)
or
1. enableNotifications(appCallControlManager)
assignmentID
notificationInfo.CallEventInfo.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
2. reportNotification(callReference, callLegReferenceSet, notificationInfo, assignmentID)
appCallBack
3. createCallLeg(callSessionID, appCallLeg)
callLeg
4. eventReportReq(callLegSessionID, eventsRequested)
5. routeReq(callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties)
6. eventReportRes(callLegSessionID, eventInfo)
or
1. enableNotifications(appCallControlManager)
assignmentID
notificationInfo.CallEventInfo.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
notificationInfo.CallEventInfo.CallEventType= P_CALL_EVENT_TERMINATING_CALL_ATTEMPT or P_CALL_EVENT_TERMINATING_CALL_ATTEMPT_AUTHORISED or P_CALL_EVENT_ALERTING or P_CALL_EVENT_ANSWER or P_CALL_EVENT_REDIRECTED or P_CALL_EVENT_QUEUED or P_CALL_EVENT_TERMINATING_SERVICE_CODE
2. reportNotification(callReference, callLegReferenceSet, notificationInfo, assignmentID)
appCallBack

Test MMCC_IpAppMultiMediaCallLeg_63

Summary: attach media, successful

Reference: ES 202 915-4-3 [3], clause 7.3.2

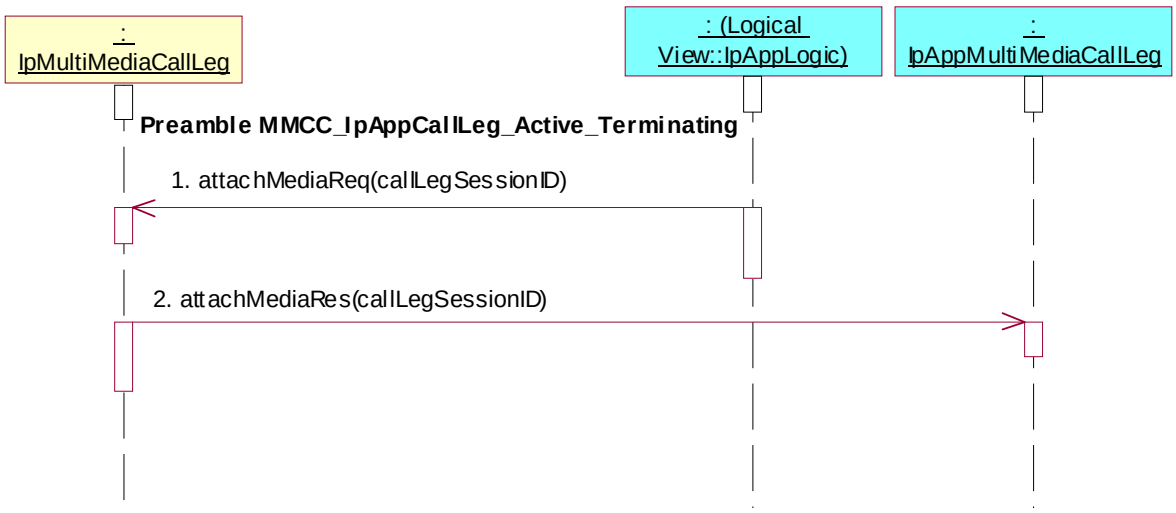
Precondition: IUT capable of invoking **attachMediaReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **attachMediaReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID

2. Method call **attachMediaRes()**
   Parameters: callLegSessionID
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallLeg_64

Summary: attach media, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.2

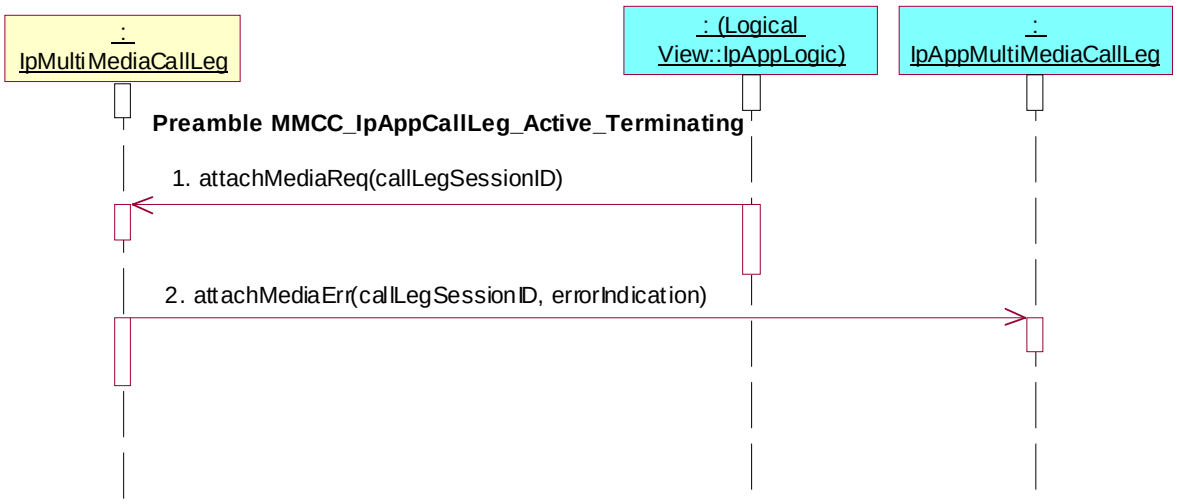
Precondition: IUT capable of invoking **attachMediaReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **attachMediaReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID

2. Method call **attachMediaErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallLeg_65

Summary: detach media, successful

Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **detachMediaReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active_Terminating**

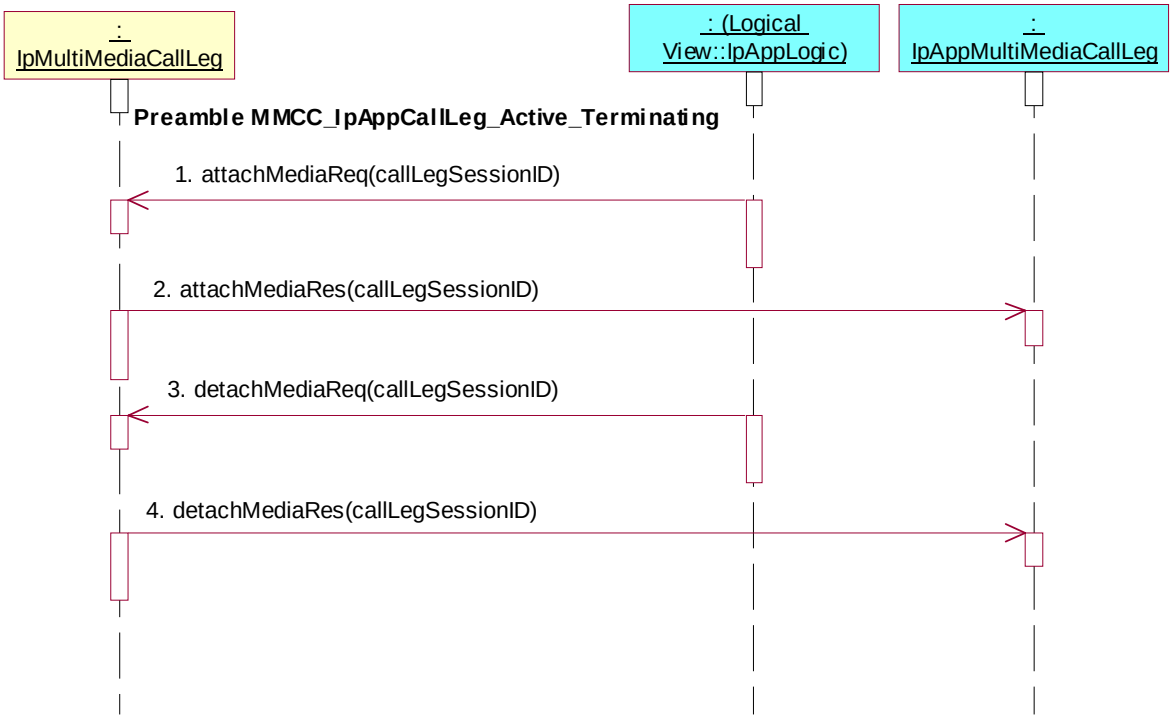
Test Sequence:

1. Triggered Action: cause IUT to call **attachMediaReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID

2. Method call **attachMediaRes()**
   Parameters: callLegSessionID
   Check: no exception is returned

3. Triggered Action: cause IUT to call **detachMediaReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID

4. Method call **detachMediaRes()**
   Parameters: callLegSessionID
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallLeg_66

Summary: detach media, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **detachMediaReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active_Terminating**

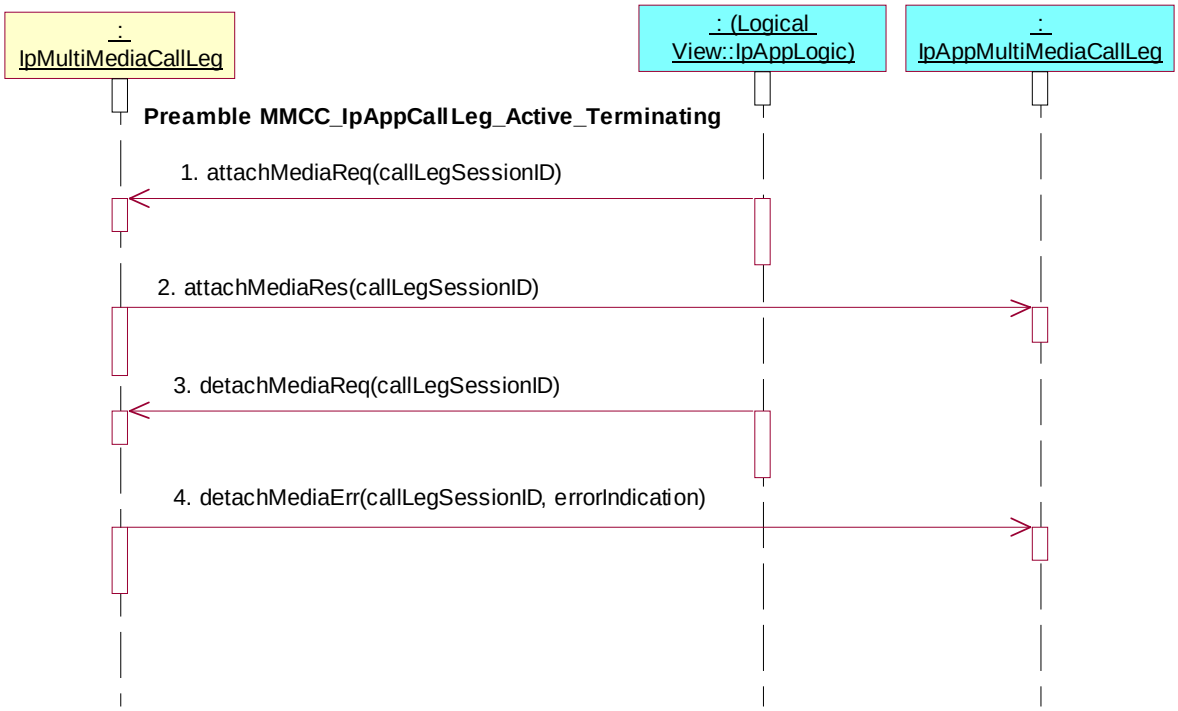
Test Sequence:

1. Triggered Action: cause IUT to call **attachMediaReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID

2. Method call **attachMediaRes()**
   Parameters: callLegSessionID
   Check: no exception is returned

3. Triggered Action: cause IUT to call **detachMediaReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID

4. Method call **detachMediaErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallLeg_67

Summary: request reference of call related to call leg

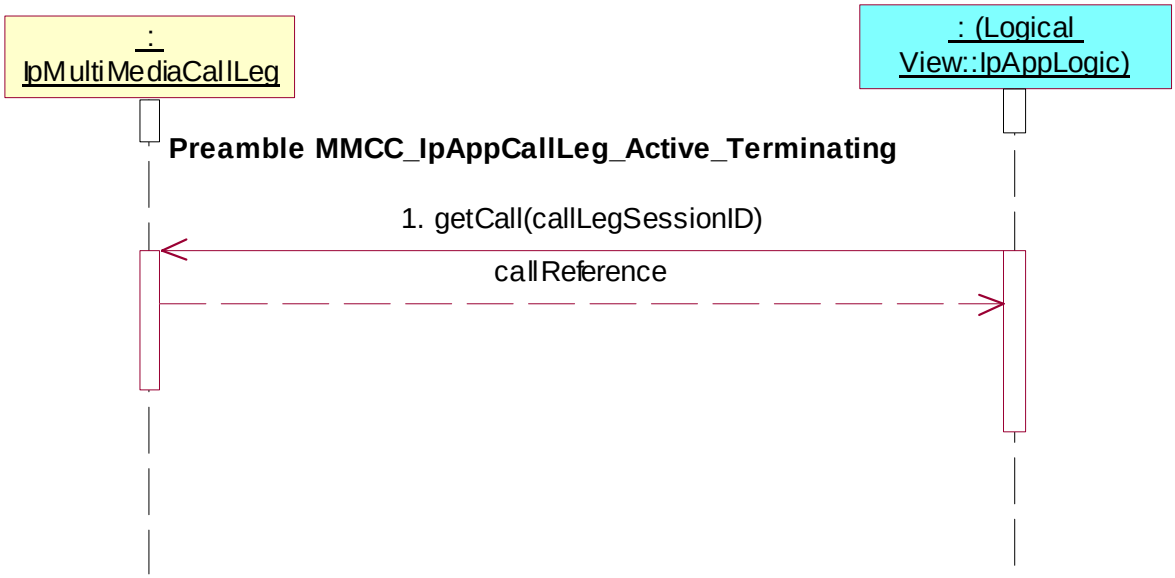
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **getCall()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **getCall()** method on the tester's (SCF's) IpMultiMediaCallLeg terminating interface.
   Parameters: callLegSessionID



Test MMCC_IpAppMultiMediaCallLeg_68

Summary: request reference of call related to call leg

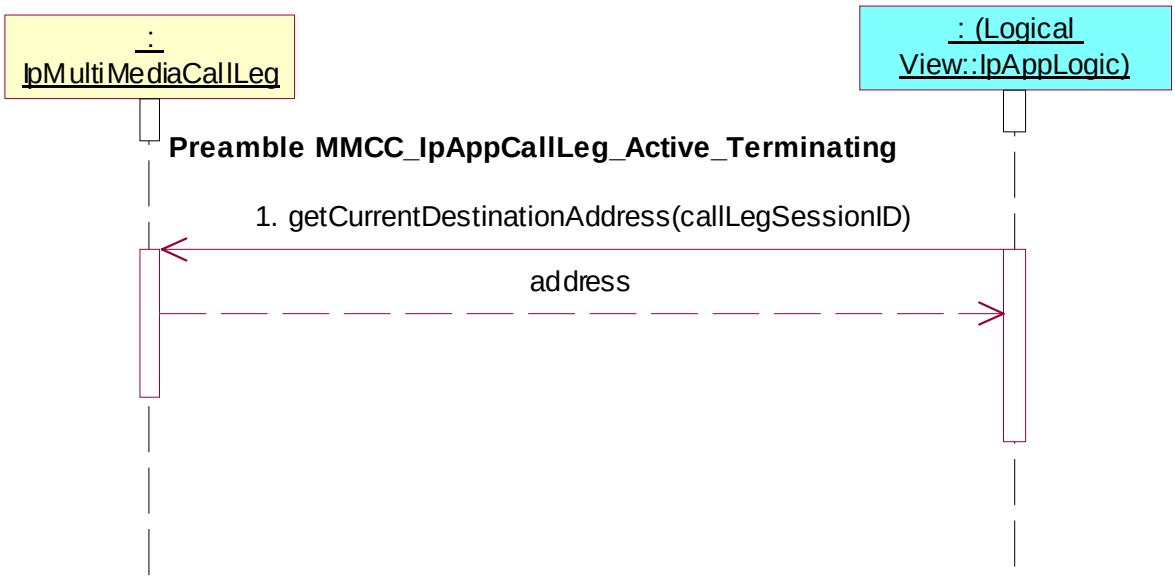
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **getCurrentDestinationAddress()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **getCurrentDestinationAddress()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID



Test MMCC_IpAppMultiMediaCallLeg_69

Summary: continue processing of call leg

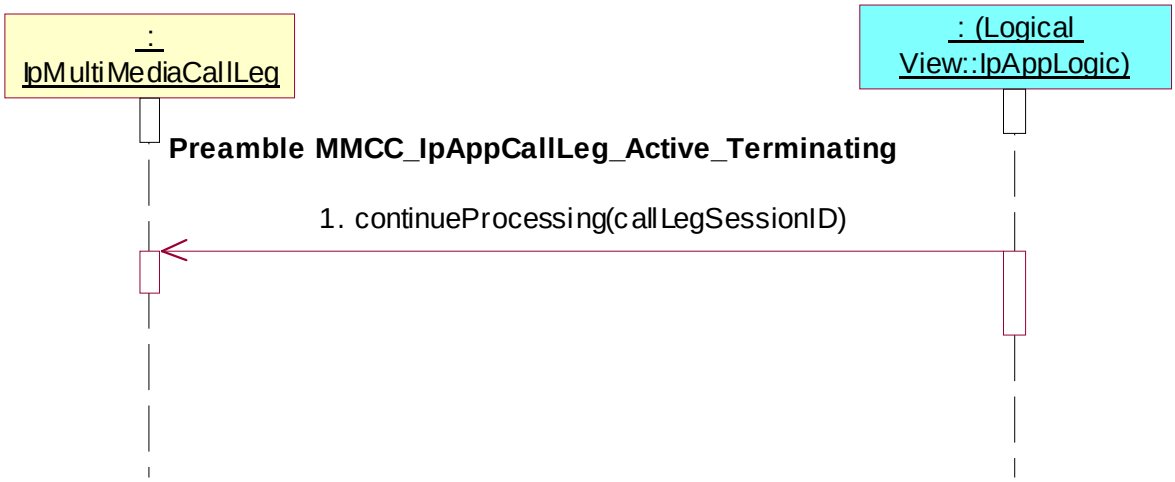
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **continueProcessing()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **continueProcessing()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID



Test MMCC_IpAppMultiMediaCallLeg_70

Summary: release call leg

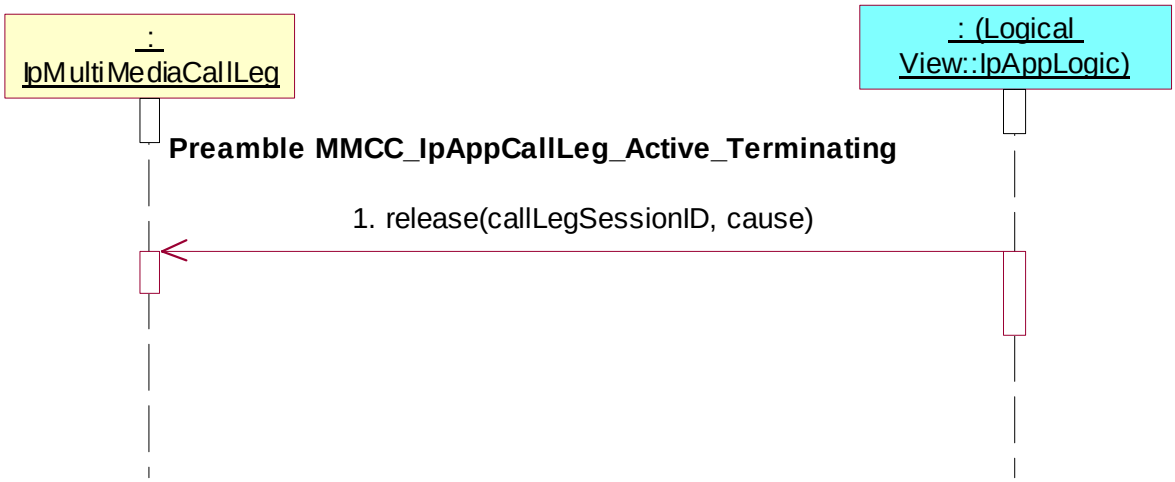
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **release()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, cause



Test MMCC_IpAppMultiMediaCallLeg_71

Summary: de-assign call leg

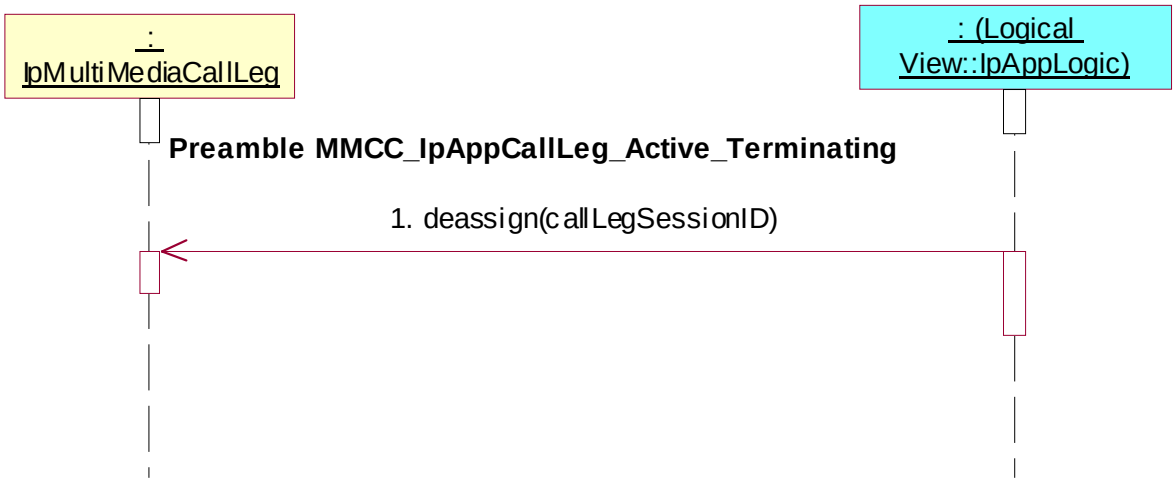
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **deassign()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **deassign()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID



Test MMCC_IpAppMultiMediaCallLeg_72

Summary: get information about call leg

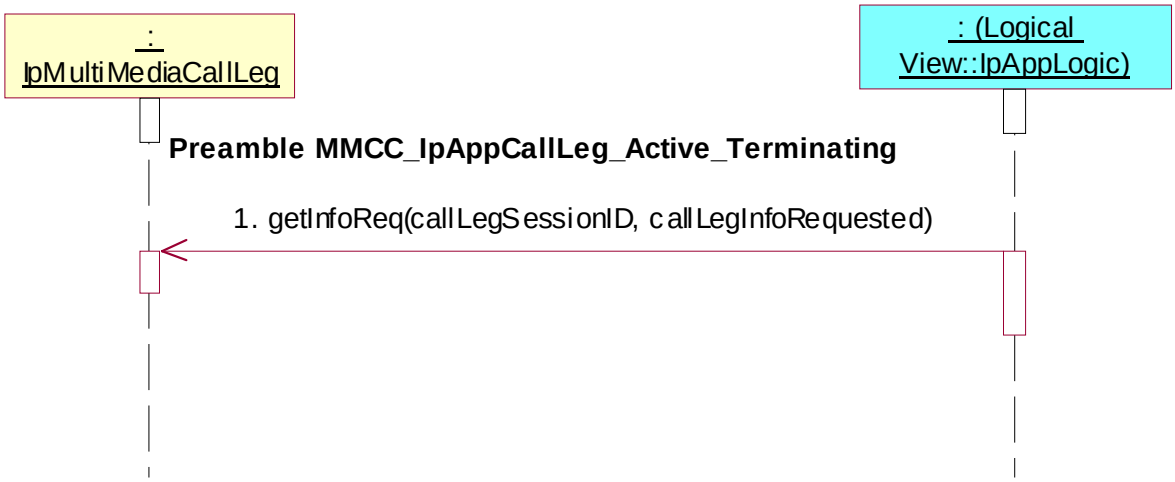
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, callLegInfoRequested



Test MMCC_IpAppMultiMediaCallLeg_73

Summary: get information about call leg, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.2

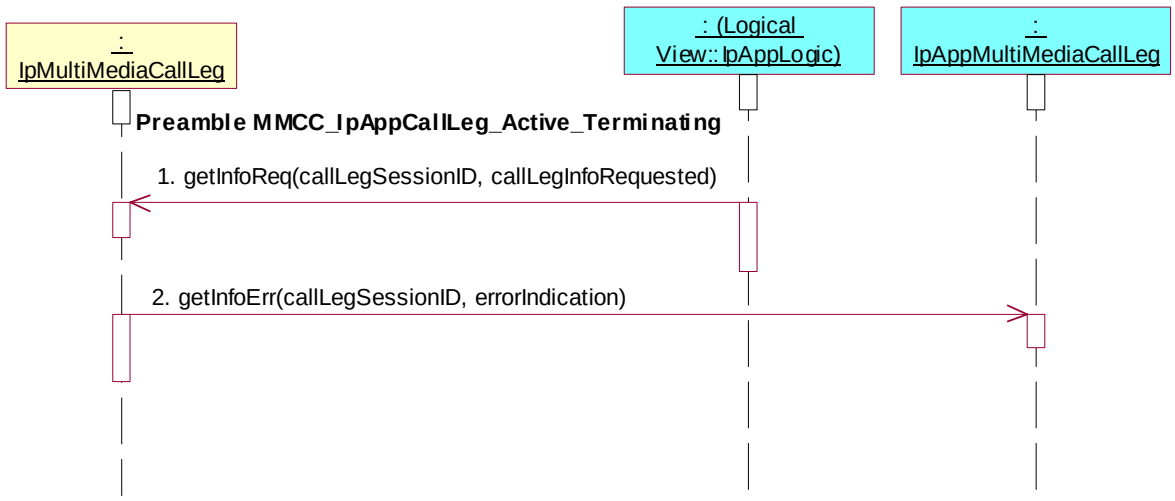
Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, callLegInfoRequested

2. Method call **getInfoErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallLeg_74

Summary: supervise call leg

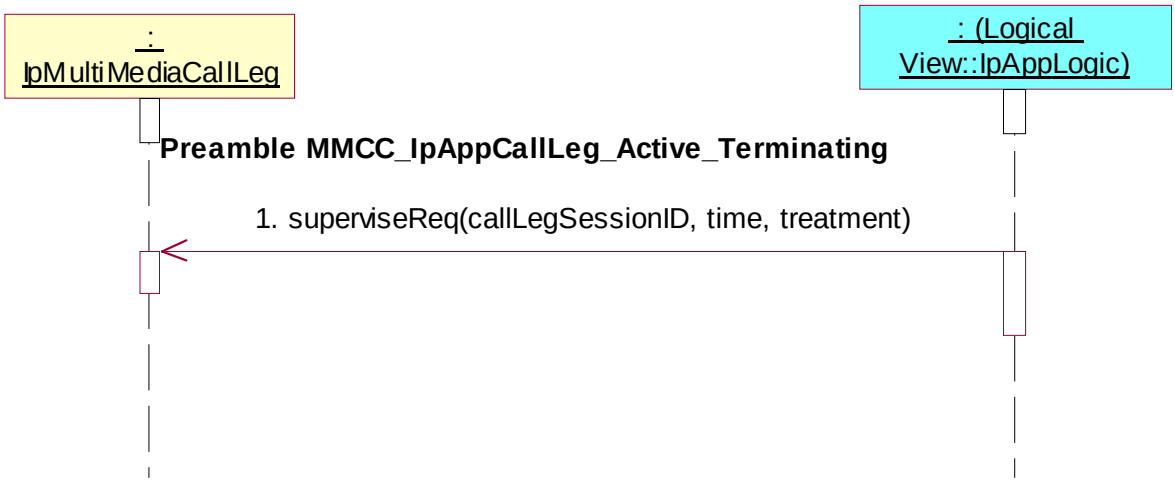
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **superviseReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, time, treatment



Test MMCC_IpAppMultiMediaCallLeg_75

Summary: supervise call leg, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.2

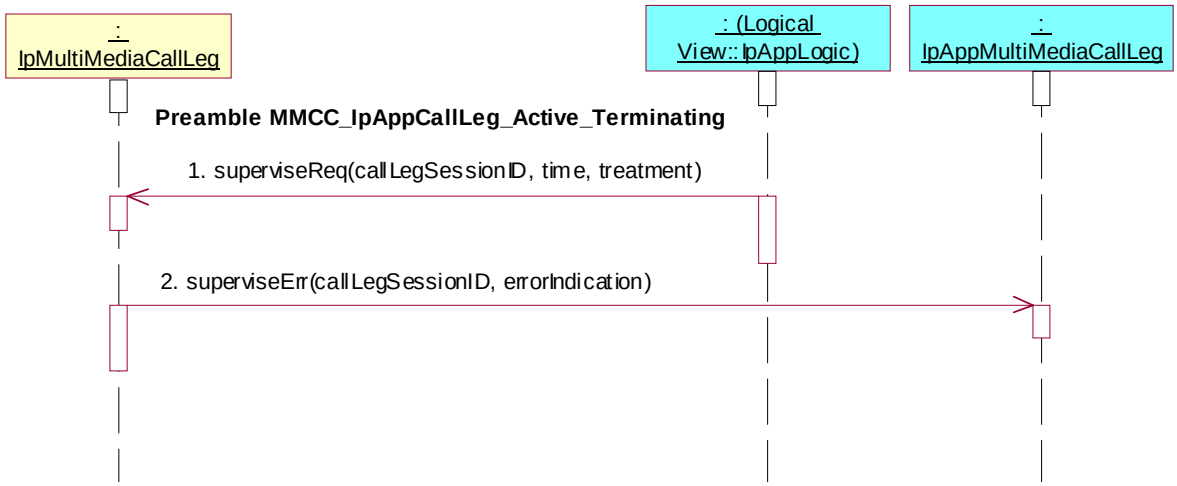
Precondition: IUT capable of invoking **superviseReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, time, treatment

2. Method call **superviseErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallLeg_76

Summary: set monitor on media streams

Reference: ES 202 915-4-3 [3], clause 7.3.2 and ES 202 915-4-4 [4], clauses 6.5 and 6.6

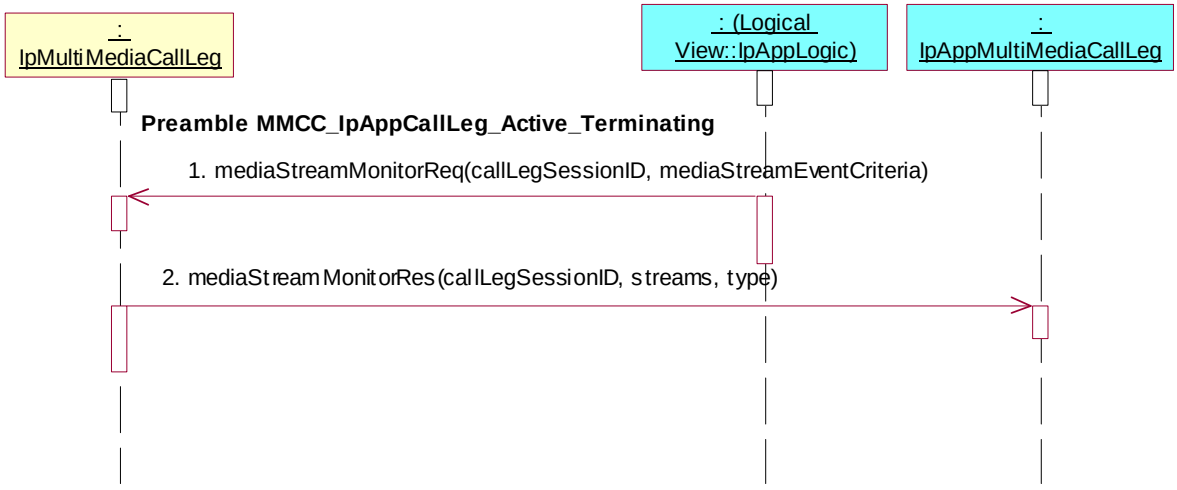
Precondition: IUT capable of invoking **mediaStreamMonitorReq()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **mediaStreamMonitorReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, mediaStreamEventCriteria

2. Method call **mediaStreamMonitorRes()**
   Parameters: callLegSessionID, streams, type
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallLeg_77

Summary: set monitor on media streams and allow setup

Reference: ES 202 915-4-3 [3], clause 7.3.2 and ES 202 915-4-4 [4], clauses 6.5 and 6.6

Precondition: IUT capable of invoking **mediaStreamMonitorReq()** and **mediaStreamAllow()**

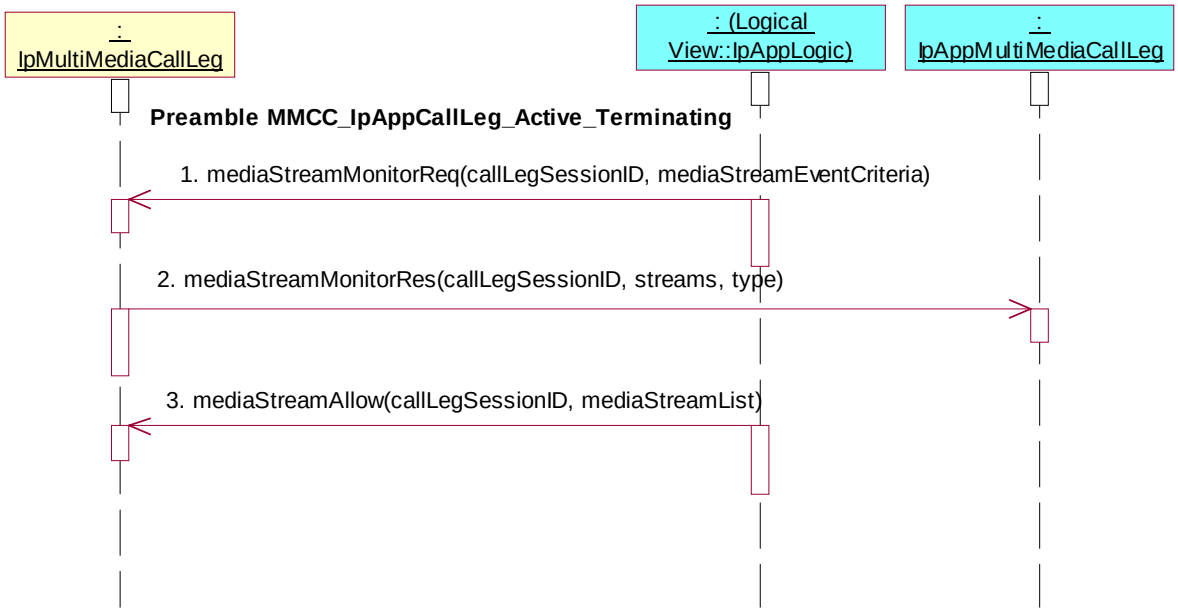
Preamble: **MMCC_IpAppMultiMediaCallLeg_Active_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **mediaStreamMonitorReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, mediaStreamEventCriteria
   mediaStreamEventCriteria.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT

2. Method call **mediaStreamMonitorRes()**
   Parameters: callLegSessionID, streams, type
   Check: no exception is returned

3. Triggered Action: cause IUT to call **mediaStreamAllow()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, mediaStreamList



Test MMCC_IpAppMultiMediaCallLeg_78

Summary: get media streams

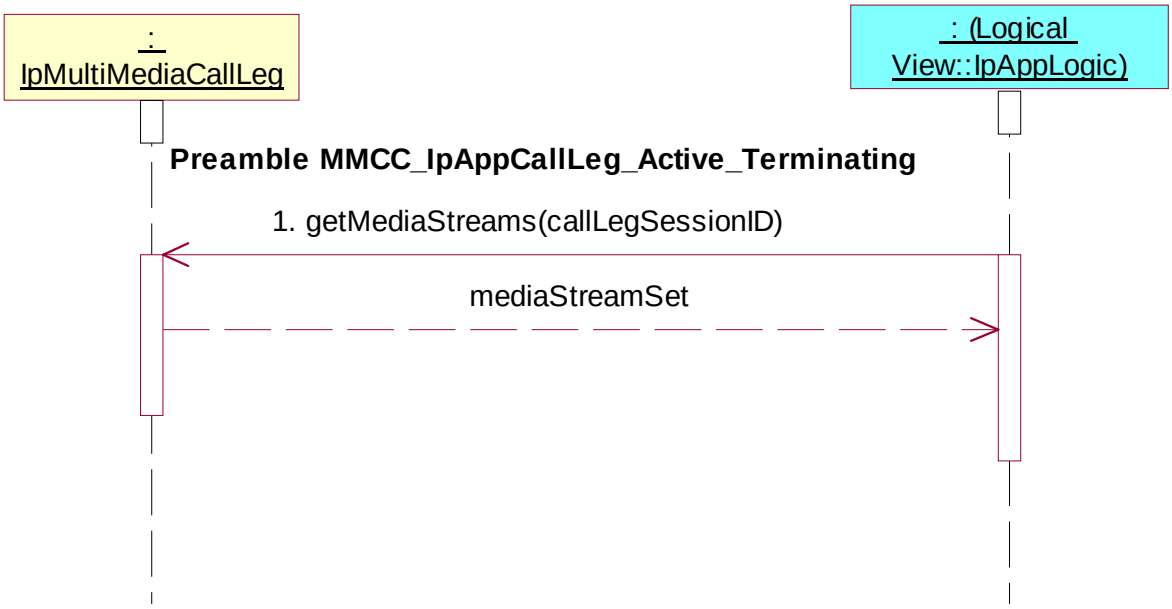
Reference: ES 202 915-4-3 [3], clause 7.3.2 and ES 202 915-4-4 [4], clauses 6.5 and 6.6

Precondition: IUT capable of invoking **getMediaStreams()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Active_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **getMediaStreams()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID



7.2.3.3.2.3 Releasing (Terminating) state

Precondition: IUT capable of invoking **eventReportReq()**, **routeReq()** and **release()**

Preamble MMCC_IpAppMultiMediaCallLeg_Releasing_Terminating

Reference: ES 202 915-4-3 [3], clause 7.3.2.3

Pre-preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpMultiMediaCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppMultiMediaCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Preamble Sequence:

1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest
   notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   Check: valid value of TpAppMultiPartyCallBack is returned

3. Triggered Action: cause IUT to call **createCallLeg()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID, AppMultiMediaCallLeg

4. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, eventsRequested

5. Triggered Action: cause IUT to call **routeReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties

6. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, cause

or

1. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCallControlManager, notificationRequest
   notificationRequest.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   notificationRequest.CallEventType= P_CALL_EVENT_TERMINATING_RELEASE

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   Check: valid value of TpAppMultiPartyCallBack is returned

or

1. Triggered Action: cause IUT to call **createCall()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCall

2. Triggered Action: cause IUT to call **createCallLeg()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID, AppMultiMediaCallLeg

3. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, eventsRequested

4. Triggered Action: cause IUT to call **routeReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties

5. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, cause

or

1. Triggered Action: cause IUT to call **enableNotifications()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCallControlManager

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   notificationInfo.CallEventInfo.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   Check: valid value of TpAppMultiMediaCallBack is returned

3. Triggered Action: cause IUT to call **createCallLeg()** method on the tester's (SCF's) IpMultiMediaCall interface.
   Parameters: callSessionID, AppMultiMediaCallLeg

4. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, eventsRequested

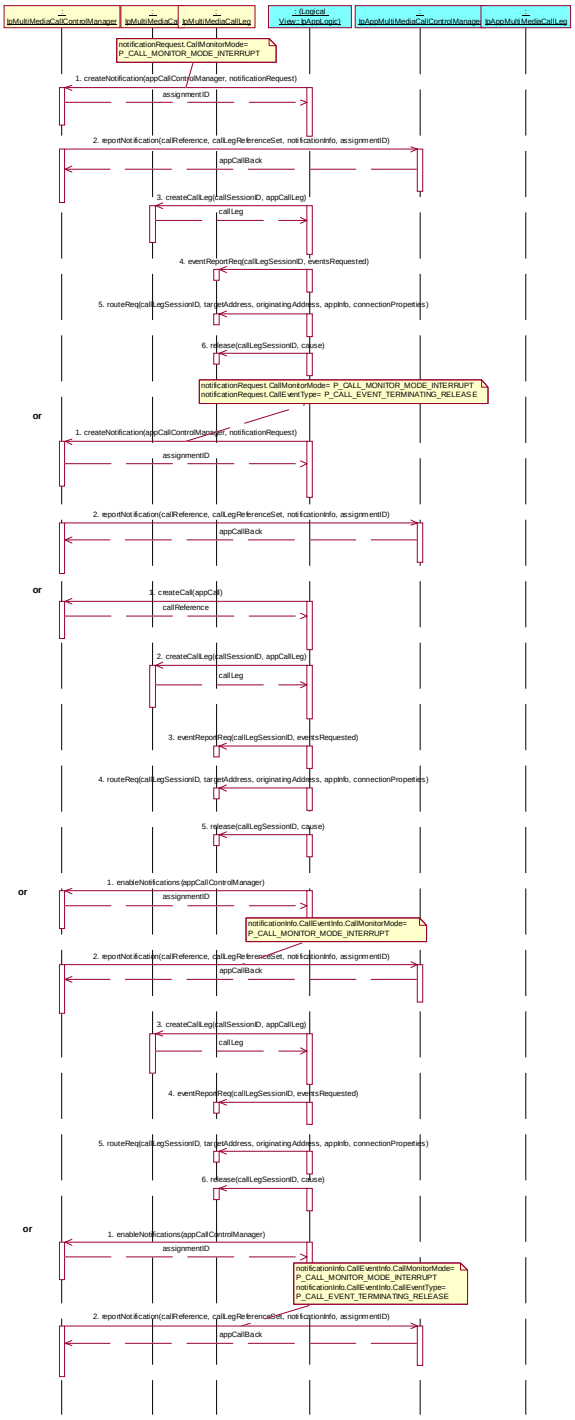
5. Triggered Action: cause IUT to call **routeReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties

6. Method call **eventReportRes()**
   Parameters: callLegSessionID, eventInfo
   Check: no exception is returned

or

1. Triggered Action: cause IUT to call **enableNotifications()** method on the tester's (SCF's) IpMultiMediaCallControlManager interface.
   Parameters: appCallControlManager

2. Method call **reportNotification()**
   Parameters: callReference, callLegReferenceSet, notificationInfo, assignmentID
   notificationInfo.CallEventInfo.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT
   notificationInfo.CallEventInfo.CallEventType= P_CALL_EVENT_TERMINATING_RELEASE
   Check: valid value of TpAppMultiMediaCallBack is returned



Test MMCC_IpAppMultiMediaCallLeg_79

Summary: request reference of call related to call leg

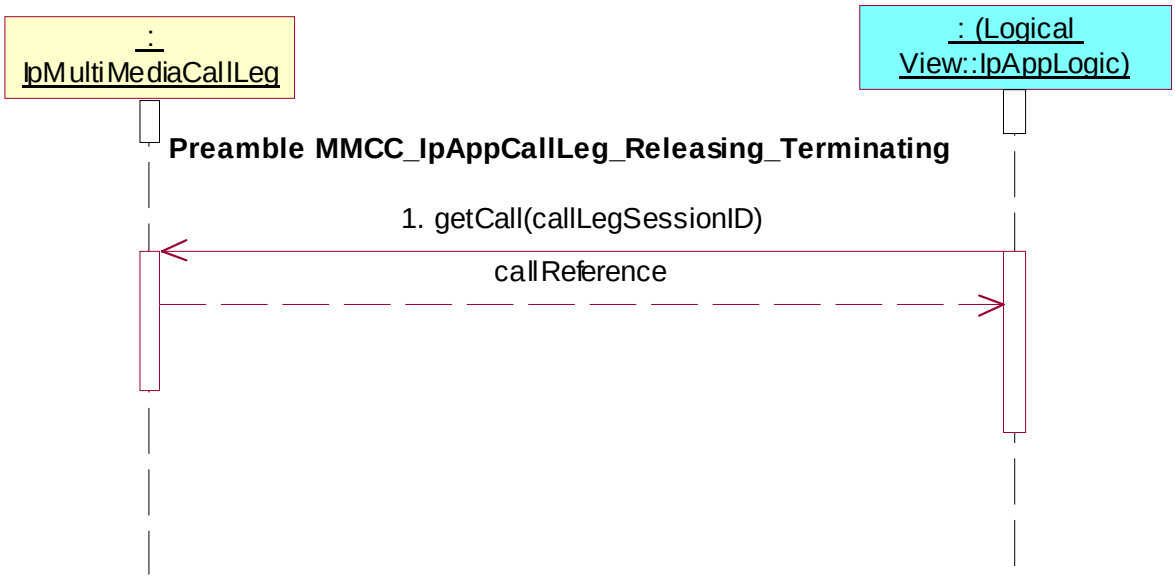
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **getCall()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Releasing_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **getCall()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID



Test MMCC_IpAppMultiMediaCallLeg_80

Summary: request reference of call related to call leg

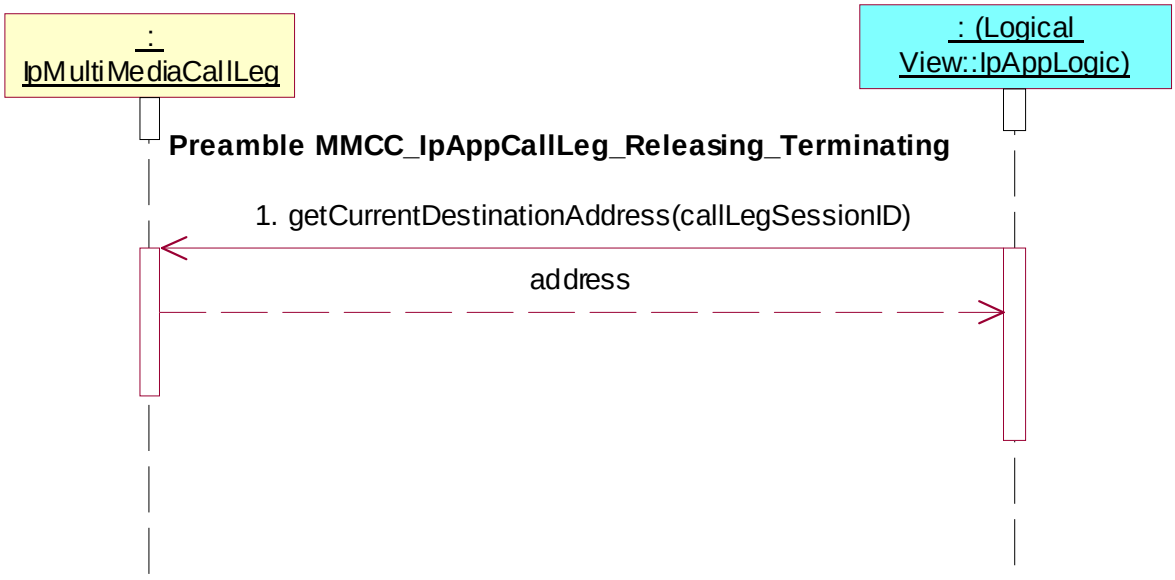
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **getCurrentDestinationAddress()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Releasing_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **getCurrentDestinationAddress()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID



Test MMCC_IpAppMultiMediaCallLeg_81

Summary: continue processing of call leg

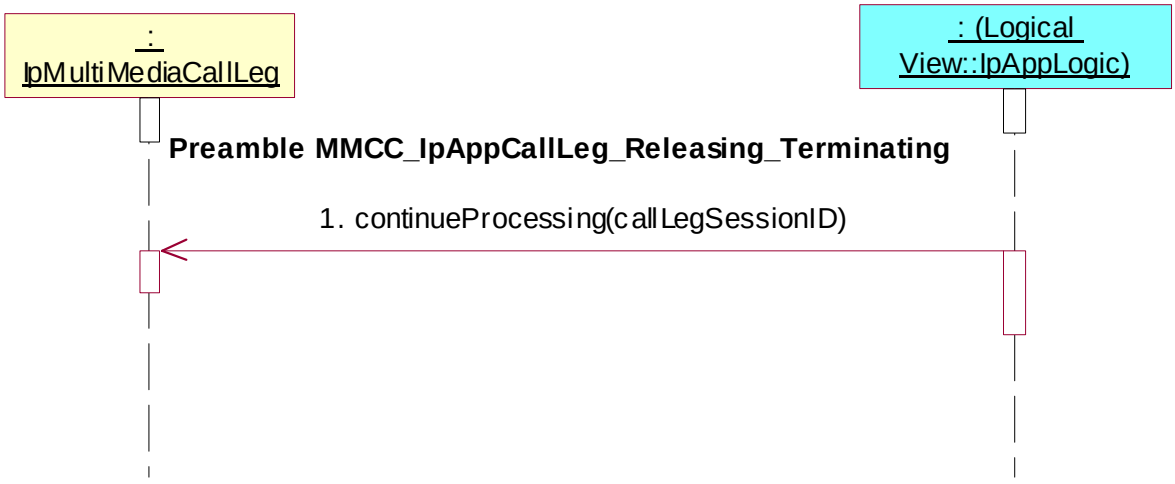
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **continueProcessing()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Releasing_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **continueProcessing()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID



Test MMCC_IpAppMultiMediaCallLeg_82

Summary: continue processing of call leg

Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **getInfoReq()**, **continueProcessing()** and **release()**

Preamble: **MMCC_IpAppCallLeg_Active_Terminating**

Test Sequence:

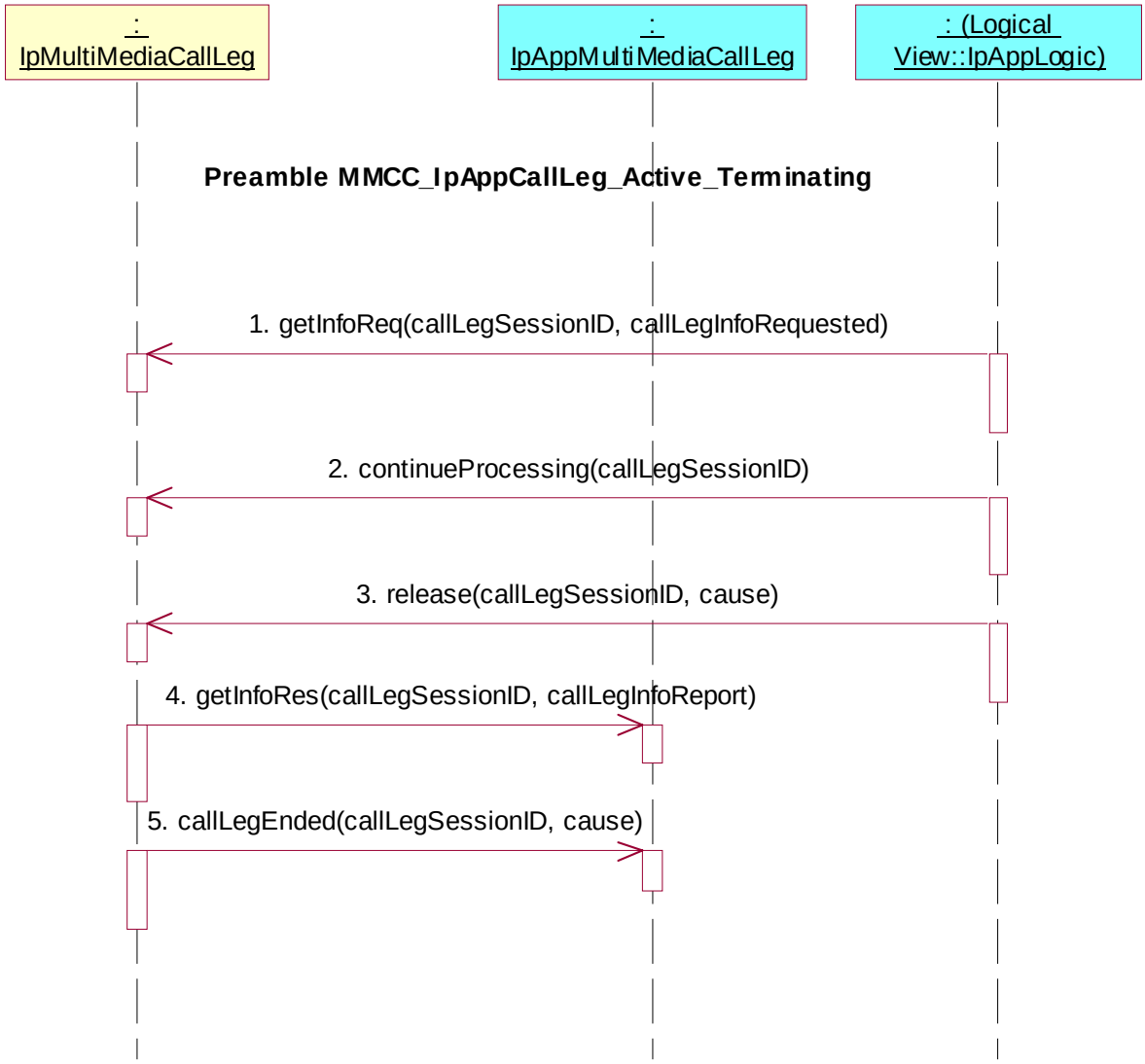
1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, callLegInfoRequested

2. Triggered Action: cause IUT to call **continueProcessing()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID

3. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, cause

4. Method call **getInfoRes()**
   Parameters: callLegSessionID, callLegInfoReport
   Check: no exception is returned

5. Method call **callLegEnded()**
   Parameters: callLegSessionID, cause
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallLeg_83

Summary: continue processing of call leg

Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **superviseReq()**, **continueProcessing()** and **release()**

Preamble: **MMCC_IpAppCallLeg_Active_Terminating**

Test Sequence:

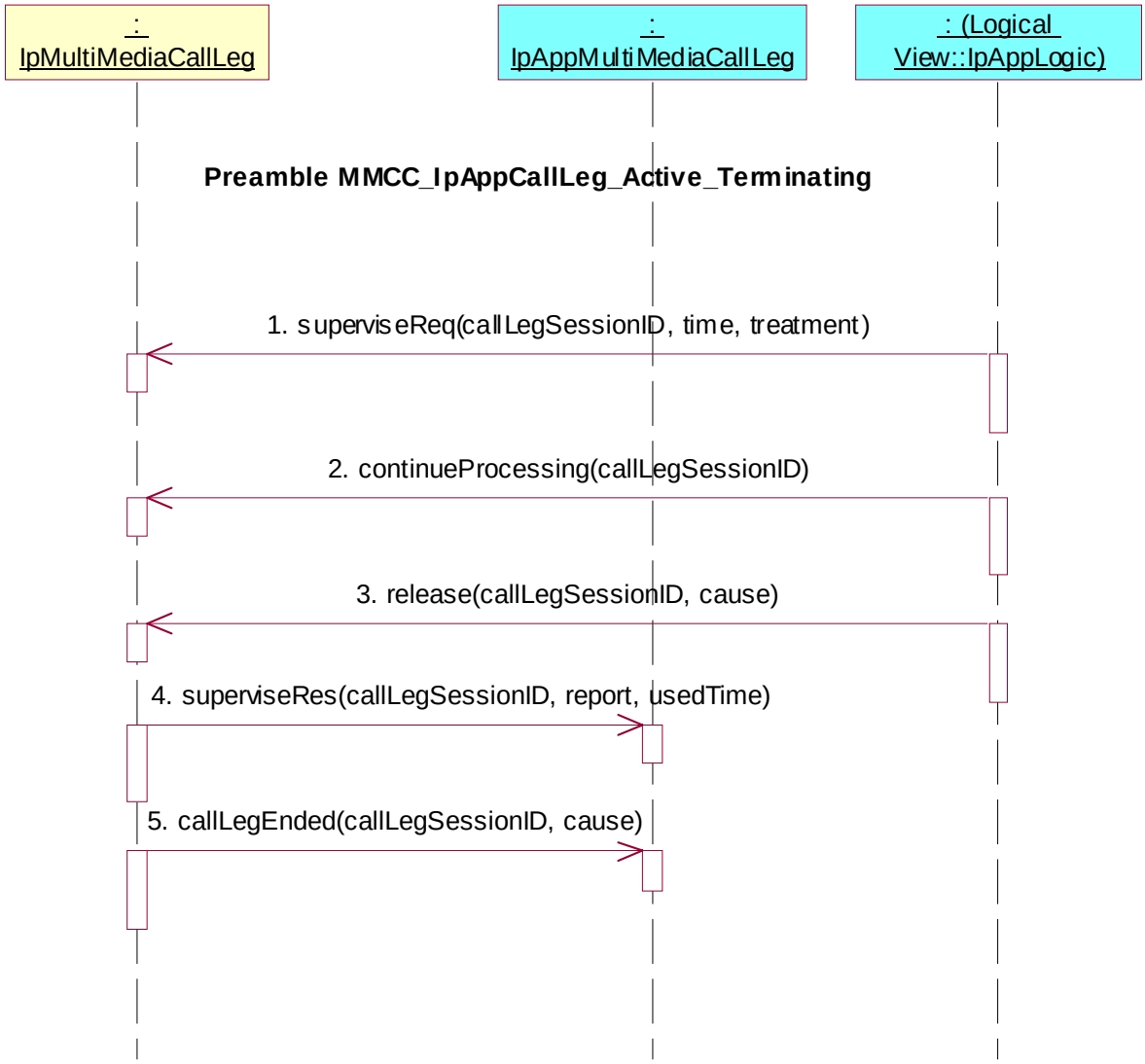
1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, time, treatment

2. Triggered Action: cause IUT to call **continueProcessing()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID

3. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, cause

4. Method call **superviseRes()**
   Parameters: callLegSessionID, report, usedTime
   Check: no exception is returned

5. Method call **callLegEnded()**
   Parameters: callLegSessionID, cause
   Check: no exception is returned



Test MMCC_IpAppMultiMediaCallLeg_84

Summary: de-assign call leg

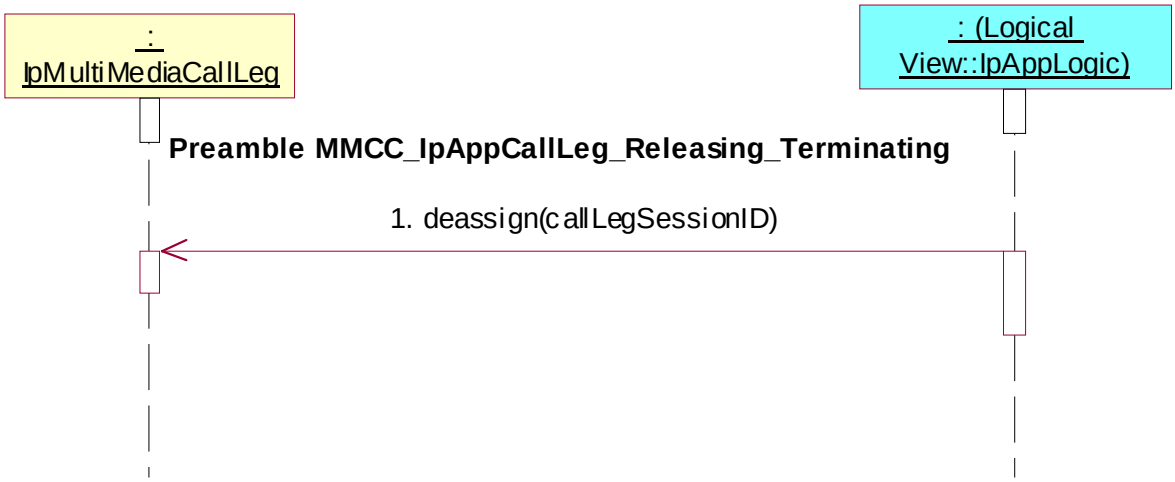
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **deassign()**

Preamble: **MMCC_IpAppMultiMediaCallLeg_Releasing_Terminating**

Test Sequence:

1. Triggered Action: cause IUT to call **deassign()** method on the tester's (SCF's) terminating IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID



7.2.3.4 IpMultiMediaStream

Test MMCC_IpAppMultiMediaStream_01

Summary: substract media stream

Reference: ES 202 915-4-4 [4], clause 6.7

Precondition: IUT capable of invoking **mediaStreamMonitorReq()** and **substract()**

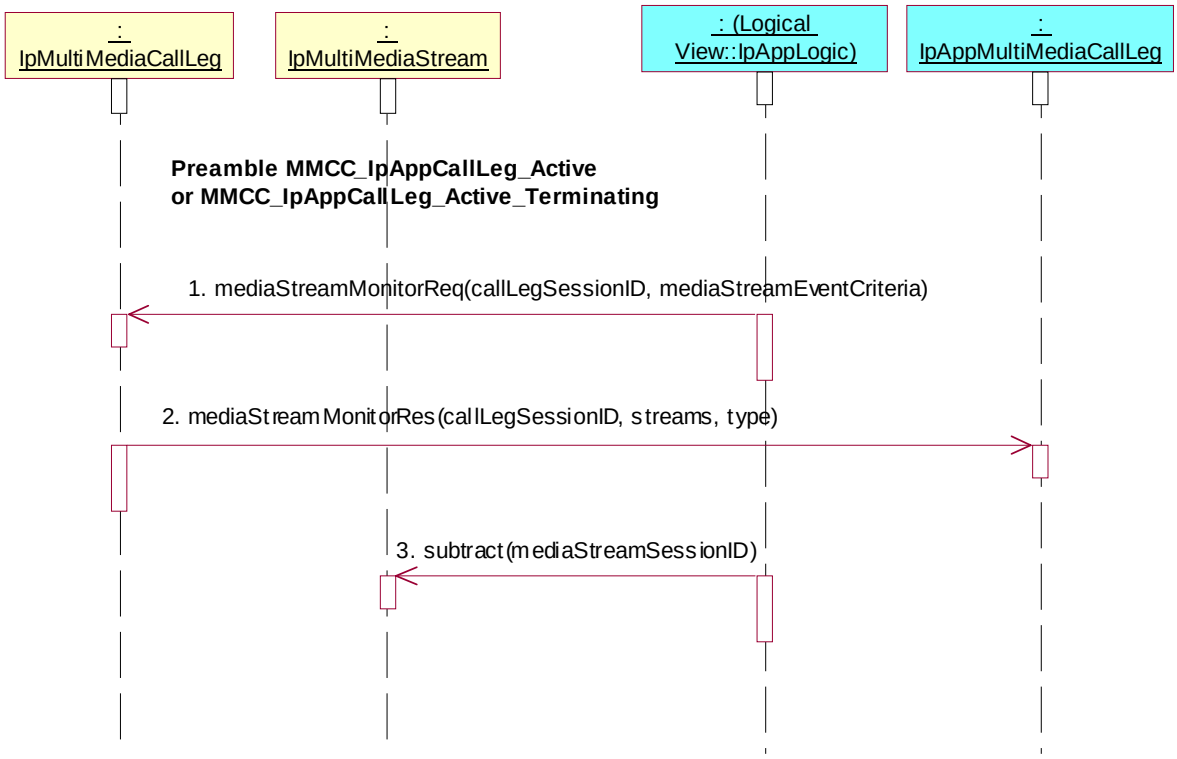
Preamble: **MMCC_IpAppMultiMediaCallLeg_Active** or
MMCC_IpAppMultiMediaCallLeg_Active_Terminating

Test Sequence:

1. Triggered Action: cause IUT to call **mediaStreamMonitorReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, mediaStreamEventCriteria

2. Method call **mediaStreamMonitorRes()**
   Parameters: callLegSessionID, streams, type
   Check: no exception is returned

3. Triggered Action: cause IUT to call **substract()** method on the tester's (SCF's) IpMultiMediaStream interface.
   Parameters: mediaStreamSessionID



7.2.4 Conference Call Control Service (CCC)

The TPs in this clause are based on ES 202 915-4-5 [5].

7.2.4.1 IpAppCallControlManager

Test CCC_IpAppConfCallControlManager_01

Summary: create conference

Reference: ES 202 915-4-5 [5], clause 6.1

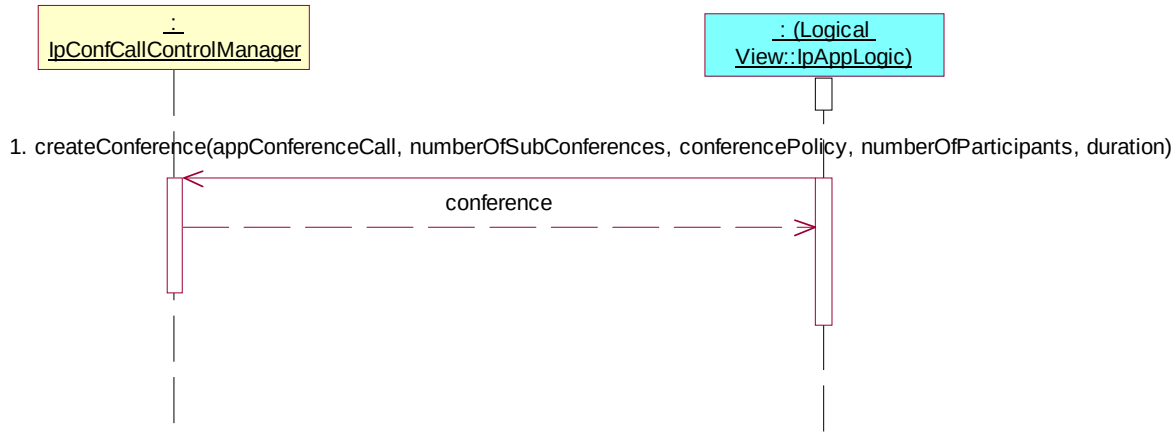
Precondition: IUT capable of invoking **createConference()**

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

The application is permitted to provide its IpAppConferenceCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **createConference()** method on the tester's (SCF's) IpConfCallControlManager interface.
   Parameters: appConferenceCall, numberOfSubConferences, conferencePolicy, numberOfParticipants, duration



Test CCC_IpAppConfCallControlManager_02

Summary: reserve conference resources and accept conference created notification

Reference: ES 202 915-4-5 [5], clauses 6.1 and 6.2

Precondition: IUT capable of invoking **reserveResources()**; **conferenceCreated()** implemented

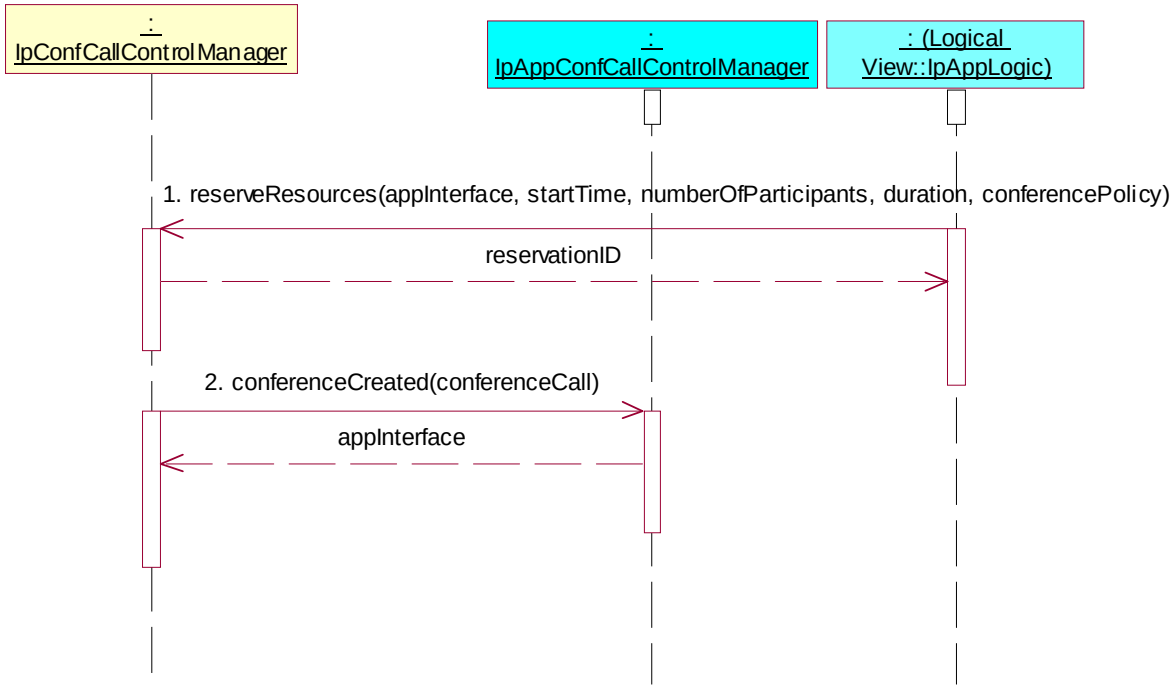
Preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpConfCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppConferenceCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **reserveResources()** method on the tester's (SCF's) IpConfCallControlManager interface.
   Parameters: appInterface, startTime, numberOfParticipants, duration, conferencePolicy

2. Method call **conferenceCreated()**
   Parameters: conferenceCall
   Check: valid value of IpAppConfCallRef is returned



Test CCC_IpAppConfCallControlManager_03

Summary: reserve and free conference resources

Reference: ES 202 915-4-5 [5], clause 6.1

Precondition: IUT capable of invoking **reserveResources()** and **freeResources()**

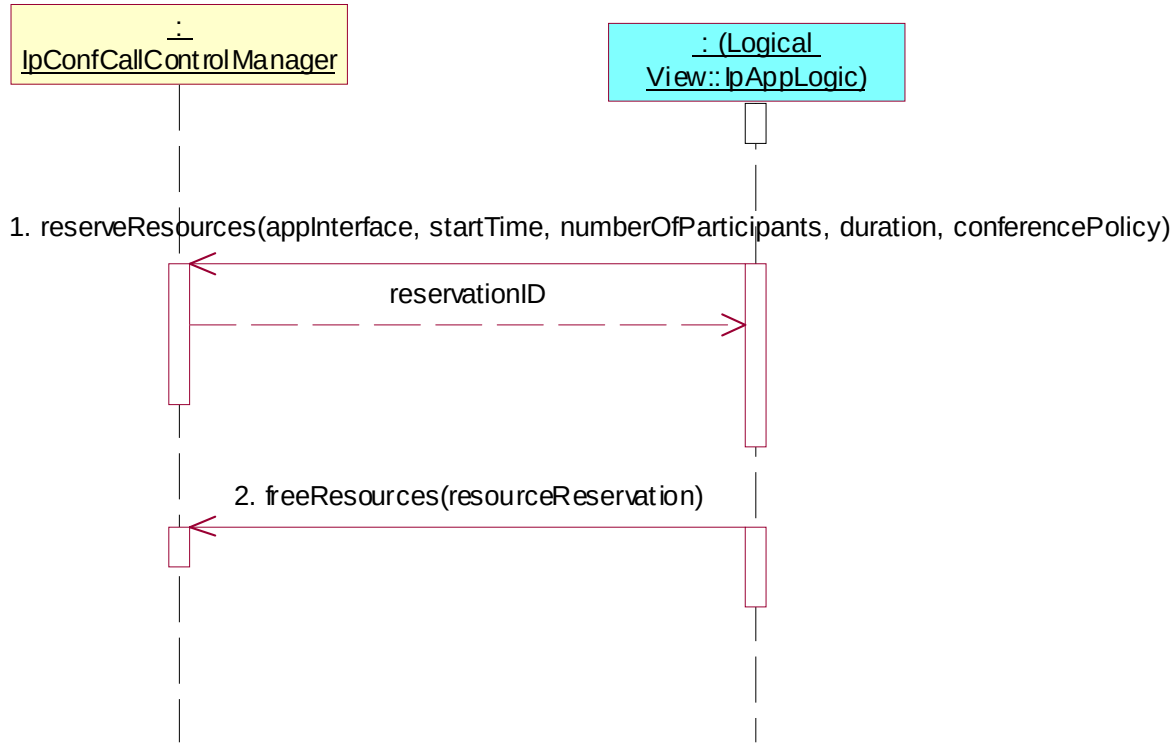
Preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpConfCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppConferenceCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **reserveResources()** method on the tester's (SCF's) IpConfCallControlManager interface.
   Parameters: appInterface, startTime, numberOfParticipants, duration, conferencePolicy

2. Triggered Action: cause IUT to call **freeResources()** method on the tester's (SCF's) IpConfCallControlManager interface.
   Parameters: resourceReservation



Test CCC_IpAppConfCallControlManager_04

Summary: check conference resources

Reference: ES 202 915-4-5 [5], clause 6.1

Precondition: IUT capable of invoking **checkResources()**

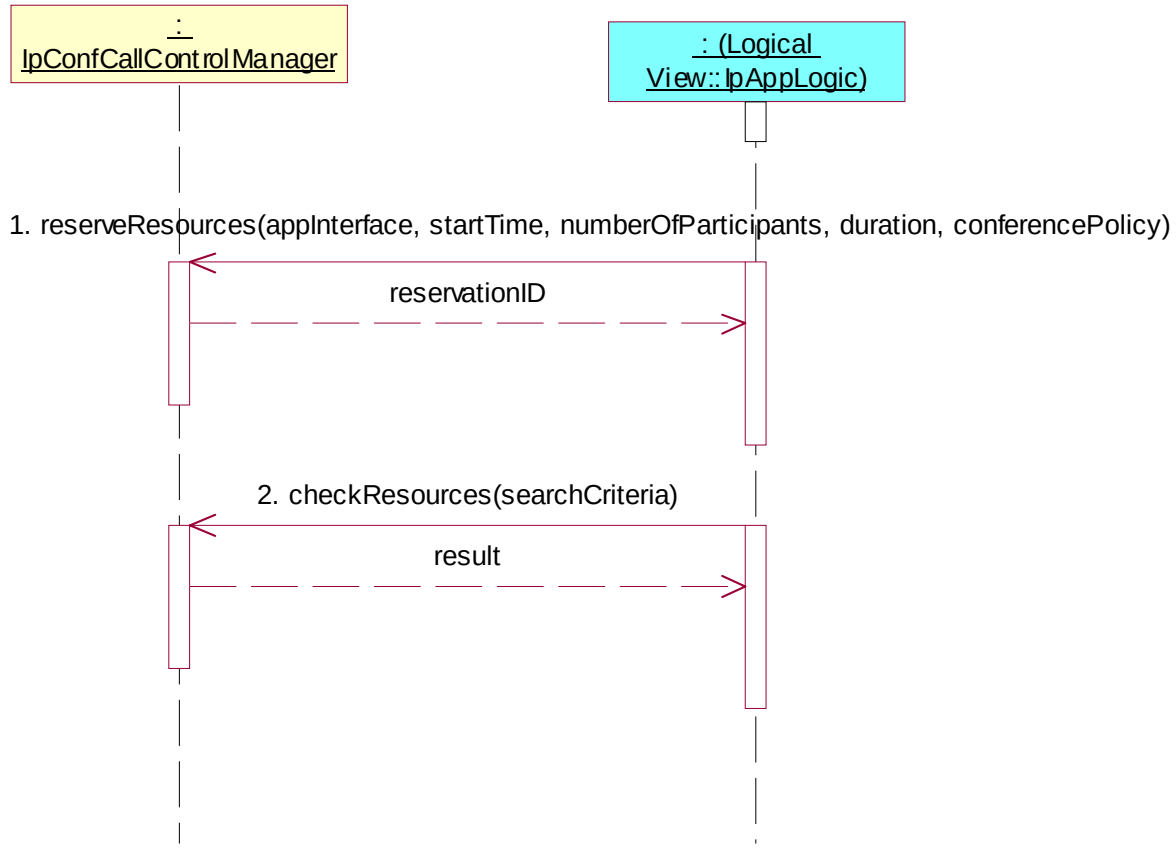
Preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpConfCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppConferenceCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Test Sequence:

1. Triggered Action: cause IUT to call **reserveResources()** method on the tester's (SCF's) IpConfCallControlManager interface.
   Parameters: appInterface, startTime, numberOfParticipants, duration, conferencePolicy

2. Triggered Action: cause IUT to call **checkResources()** method on the tester's (SCF's) IpConfCallControlManager interface.
   Parameters: searchCriteria



7.2.4.2 IpAppConfCall

Applications need not be capable of performing each of the sequences below, even if they support the methods indicated below.

Reference: ES 202 915-4-5 [5], clauses 6.3 and 6.4

Precondition: IUT capable of invoking **reserveResources()** or **createConference()**

Preamble CCC_IpAppConfCall_Conference_Active

Reference: ES 202 915-4-5 [5], clauses 6.1 and 6.2

Pre-preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpConfCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppConferenceCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

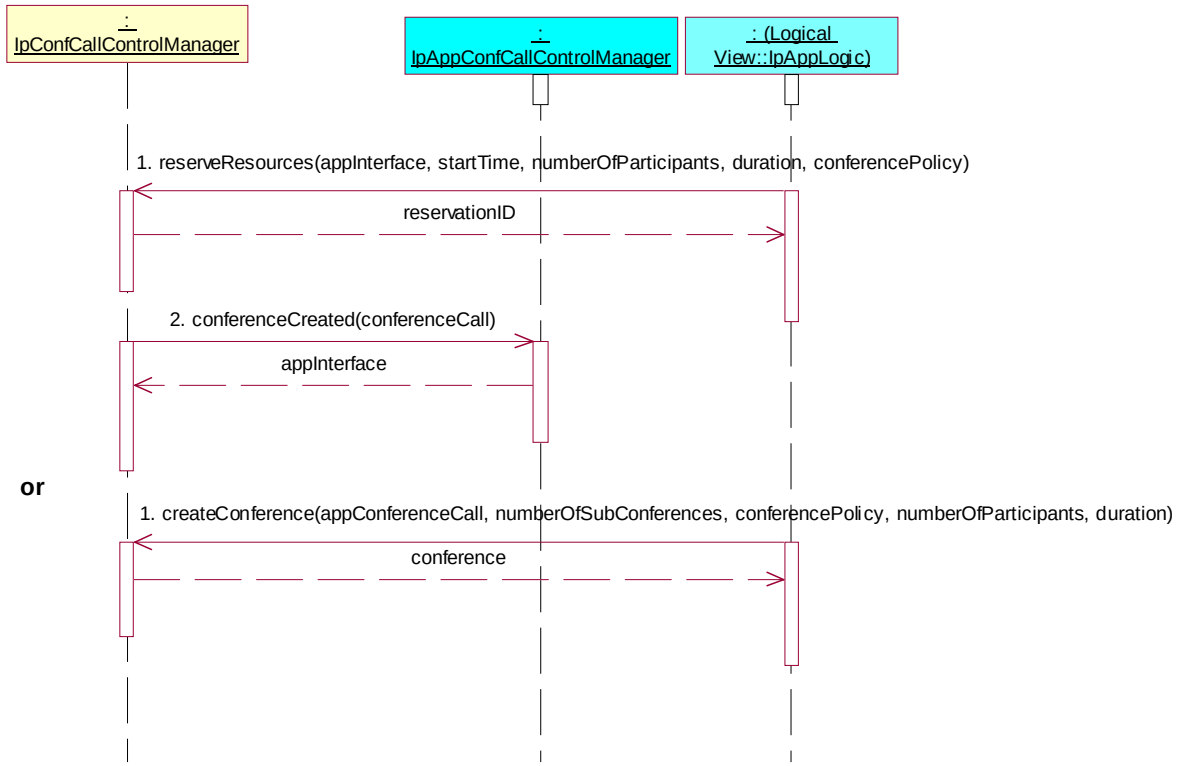
Preamble Sequence:

1. Triggered Action: cause IUT to call **reserveResources()** method on the tester's (SCF's) IpConfCallControlManager interface.
   Parameters: appInterface, startTime, numberOfParticipants, duration, conferencePolicy

2. Method call **conferenceCreated()**
   Parameters: conferenceCall
   Check: valid value of IpAppConfCallRef is returned

or

1. Triggered Action: cause IUT to call **createConference()** method on the tester's (SCF's) IpConfCallControlManager interface.
   Parameters: appConferenceCall, numberOfSubConferences, conferencePolicy, numberOfParticipants, duration



Test CCC_IpAppConfCall_01

Summary: get subconferences

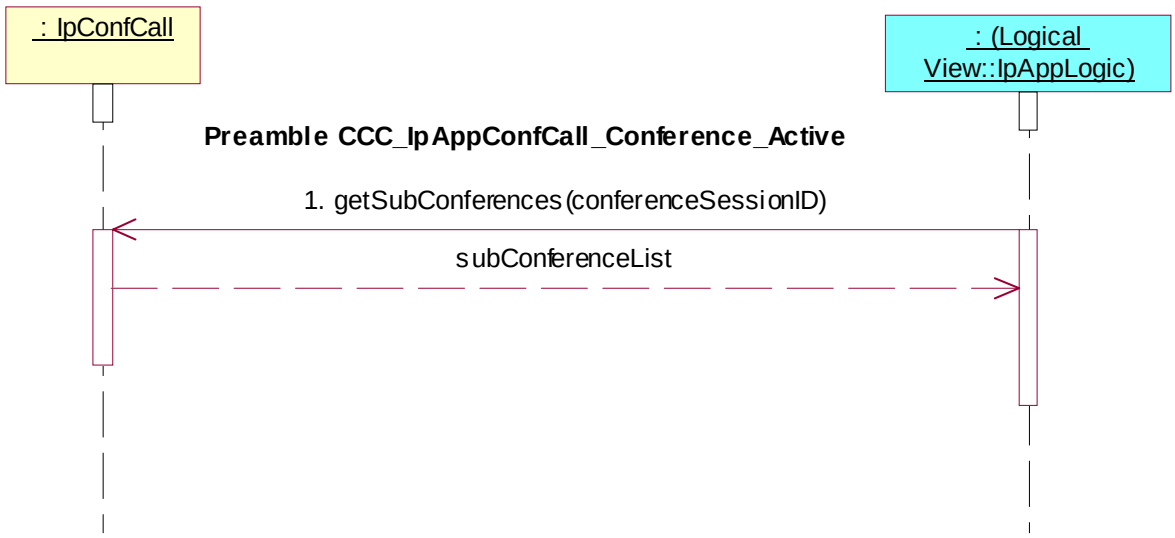
Reference: ES 202 915-4-5 [5], clause 6.3

Precondition: IUT capable of invoking **getSubConferences()**

Preamble: **CCC_IpAppConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **getSubConferences()** method on the tester's (SCF's) IpConfCall interface.
   Parameters: conferenceSessionID



Test CCC_IpAppConfCall_02

Summary: create subconference

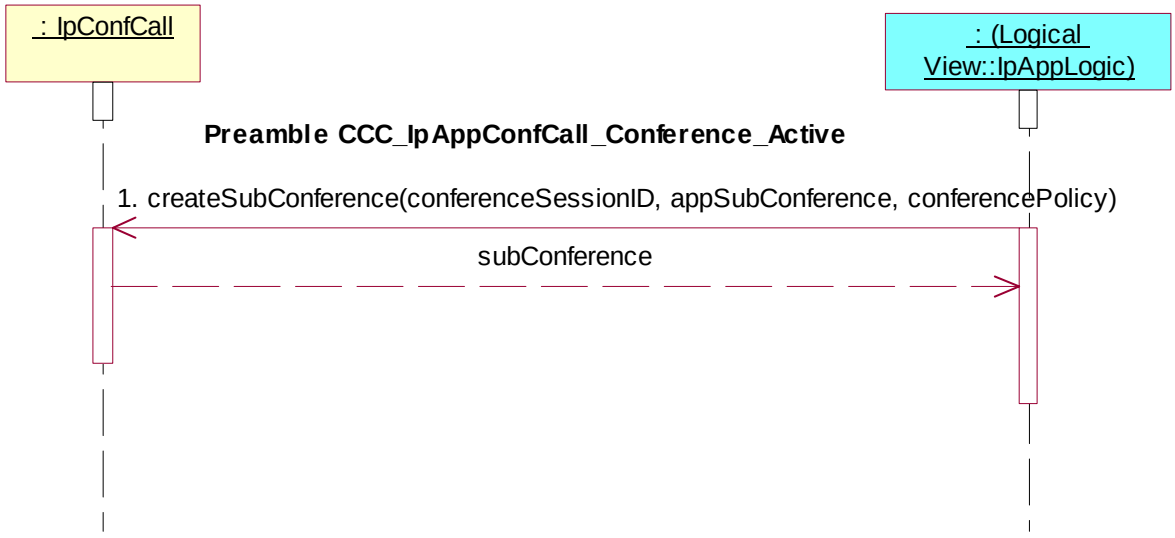
Reference: ES 202 915-4-5 [5], clause 6.3

Precondition: IUT capable of invoking **createSubConference()**

Preamble: **CCC_IpAppConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **createSubConference()** method on the tester's (SCF's) IpConfCall interface.
   Parameters: conferenceSessionID, appSubConference, conferencePolicy



Test CCC_IpAppConfCall_03

Summary: request and accept leave notifications

Reference: ES 202 915-4-5 [5], clauses 6.3 and 6.4

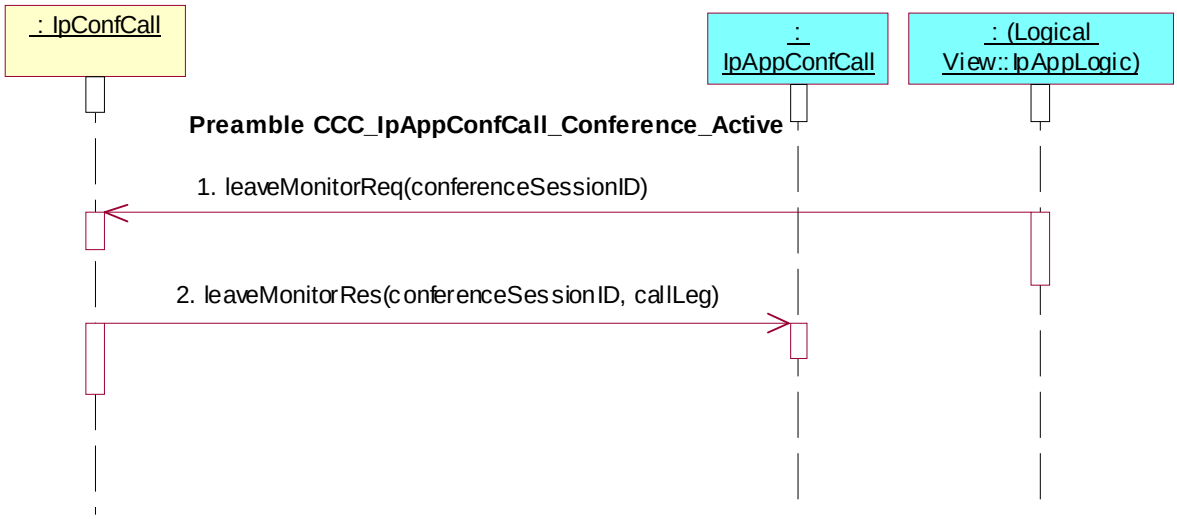
Precondition: IUT capable of invoking **leaveMonitorReq()**

Preamble: **CCC_IpAppConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **leaveMonitorReq()** method on the tester's (SCF's) IpConfCall interface.
   Parameters: conferenceSessionID

2. Method call **leaveMonitorRes()**
   Parameters: conferenceSessionID, callLeg
   Check: valid value of IpAppConfCallRef is returned



Test CCC_IpAppConfCall_04

Summary: accept new party indication

Reference: ES 202 915-4-5 [5], clause 6.4

Precondition: **partyJoined()** implemented

Preamble: **CCC_IpAppConfCall_Conference_Active** with conferencePolicy.JoinAllowed = TRUE

Test Sequence:

1. Method call **partyJoined()**
   Parameters: conferenceSessionID, callLeg, eventInfo
   Check: valid value of mpccs:IpAppCallLegRef is returned

Test CCC_IpAppConfCall_05

Summary: release call

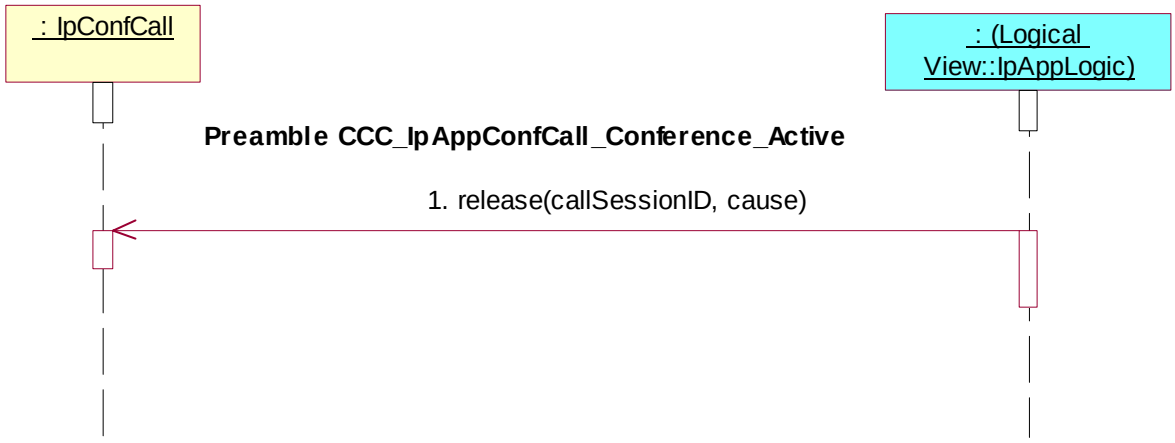
Reference: ES 202 915-4-3 [3], clause 7.2.2 and ES 202 915-4-5 [5], clause 6.3

Precondition: IUT capable of invoking **release()**

Preamble: **CCC_IpAppConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) IpConfCall interface.
   Parameters: callSessionID, cause



Test CCC_IpAppConfCall_06

Summary: deassign call

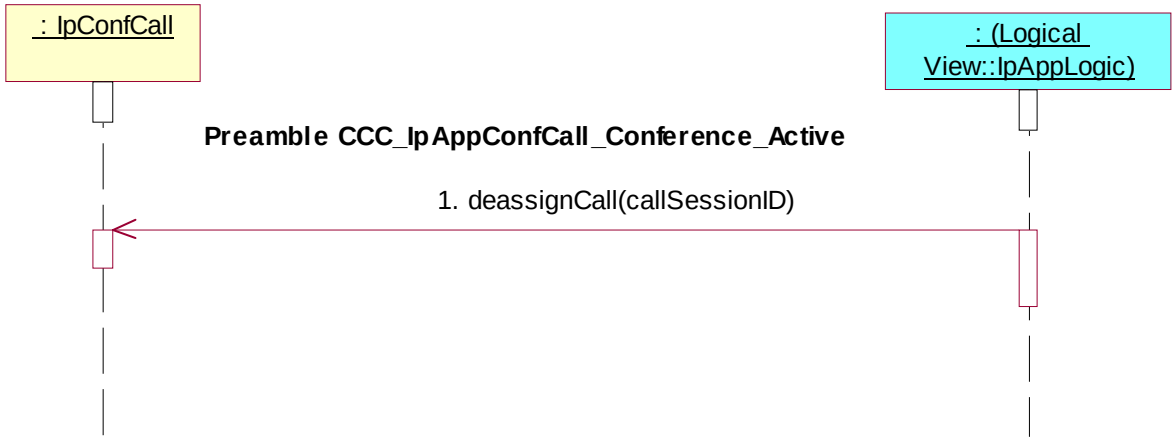
Reference: ES 202 915-4-3 [3], clause 7.2.2 and ES 202 915-4-5 [5], clause 6.5

Precondition: IUT capable of invoking **deassignCall()**

Preamble: **CCC_IpAppConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **deassignCall()** method on the tester's (SCF's) IpConfCall interface.
   Parameters: callSessionID



Test CCC_IpAppConfCall_07

Summary: supervise call

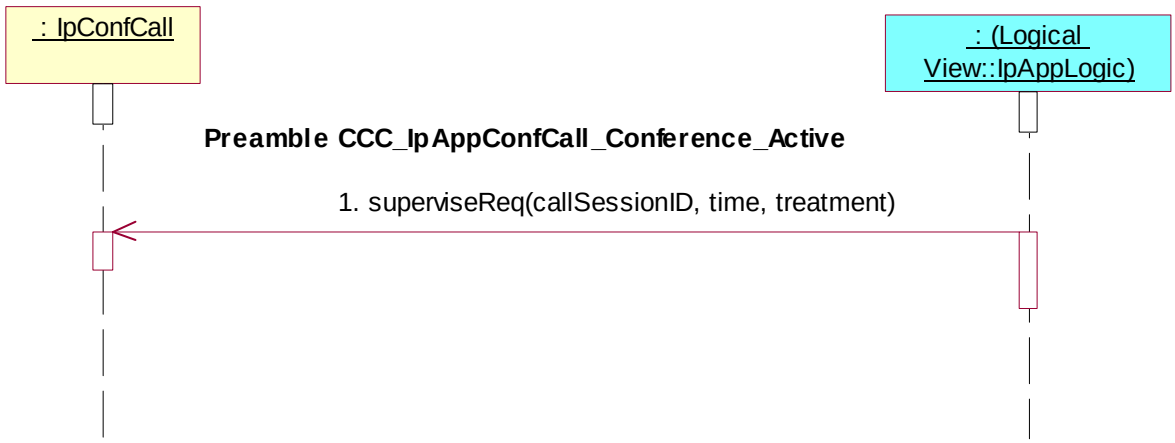
Reference: ES 202 915-4-3 [3], clause 7.2.2 and ES 202 915-4-5 [5], clause 6.3

Precondition: IUT capable of invoking **superviseReq()**

Preamble: **CCC_IpAppConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) IpConfCall interface.
   Parameters: callSessionID, time, treatment



Test CCC_IpAppConfCall_08

Summary: supervise call, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.2.2 and ES 202 915-4-5 [5], clause 6.3

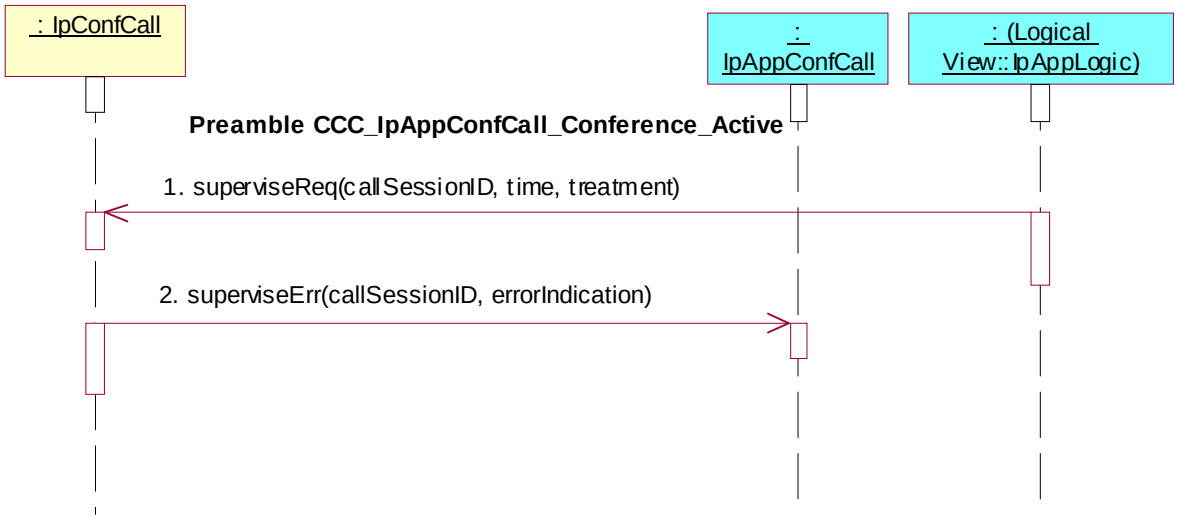
Precondition: IUT capable of invoking **superviseReq()**

Preamble: **CCC_IpAppConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) IpConfCall interface.
   Parameters: callSessionID, time, treatment

2. Method call **superviseErr()**
   Parameters: callSessionID, errorIndication
   Check: no exception is returned



Test CCC_IpAppConfCall_09

Summary: supervise call with granted volume

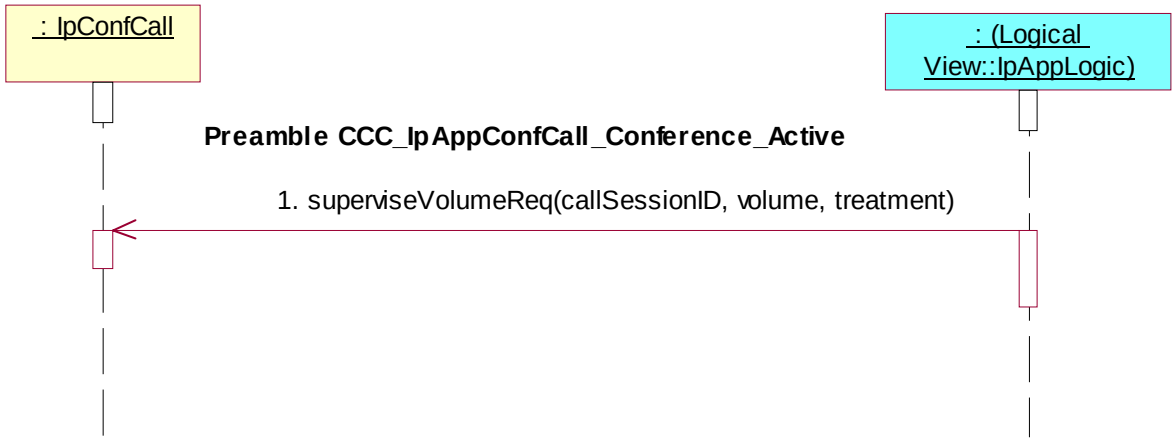
Reference: ES 202 915-4-3 [3], clause 7.2.2 and ES 202 915-4-5 [5], clause 6.3

Precondition: IUT capable of invoking**superviseVolumeReq()**

Preamble: **CCC_IpAppConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseVolumeReq()** method on the tester's (SCF's) IpConfCall interface.
   Parameters: callSessionID, volume, treatment



Test CCC_IpAppConfCall_10

Summary: supervise call with granted volume

Reference: ES 202 915-4-4 [4], clause 6.3 and 6.4 and ES 202 915-4-5 [5], clause 6.3

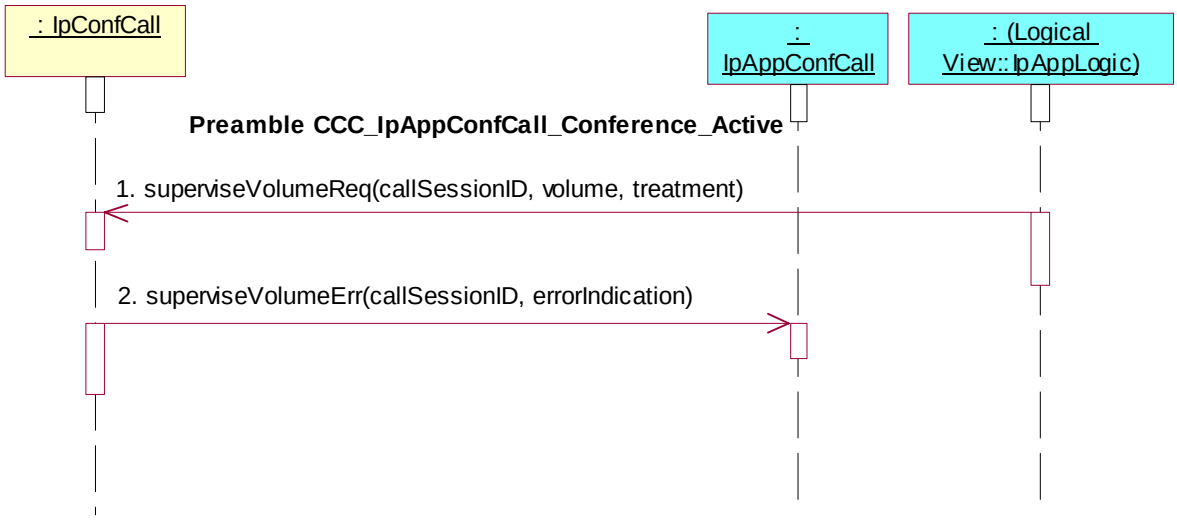
Precondition: IUT capable of invoking **superviseVolumeReq()**

Preamble: **CCC_IpAppConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseVolumeReq()** method on the tester's (SCF's) IpConfCall interface.
   Parameters: callSessionID, volume, treatment

2. Method call **superviseVolumeErr()**
   Parameters: callSessionID, errorIndication
   Check: no exception is returned



Test CCC_IpAppConfCall_11

Summary: request call information

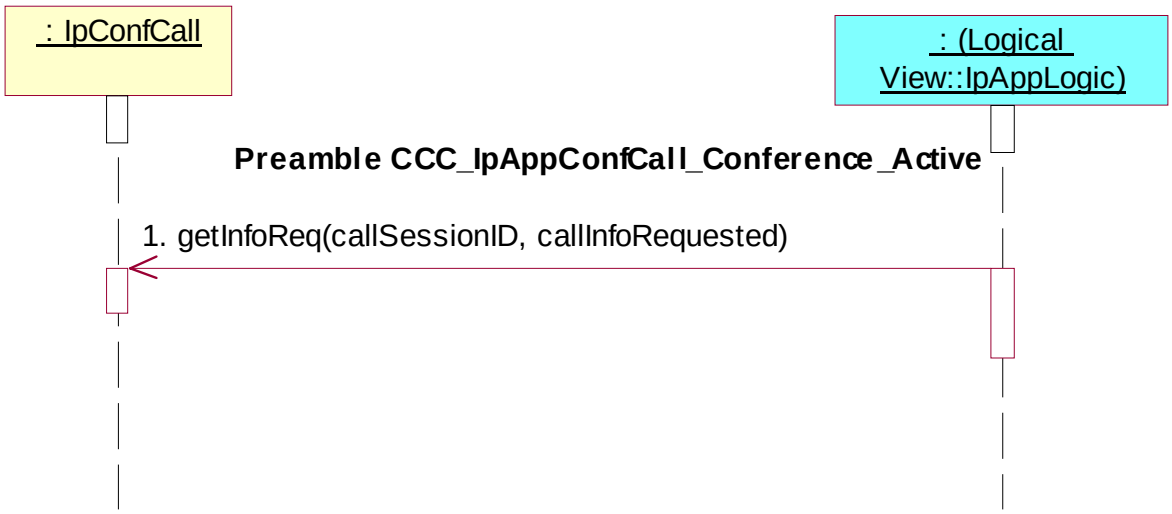
Reference: ES 202 915-4-3 [3], clause 7.2.2 and ES 202 915-4-5 [5], clause 6.3

Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **CCC_IpAppConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) IpConfCall interface.
   Parameters: callSessionID, callInfoRequested



Test CCC_IpAppConfCall_12

Summary: request call information, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.2.2 and ES 202 915-4-5 [5], clause 6.3

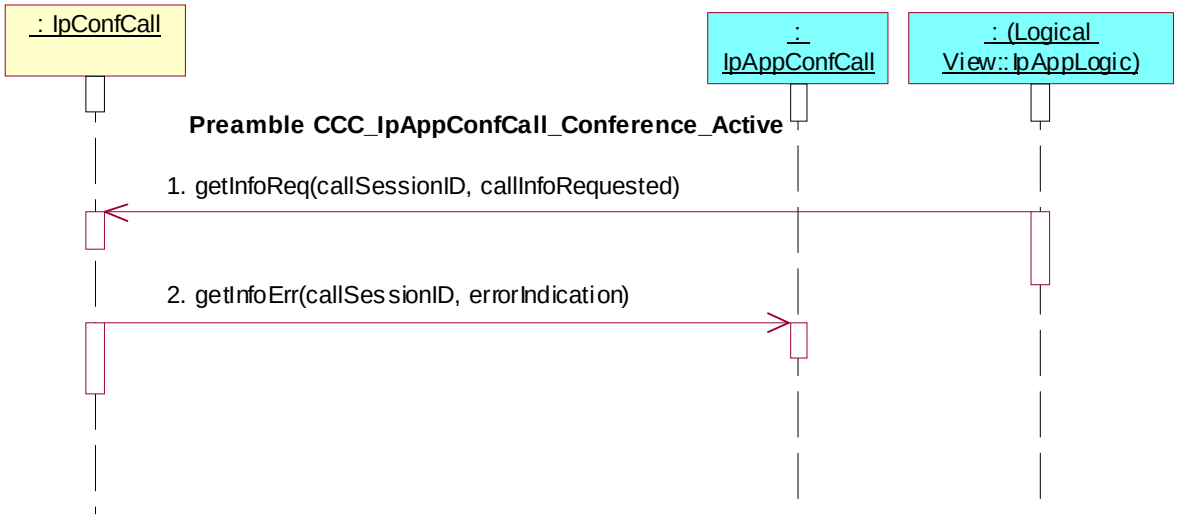
Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **CCC_IpAppConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) IpConfCall interface.
   Parameters: callSessionID, callInfoRequested

2. Method call **getInfoErr()**
   Parameters: callSessionID, errorIndication
   Check: no exception is returned



Test CCC_IpAppConfCall_13

Summary: set charge plan for the call

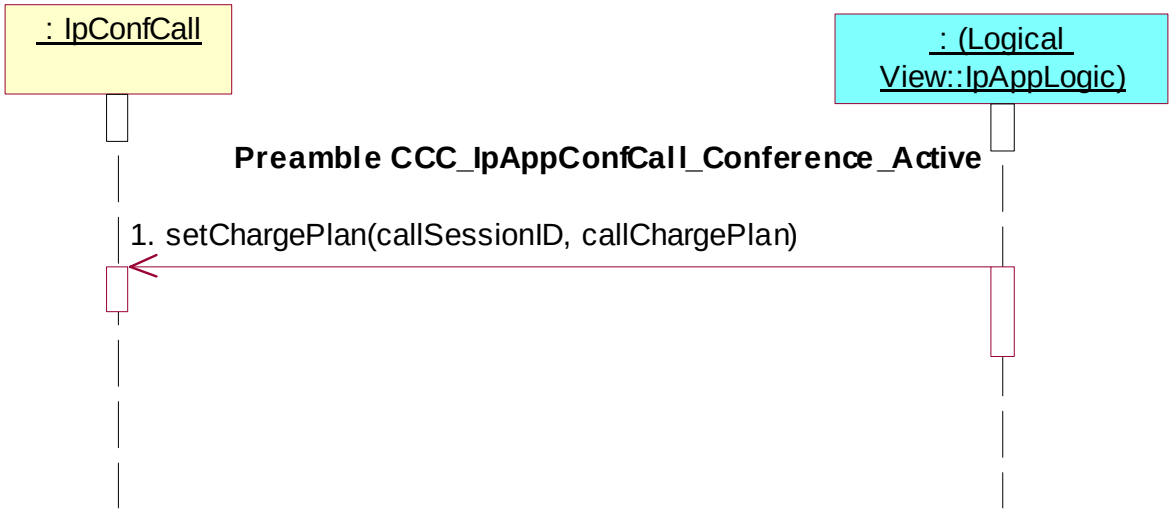
Reference: ES 202 915-4-3 [3], clause 7.2.2 and ES 202 915-4-5 [5], clause 6.3

Precondition: IUT capable of invoking **setChargePlan()**

Preamble: **CCC_IpAppConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **setChargePlan()** method on the tester's (SCF's) IpConfCall interface.
   Parameters: callSessionID, callChargePlan



Test CCC_IpAppConfCall_14

Summary: allow advice of charge information

Reference: ES 202 915-4-3 [3], clause 7.2.2 and ES 202 915-4-5 [5], clause 6.3

Precondition: IUT capable of invoking **setAdviceOfCharge()**

Preamble: **CCC_IpAppConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **setAdviceOfCharge()** method on the tester's (SCF's) IpConfCall interface.
   Parameters: callSessionID, aOCInfo, tariffSwitch

Test CCC_IpAppConfCall_15

Summary: get conference address

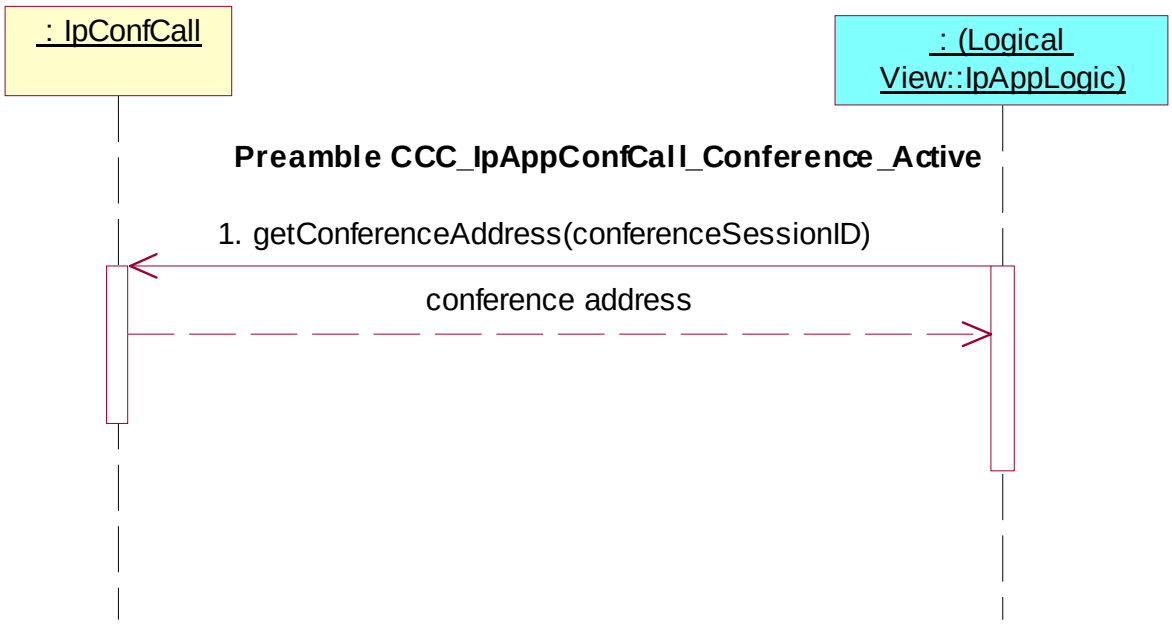
Reference: ES 202 915-4-3 [3], clause 7.2.2 and ES 202 915-4-5 [5], clause 6.3

Precondition: IUT capable of invoking **getConferenceAddress()**

Preamble: **CCC_IpAppConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **getConferenceAddress()** method on the tester's (SCF's) IpConfCall interface.
   Parameters: conferenceSessionID



7.2.4.3 IpAppSubConfCall

Applications need not be capable of performing each of the sequences below, even if they support the methods indicated below.

Reference: ES 202 915-4-5 [5], clauses 6.5 and 6.6

Precondition: IUT capable of invoking **createSubConference()** and **reserveResources()** or **createConference()**

Preamble CCC_IpAppSubConfCall_Conference_Active

Reference: ES 202 915-4-3 [3], clauses 6.3 and 6.5 and ES 202 915-4-5 [5], clauses 6.1, 6.2 and 6.3

Pre-preamble: Registration of the IUT (application) and the tester (Call Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpConfCallControlManager interface through selecting that service and signing the required service agreement.

The application is permitted to provide its IpAppConferenceCallControlManager interface reference in a setCallback() method which it calls prior to invoking further methods.

Preamble Sequence:

1. Triggered Action: cause IUT to call **reserveResources()** method on the tester's (SCF's) IpConfCallControlManager interface.
   Parameters: appInterface, startTime, numberOfParticipants, duration, conferencePolicy

2. Method call **conferenceCreated()**
   Parameters: conferenceCall
   Check: valid value of IpAppConfCallRef is returned

3. Triggered Action: cause IUT to call **createSubConference()** method on the tester's (SCF's) IpConfCall interface.
   Parameters: conferenceSessionID, appSubConference, conferencePolicy

and either

4. Triggered Action: cause IUT to call **createCallLeg()** method on the tester's (SCF's) IpSubConfCall interface.
   Parameters: callSessionID, appCallLeg

5. Triggered Action: cause IUT to call **routeReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callSessionID, targetAddress, originatingAddress, appInfo, connectionProperties

or

4. Triggered Action: cause IUT to call **createAndRouteCallLeg()** method on the tester's (SCF's) IpSubConfCall interface.
   Parameters: callSessionID, eventsRequested, targetAddress, originatingAddress, appInfo, appCallLegInterface

or

1. Triggered Action: cause IUT to call **createConference()** method on the tester's (SCF's) IpConfCallControlManager interface.
   Parameters: appConferenceCall, numberOfSubConferences, conferencePolicy, numberOfParticipants, duration

2. Triggered Action: cause IUT to call **createSubConference()** method on the tester's (SCF's) IpConfCall interface.
   Parameters: conferenceSessionID, appSubConference, conferencePolicy

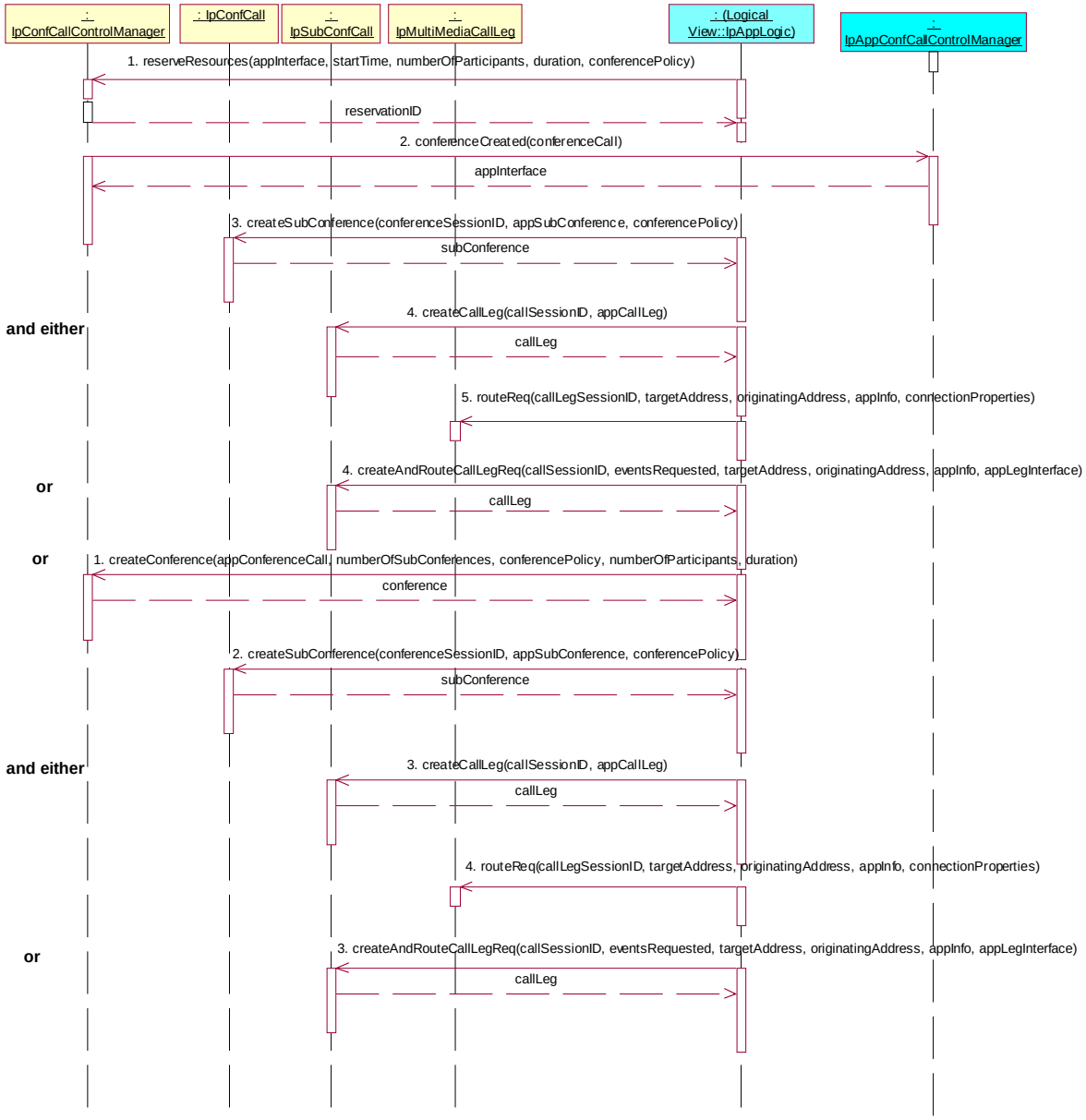
and either

3. Triggered Action: cause IUT to call **createCallLeg()** method on the tester's (SCF's) IpSubConfCall interface.
   Parameters: callSessionID, appCallLeg

4. Triggered Action: cause IUT to call **routeReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callSessionID, targetAddress, originatingAddress, appInfo, connectionProperties

or

3. Triggered Action: cause IUT to call **createAndRouteCallLeg()** method on the tester's (SCF's) IpSubConfCall interface.
   Parameters: callSessionID, eventsRequested, targetAddress, originatingAddress, appInfo, appCallLegInterface


: IpConfCallControlManager
: IpConfCall
: IpSubConfCall
: IpMultiMediaCallLeg
: (Logical View::IpAppLogic)
: IpAppConfCallControlManager
1. reserveResources(appInterface, startTime, numberOfParticipants, duration, conferencePolicy)
reservationID
2. conferenceCreated(conferenceCall)
appInterface
3. createSubConference(conferenceSessionID, appSubConference, conferencePolicy)
subConference
and either
4. createCallLeg(callSessionID, appCallLeg)
callLeg
5. routeReq(callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties)
or
4. createAndRouteCallLegReq(callSessionID, eventsRequested, targetAddress, originatingAddress, appInfo, appLegInterface)
callLeg
or
1. createConference(appConferenceCall, numberOfSubConferences, conferencePolicy, numberOfParticipants, duration)
conference
2. createSubConference(conferenceSessionID, appSubConference, conferencePolicy)
subConference
and either
3. createCallLeg(callSessionID, appCallLeg)
callLeg
4. routeReq(callLegSessionID, targetAddress, originatingAddress, appInfo, connectionProperties)
or
3. createAndRouteCallLegReq(callSessionID, eventsRequested, targetAddress, originatingAddress, appInfo, appLegInterface)
callLeg

Test CCC_IpAppSubConfCall_01

Summary: split subconference

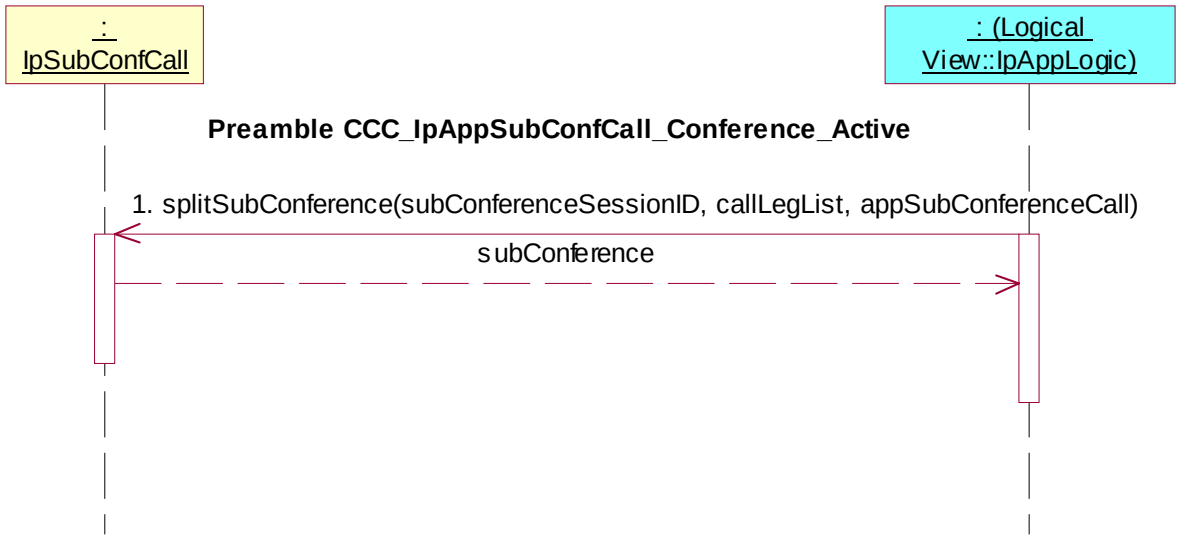
Reference: ES 202 915-4-5 [5], clause 6.5

Precondition: IUT capable of invoking **splitSubConference()**

Preamble CCC_IpAppSubConfCall_Conference_Active

Test Sequence:

1. Triggered Action: cause IUT to call **splitSubConference()** method on the tester's (SCF's) IpSubConfCall interface.
   Parameters: subConferenceSessionID, CallLegList, appSubConferenceCall



Test CCC_IpAppSubConfCall_02

Summary: merge subconferences

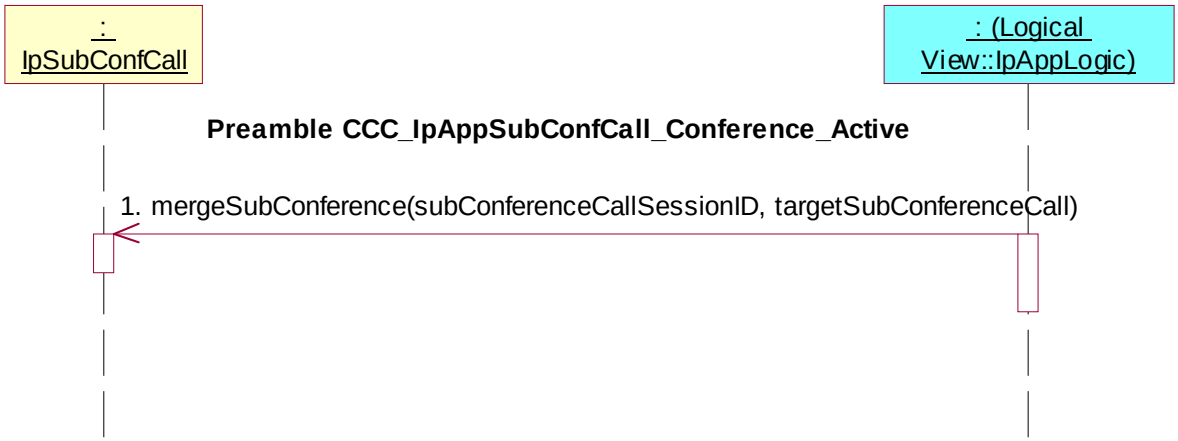
Reference: ES 202 915-4-5 [5], clause 6.5

Precondition: IUT capable of invoking **mergeSubConference()**

Preamble CCC_IpAppSubConfCall_Conference_Active

Test Sequence:

1. Triggered Action: cause IUT to call **mergeSubConference()** method on the tester's (SCF's) IpSubConfCall interface.
   Parameters: subConferenceCallSessionID, targetSubConferenceCall



Test CCC_IpAppSubConfCall_03

Summary: move call leg between subconferences

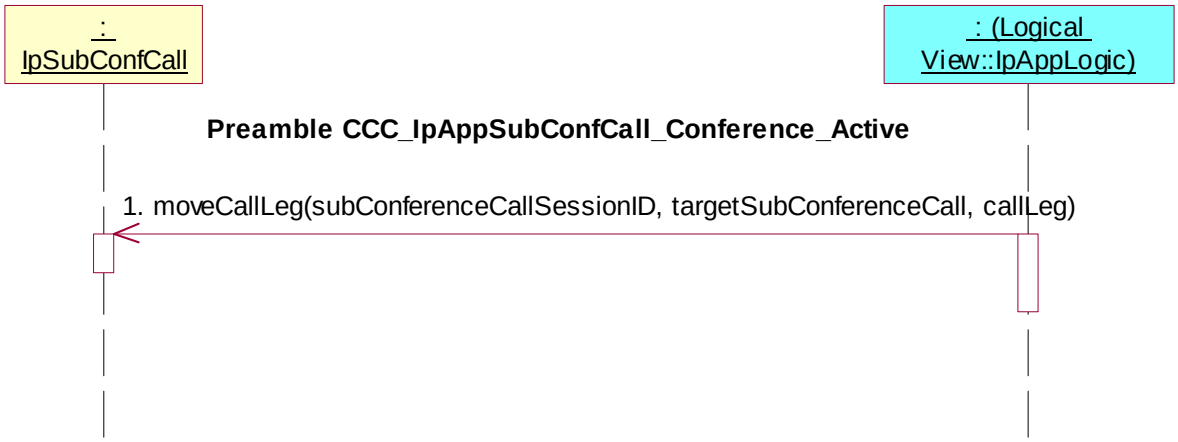
Reference: ES 202 915-4-5 [5], clause 6.5

Precondition: IUT capable of invoking **moveCallLeg()**

Preamble CCC_IpAppSubConfCall_Conference_Active

Test Sequence:

1. Triggered Action: cause IUT to call **moveCallLeg()** method on the tester's (SCF's) IpSubConfCall interface.
   Parameters: subConferenceCallSessionID, targetSubConferenceCall, callLeg



Test CCC_IpAppSubConfCall_04

Summary: appoint conferences policy

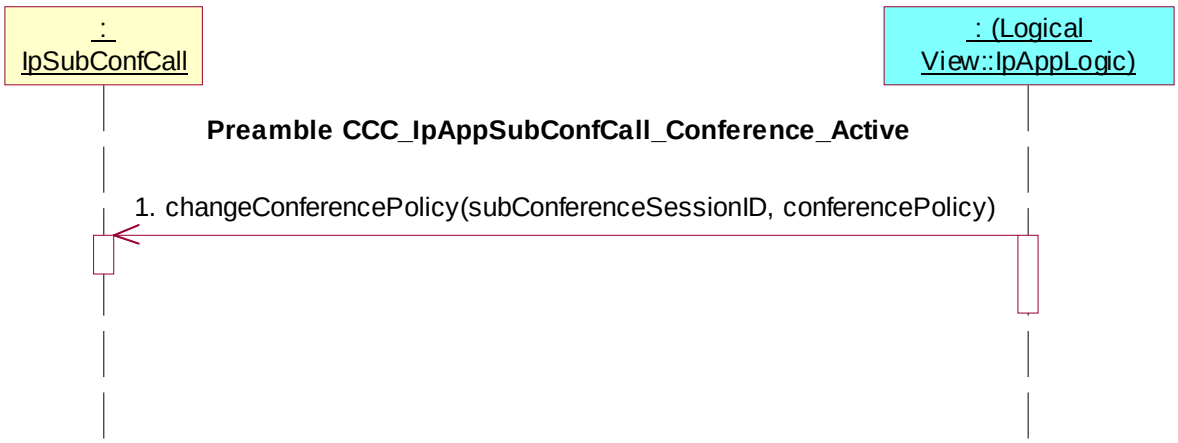
Reference: ES 202 915-4-5 [5], clause 6.5

Precondition: IUT capable of invoking **changeConferencePolicy()**

Preamble CCC_IpAppSubConfCall_Conference_Active

Test Sequence:

1. Triggered Action: cause IUT to call **changeConferencePolicy()** method on the tester's (SCF's) IpSubConfCall interface.
   Parameters: subConferenceSessionID,conferencePolicy



Test CCC_IpAppSubConfCall_05

Summary: appoint speaker on request

Reference: ES 202 915-4-5 [5], clauses 6.5 and 6.6

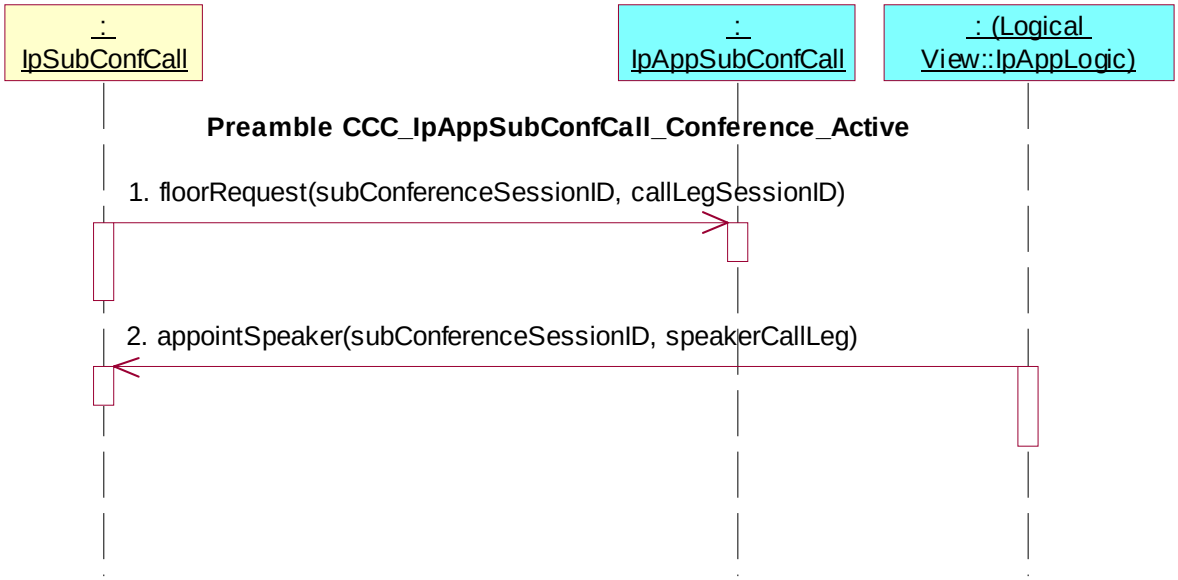
Precondition: IUT capable of invoking **appointSpeaker()**

Preamble CCC_IpAppSubConfCall_Conference_Active

Test Sequence:

1. Method call **floorRequest()**
   Parameters: subConferenceSessionID, CallLegSessionID
   Check: no exception is returned

2. Triggered Action: cause IUT to call **appointSpeaker()** method on the tester's (SCF's) IpSubConfCall interface.
   Parameters: subConferenceSessionID, speakerCallLeg



Test CCC_IpAppSubConfCall_06

Summary: select chair on request

Reference: ES 202 915-4-5 [5], clauses 6.5 and 6.6

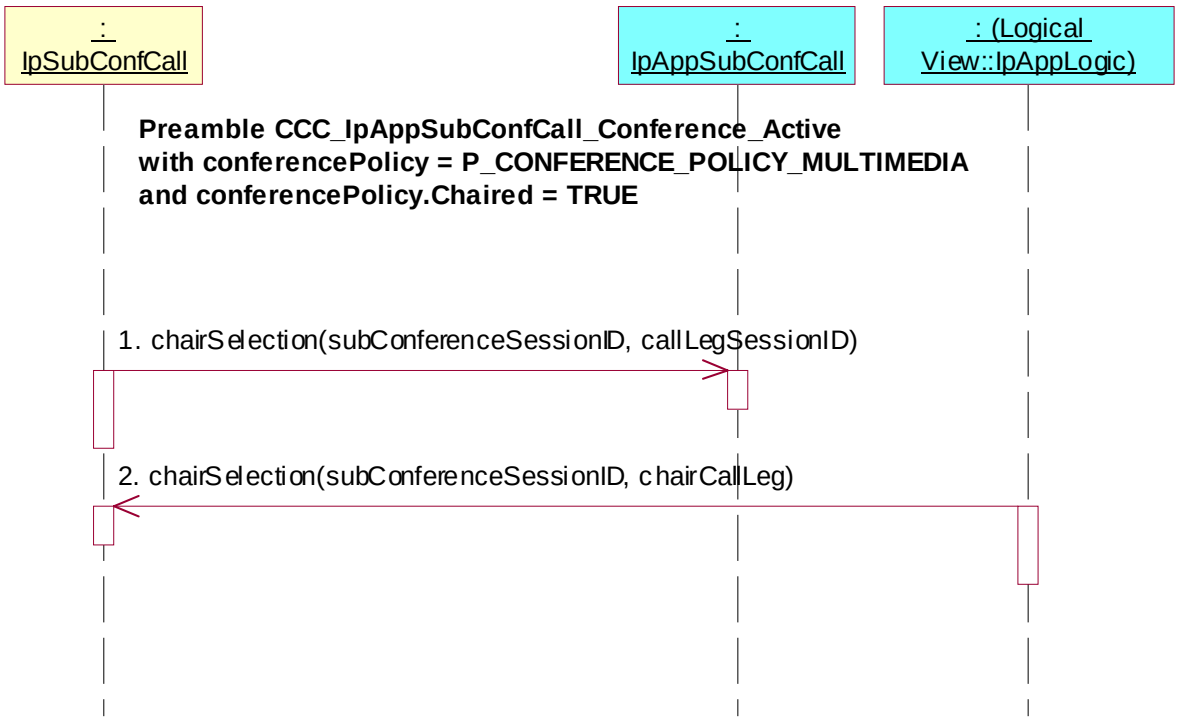
Precondition: IUT capable of invoking **chairSelection()**

Preamble CCC_IpAppSubConfCall_Conference_Active with conferencePolicy = P_CONFERENCE_POLICY_MULTIMEDIA and conferencePolicy.Chaired = TRUE

Test Sequence:

1. Method call **chairSelection()**
   Parameters: subConferenceSessionID, CallLegSessionID
   Check: no exception is returned

2. Triggered Action: cause IUT to call **chairSelection()** method on the tester's (SCF's) IpSubConfCall interface.
   Parameters: subConferenceSessionID, chairCallLeg



Test CCC_IpAppSubConfCall_07

Summary: select chair on request and inspect video

Reference: ES 202 915-4-5 [5], clauses 6.5 and 6.6

Precondition: IUT capable of invoking **chairSelection()**, **inspectVideo()** and **InspectVideoCancel()**

Preamble CCC_IpAppSubConfCall_Conference_Active with conferencePolicy = P_CONFERENCE_POLICY_MULTIMEDIA and conferencePolicy.Chaired = TRUE and conferencePolicy.MediaAllowed = Video

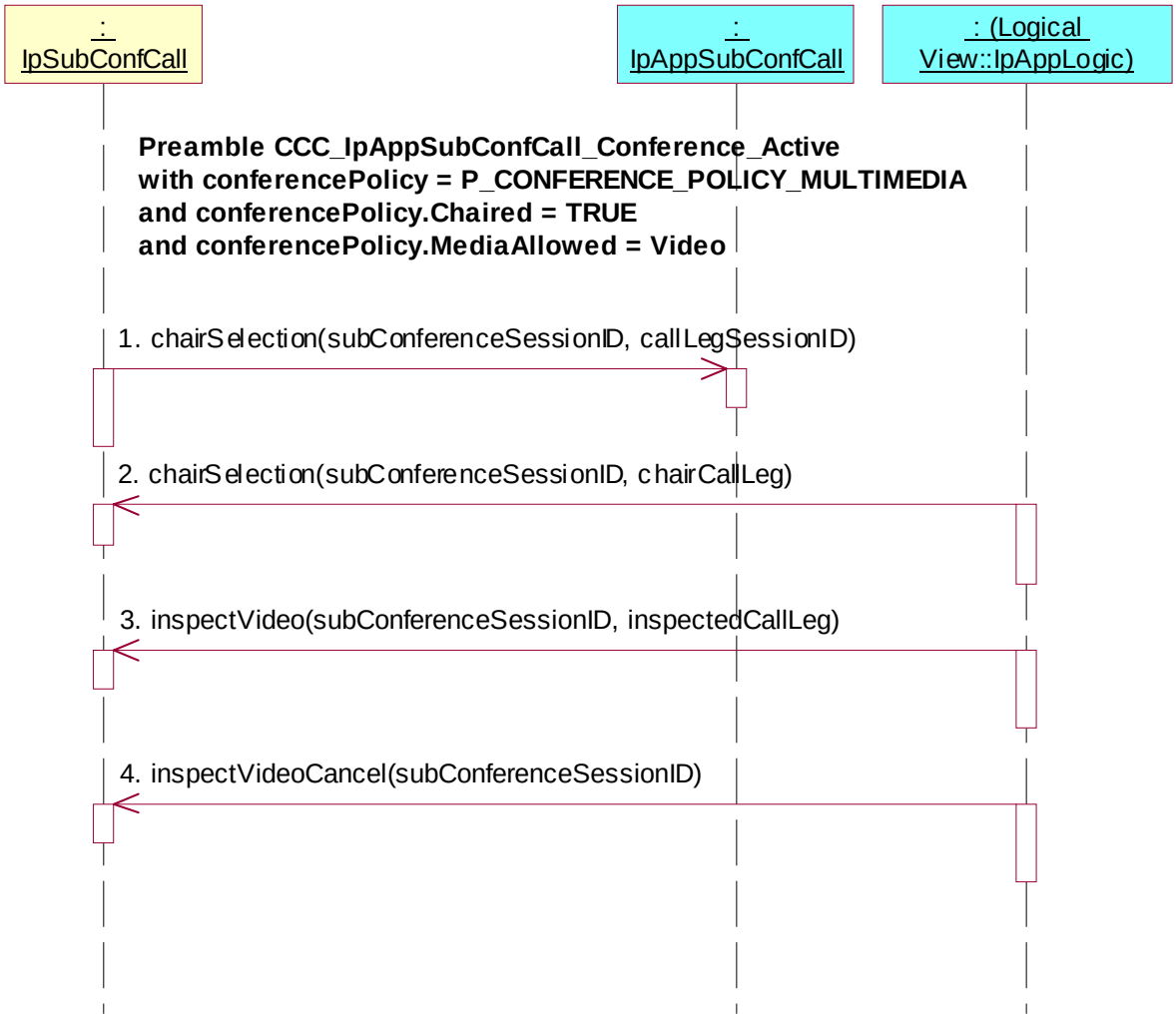
Test Sequence:

1. Method call **chairSelection()**
   Parameters: subConferenceSessionID, CallLegSessionID
   Check: no exception is returned

2. Triggered Action: cause IUT to call **chairSelection()** method on the tester's (SCF's) IpSubConfCall interface.
   Parameters: subConferenceSessionID, chairCallLeg

3. Triggered Action: cause IUT to call **inspectVideo()** method on the tester's (SCF's) IpSubConfCall interface.
   Parameters: subConferenceSessionID, inspectedCallLeg

4. Triggered Action: cause IUT to call **InspectVideoCancel()** method on the tester's (SCF's) IpSubConfCall interface.
   Parameters: subConferenceSessionID



Test CCC_IpAppSubConfCall_08

Summary: release call

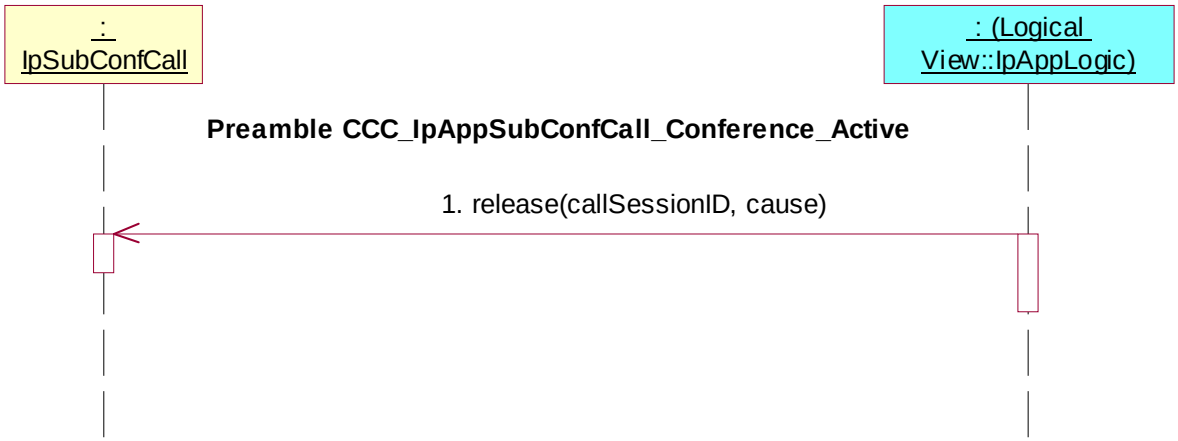
Reference: ES 202 915-4-3 [3], clause 7.2.2 and ES 202 915-4-5 [5], clause 6.5

Precondition: IUT capable of invoking **release()**

Preamble: **CCC_IpAppSubConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) IpConfCall interface.
   Parameters: callSessionID, cause



Test CCC_IpAppSubConfCall_09

Summary: deassign call

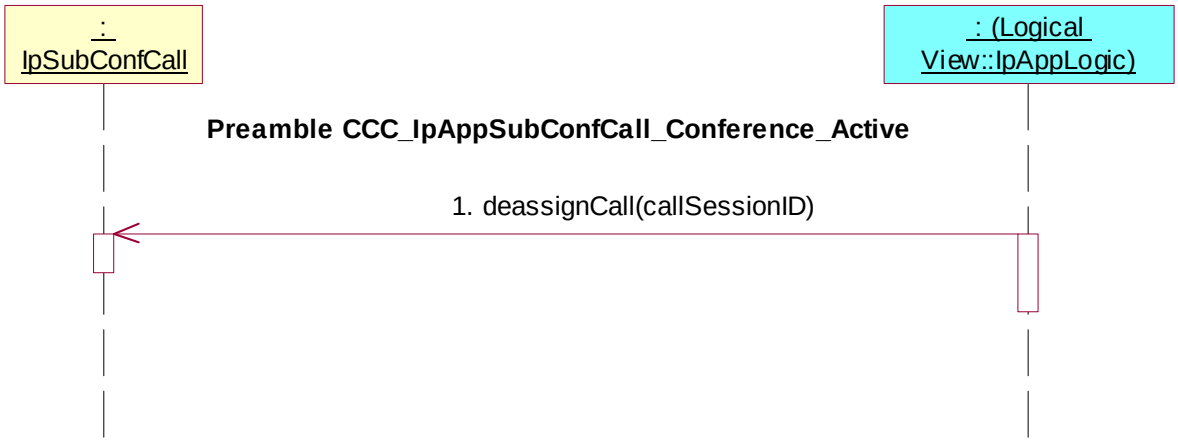
Reference: ES 202 915-4-3 [3], clause 7.2.2 and ES 202 915-4-5 [5], clause 6.5

Precondition: IUT capable of invoking **deassignCall()**

Preamble: **CCC_IpAppSubConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **deassignCall()** method on the tester's (SCF's) IpConfCall interface.
   Parameters: callSessionID



Test CCC_IpAppConfCall_10

Summary: create call leg

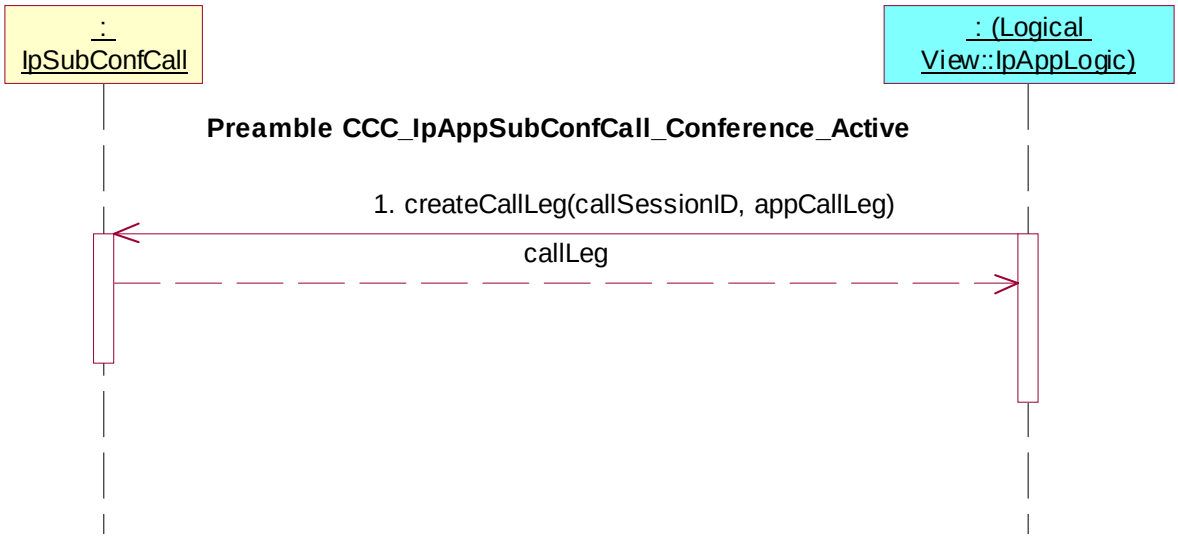
Reference: ES 202 915-4-3 [3], clause 7.2.2 and ES 202 915-4-5 [5], clause 6.5

Precondition: IUT capable of invoking **createCallLeg()**

Preamble: **CCC_IpAppSubConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **createCallLeg()** method on the tester's (SCF's) IpConfCall interface.
   Parameters: callSessionID, appCallLeg



Test CCC_IpAppSubConfCall_11

Summary: create and route call

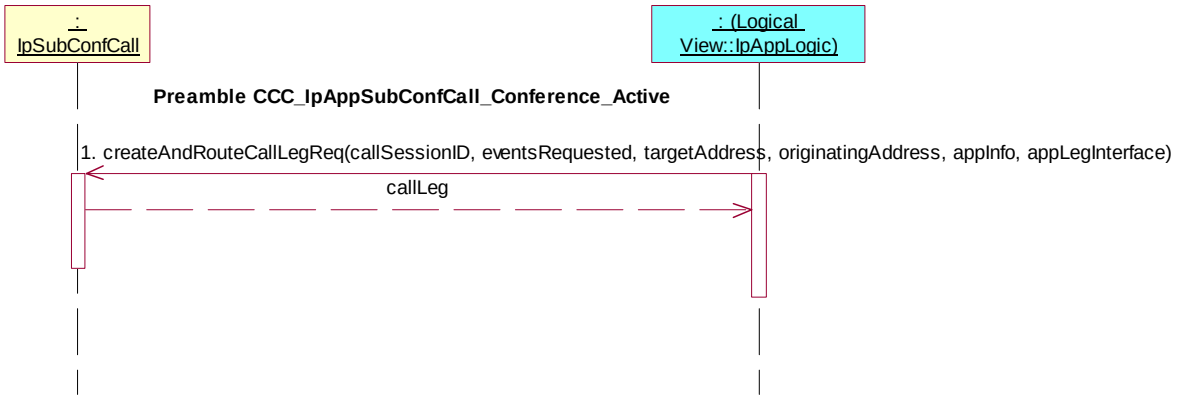
Reference: ES 202 915-4-3 [3], clause 7.2.2 and ES 202 915-4-5 [5], clause 6.5

Precondition: IUT capable of invoking **createAndRouteCallLeg()**

Preamble: **CCC_IpAppSubConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **createAndRouteCallLeg()** method on the tester's (SCF's) IpSubConfCall interface.
   Parameters: callSessionID, eventsRequested, targetAddress, originatingAddress, appInfo, appCallLegInterface



Test CCC_IpAppSubConfCall_12

Summary: request call leg information

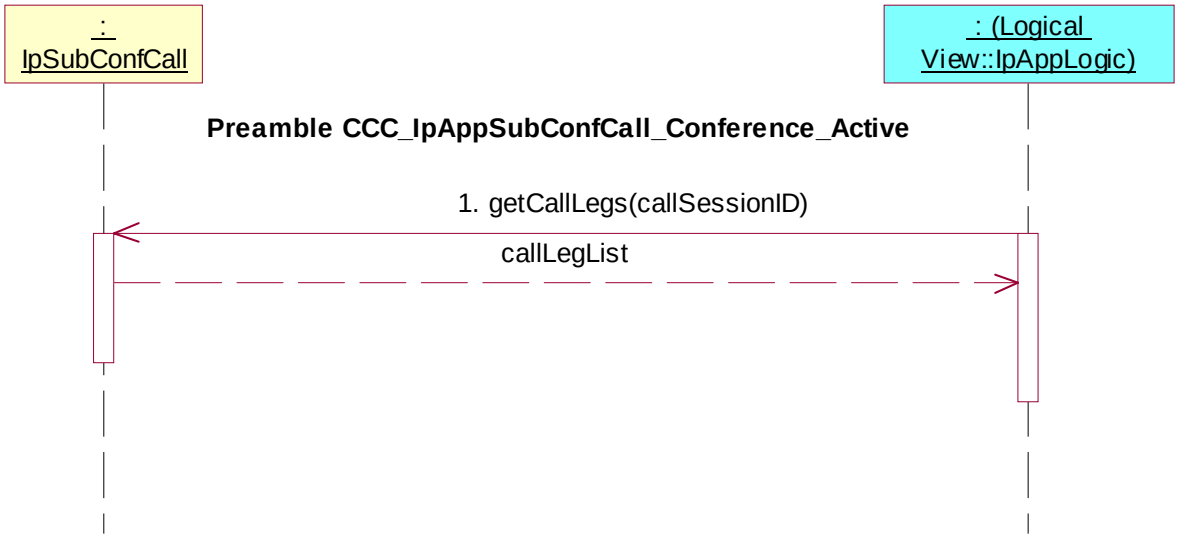
Reference: ES 202 915-4-3 [3], clause 7.2.2 and ES 202 915-4-5 [5], clause 6.5

Precondition: IUT capable of invoking **getCallLegs()**

Preamble: **CCC_IpAppSubConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **getCallLegs()** method on the tester's (SCF's) IpConfCall interface.
   Parameters: callSessionID



Test CCC_IpAppSubConfCall_13

Summary: request call information

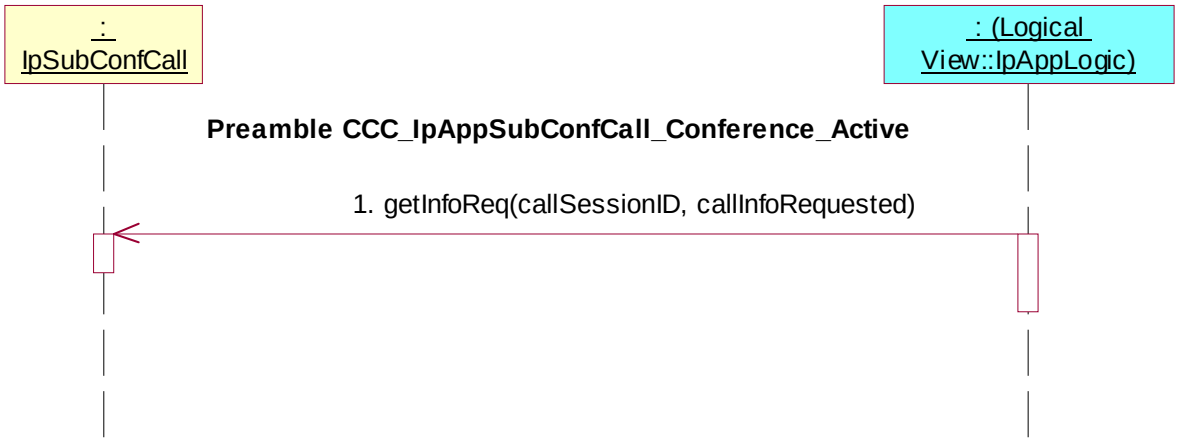
Reference: ES 202 915-4-3 [3], clause 7.2.2 and ES 202 915-4-5 [5], clause 6.5

Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **CCC_IpAppSubConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) IpConfCall interface.
   Parameters: callSessionID, callInfoRequested



Test CCC_IpAppSubConfCall_14

Summary: request call information, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.2.2 and ES 202 915-4-5 [5], clause 6.5

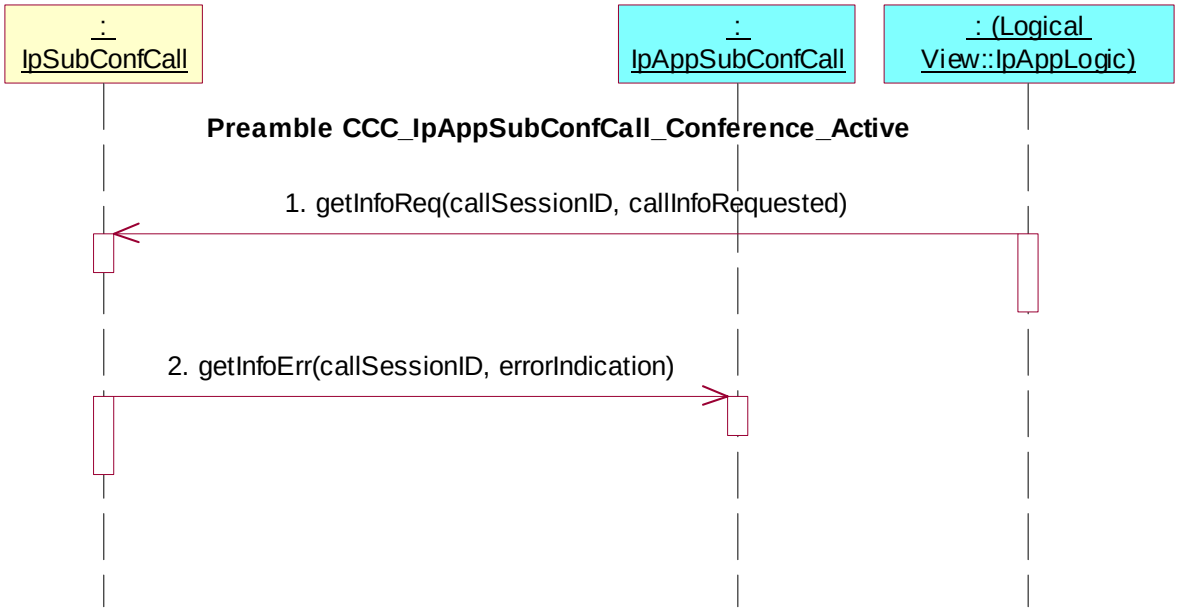
Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **CCC_IpAppSubConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) IpConfCall interface.
   Parameters: callSessionID, callInfoRequested

2. Method call **getInfoErr()**
   Parameters: callSessionID, errorIndication
   Check: no exception is returned



Test CCC_IpAppSubConfCall_15

Summary: allow advice of charge information

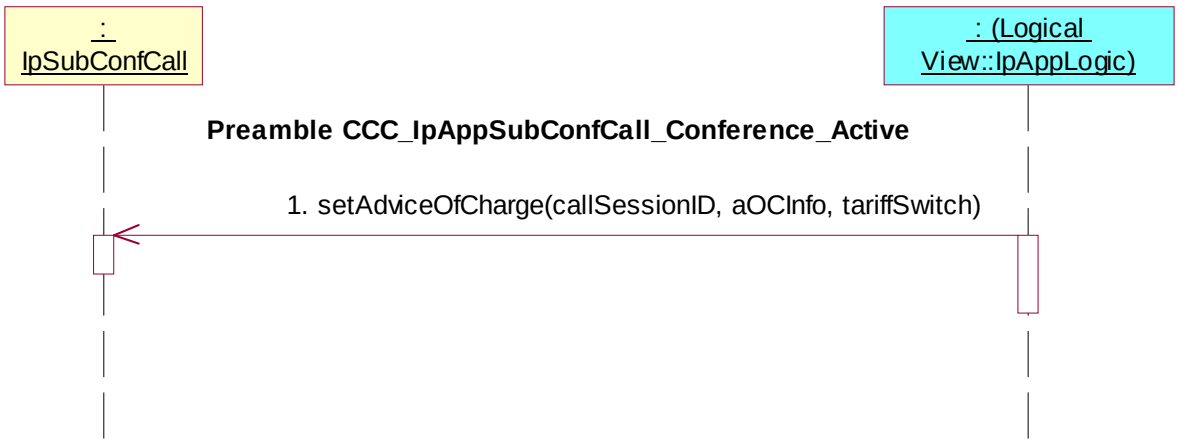
Reference: ES 202 915-4-3 [3], clause 7.2.2 and ES 202 915-4-5 [5], clause 6.5

Precondition: IUT capable of invoking **setAdviceOfCharge()**

Preamble: **CCC_IpAppSubConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **setAdviceOfCharge()** method on the tester's (SCF's) IpConfCall interface.
   Parameters: callSessionID, aOCInfo, tariffSwitch



7.2.4.4 IpAppMultiMediaCallLeg

Applications need not be capable of performing each of the sequences below, even if they support the methods indicated below.

Reference: ES 202 915-4-3 [3], clause 7.3 and ES 202 915-4-4 [4], clauses 6.5 and 6.6

Precondition: IUT capable of invoking **createSubConference()** and **reserveResources()** or **creat**eConference()

Test CCC_IpAppMultiMediaCallLeg_01

Summary: attach media, successful

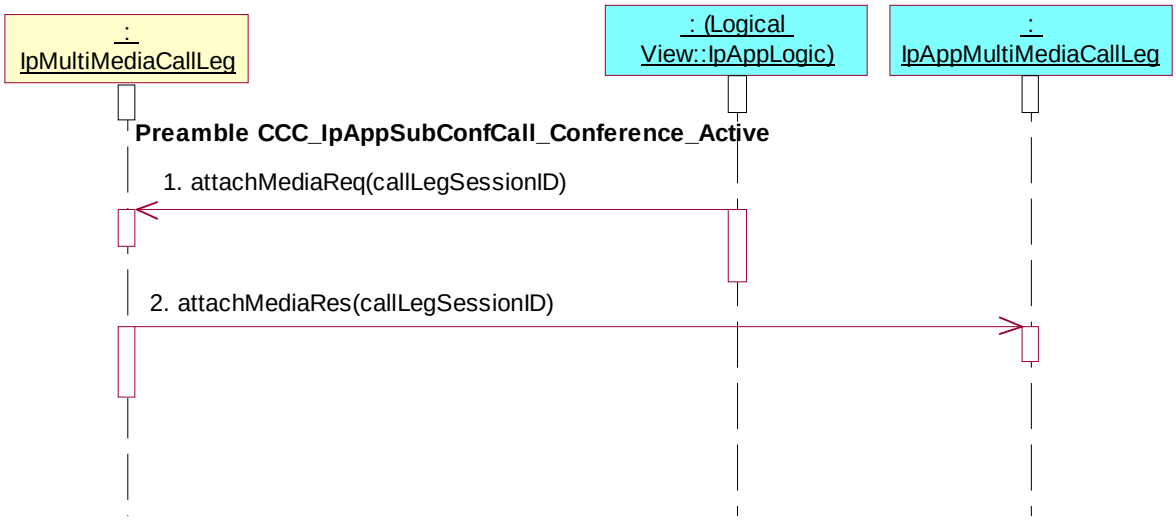
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **attachMediaReq()**

Preamble: **CCC_IpAppSubConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **attachMediaReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID
2. Method call **attachMediaRes()**
   Parameters: callLegSessionID
   Check: no exception is returned



Test CCC_IpAppMultiMediaCallLeg_02

Summary: attach media, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.2

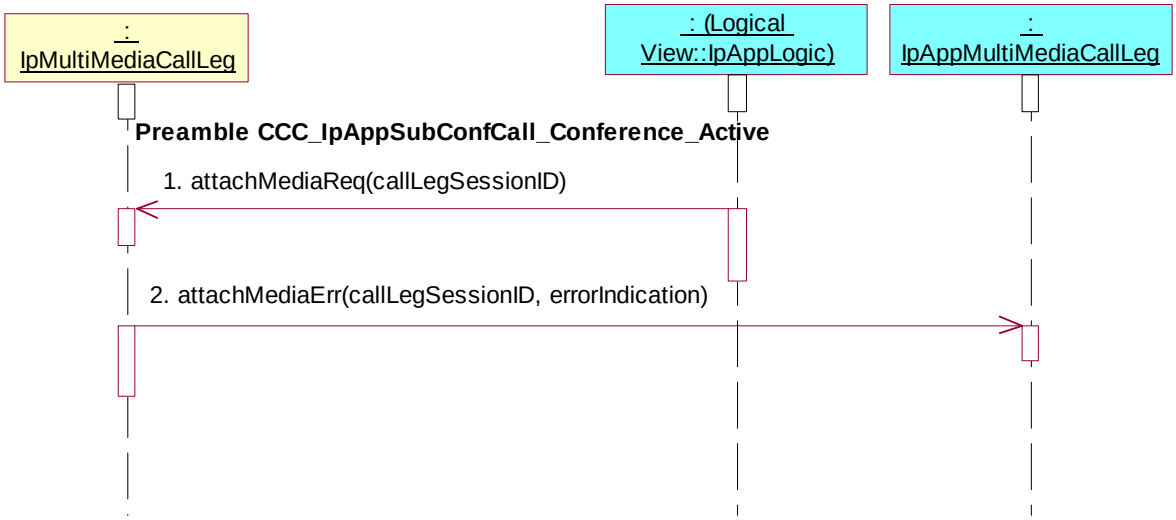
Precondition: IUT capable of invoking **attachMediaReq()**

Preamble: **CCC_IpAppSubConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **attachMediaReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID

2. Method call **attachMediaErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test CCC_IpAppMultiMediaCallLeg_03

Summary: detach media, successful

Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **detachMediaReq()**

Preamble: **CCC_IpAppSubConfCall_Conference_Active**

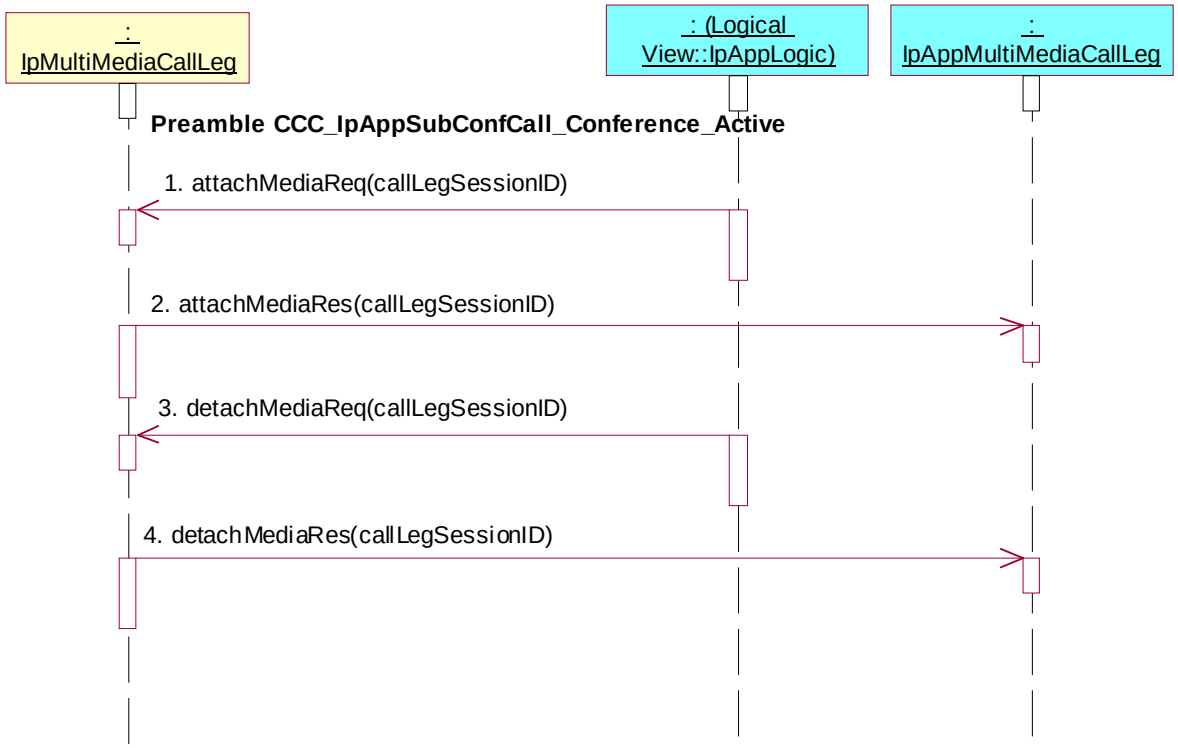
Test Sequence:

1. Triggered Action: cause IUT to call **attachMediaReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID

2. Method call **attachMediaRes()**
   Parameters: callLegSessionID
   Check: no exception is returned

3. Triggered Action: cause IUT to call **detachMediaReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID

4. Method call **detachMediaRes()**
   Parameters: callLegSessionID
   Check: no exception is returned



Test CCC_IpAppMultiMediaCallLeg_04

Summary: detach media, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **detachMediaReq()**

Preamble: **CCC_IpAppSubConfCall_Conference_Active**

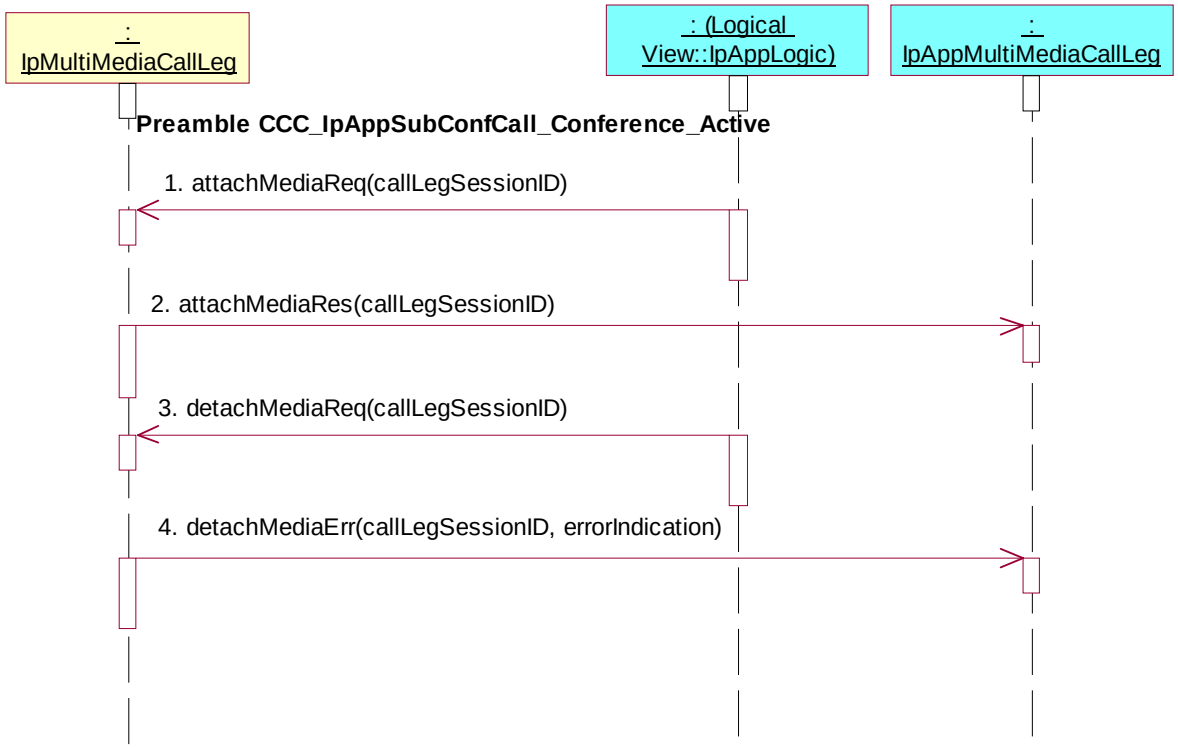
Test Sequence:

1. Triggered Action: cause IUT to call **attachMediaReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID

2. Method call **attachMediaRes()**
   Parameters: callLegSessionID
   Check: no exception is returned

3. Triggered Action: cause IUT to call **detachMediaReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID

4. Method call **detachMediaErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test CCC_IpAppMultiMediaCallLeg_05

Summary: request reference of call related to call leg

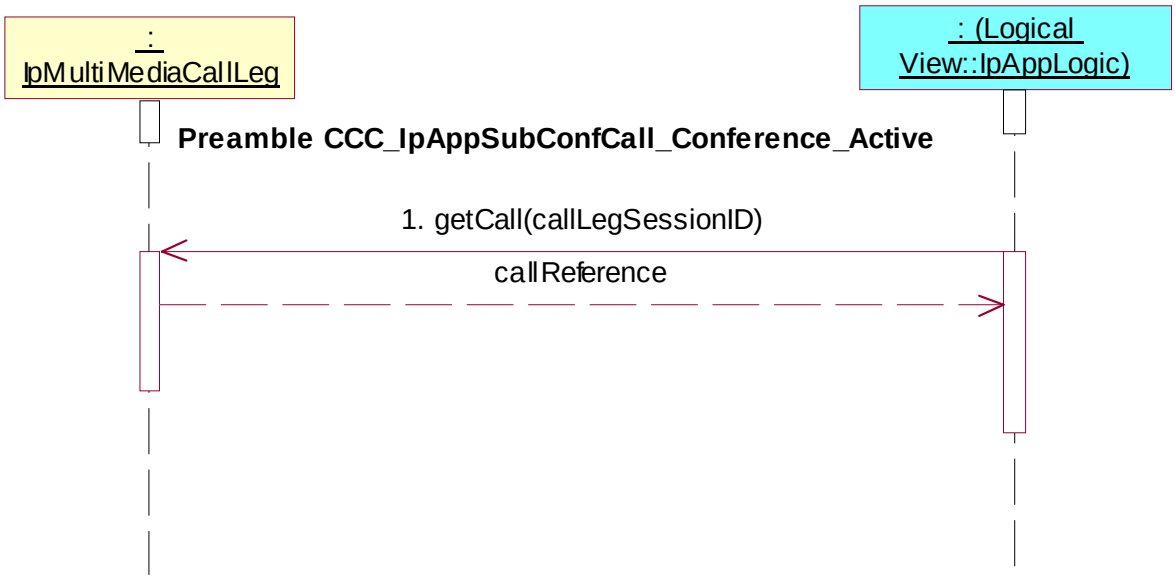
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **getCall()**

Preamble: **CCC_IpAppSubConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **getCall()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID



Test CCC_IpAppMultiMediaCallLeg_06

Summary: request reference of call related to call leg

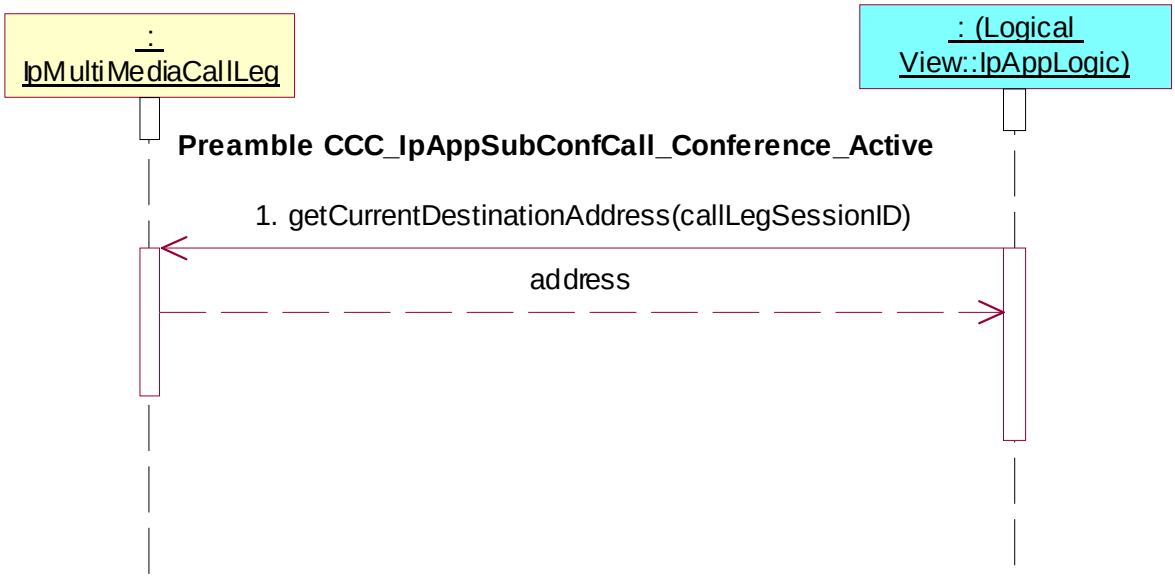
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **getCurrentDestinationAddress()**

Preamble: **CCC_IpAppSubConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **getCurrentDestinationAddress()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID



Test CCC_IpAppMultiMediaCallLeg_07

Summary: continue processing of call leg

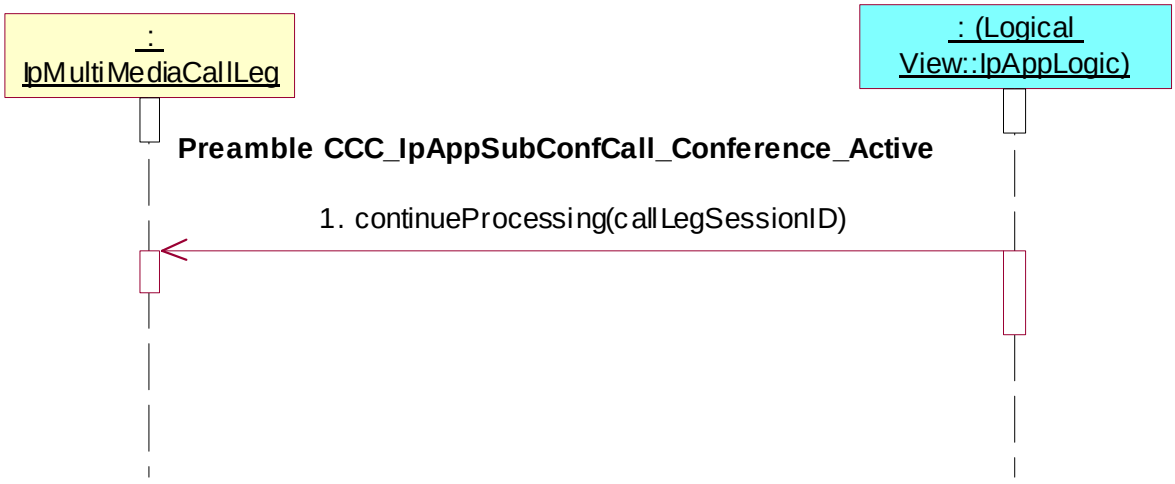
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **continueProcessing()**

Preamble: **CCC_IpAppSubConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **continueProcessing()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID



Test CCC_IpAppMultiMediaCallLeg_08

Summary: release call leg

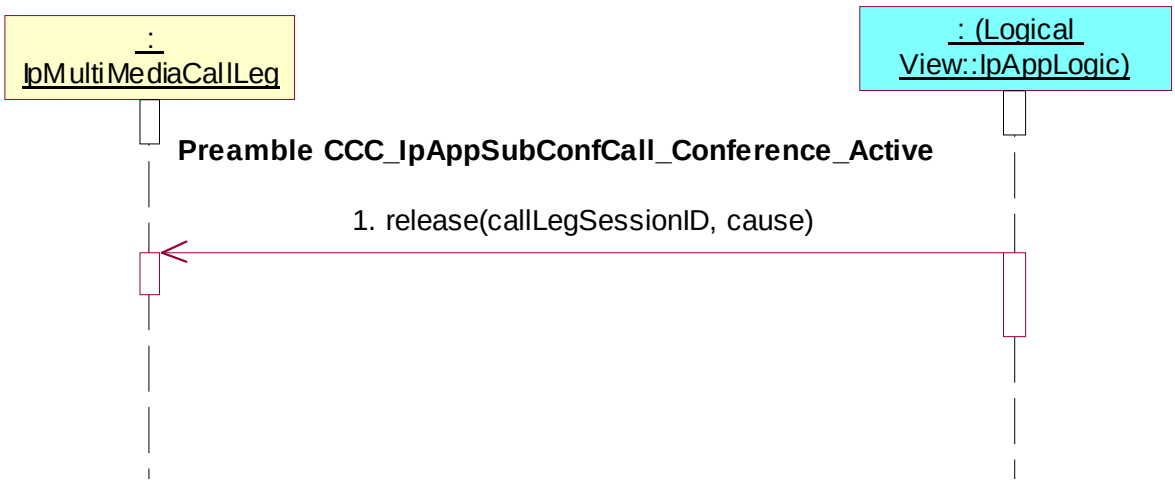
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **release()**

Preamble: **CCC_IpAppSubConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, cause



Test CCC_IpAppMultiMediaCallLeg_09

Summary: de-assign call leg

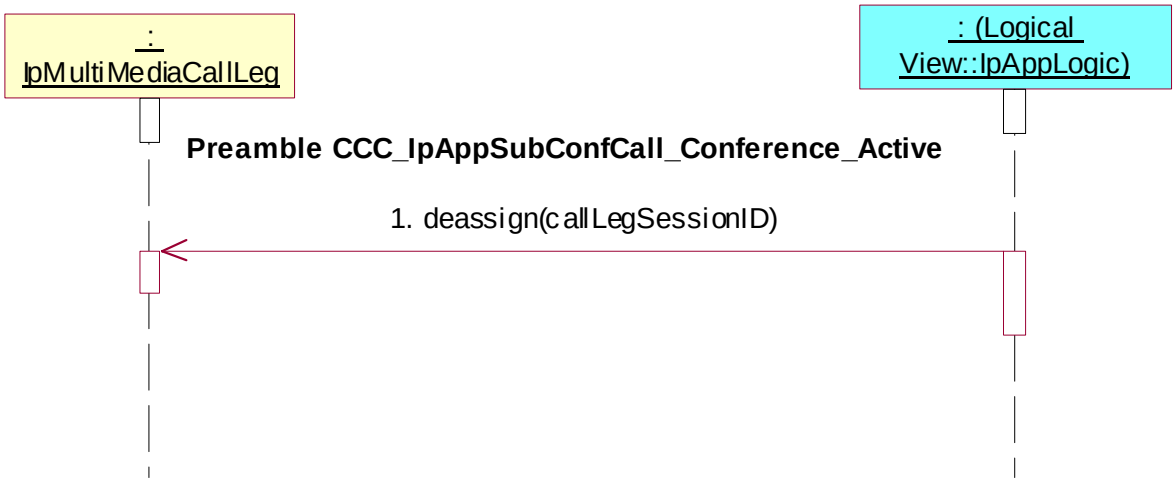
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **deassign()**

Preamble: **CCC_IpAppSubConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **deassign()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID



Test CCC_IpAppMultiMediaCallLeg_10

Summary: change or clear event criteria, successful

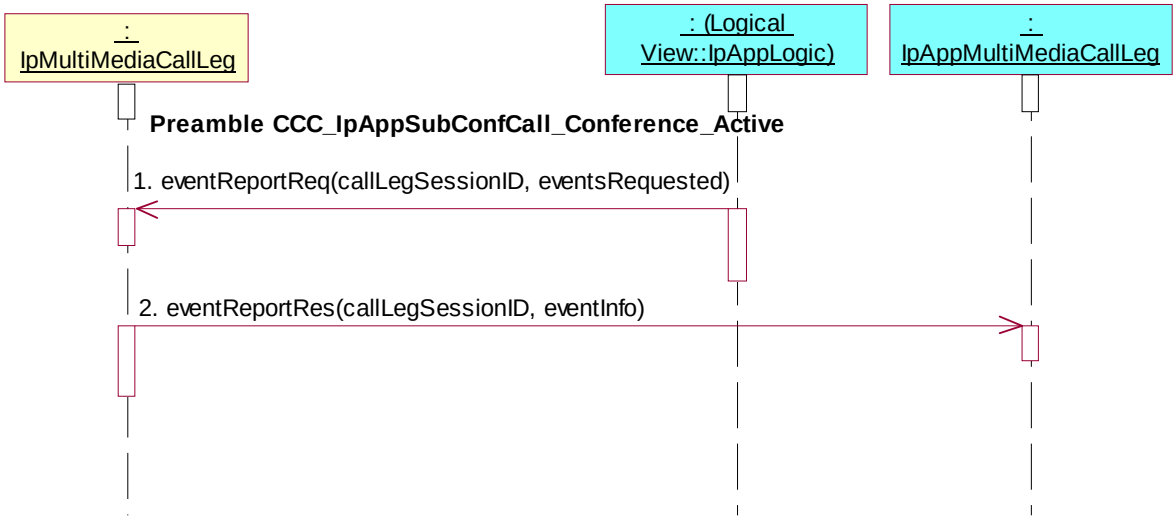
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **eventReportReq()**

Preamble: **CCC_IpAppSubConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, eventsRequested
2. Method call **eventReportRes()**
   Parameters: callLegSessionID, eventInfo
   Check: no exception is returned



Test CCC_IpAppMultiMediaCallLeg_11

Summary: change or clear event criteria, unsuccessful

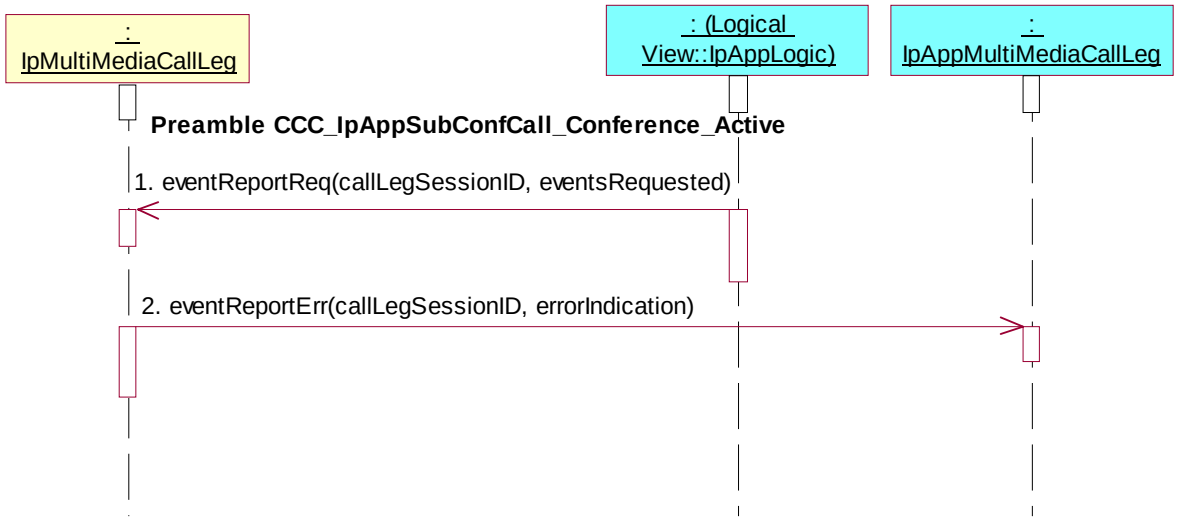
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **eventReportReq()**

Preamble: **CCC_IpAppSubConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **eventReportReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, eventsRequested
2. Method call **eventReportErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test CCC_IpAppMultiMediaCallLeg_12

Summary: get information about call leg, successful

Reference: ES 202 915-4-3 [3], clause 7.3.2

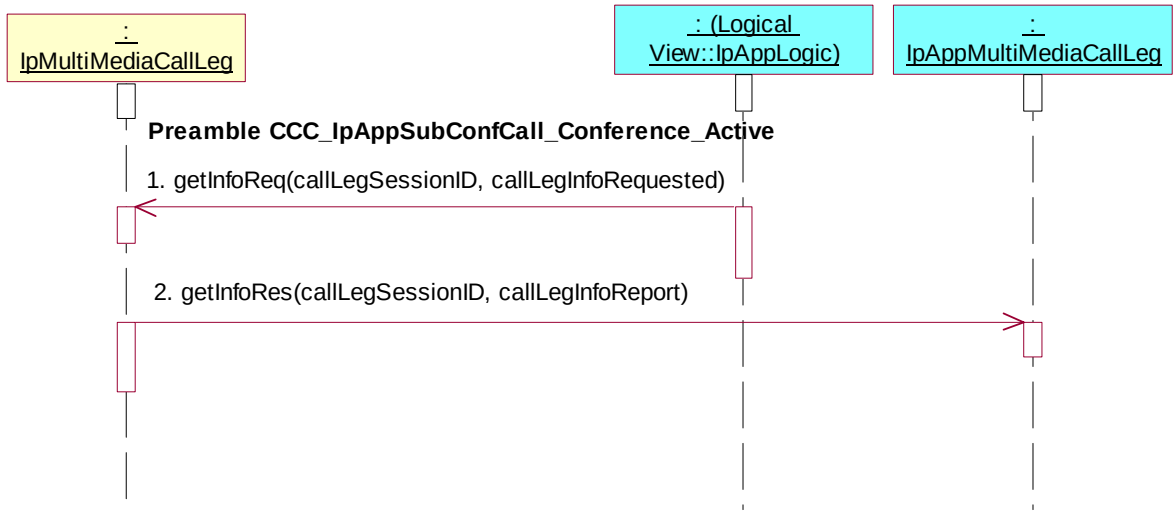
Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **CCC_IpAppSubConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, callLegInfoRequested

2. Method call **getInfoRes()**
   Parameters: callLegSessionID, callLegInfoReport
   Check: no exception is returned



Test CCC_IpAppMultiMediaCallLeg_13

Summary: get information about call leg, unsuccessful

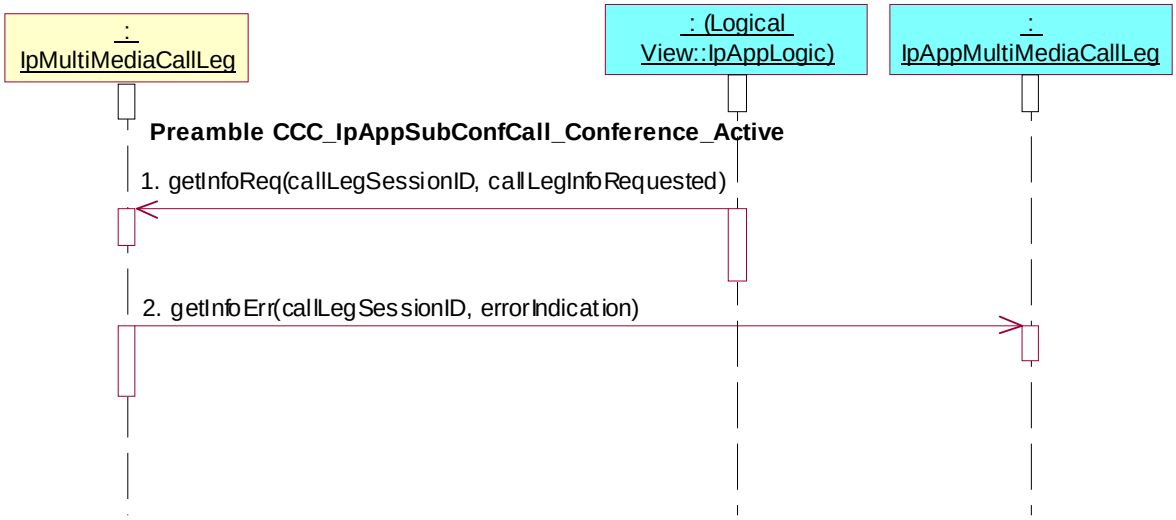
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **getInfoReq()**

Preamble: **CCC_IpAppSubConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **getInfoReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, callLegInfoRequested
2. Method call **getInfoErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test CCC_IpAppMultiMediaCallLeg_14

Summary: set charge plan for call leg

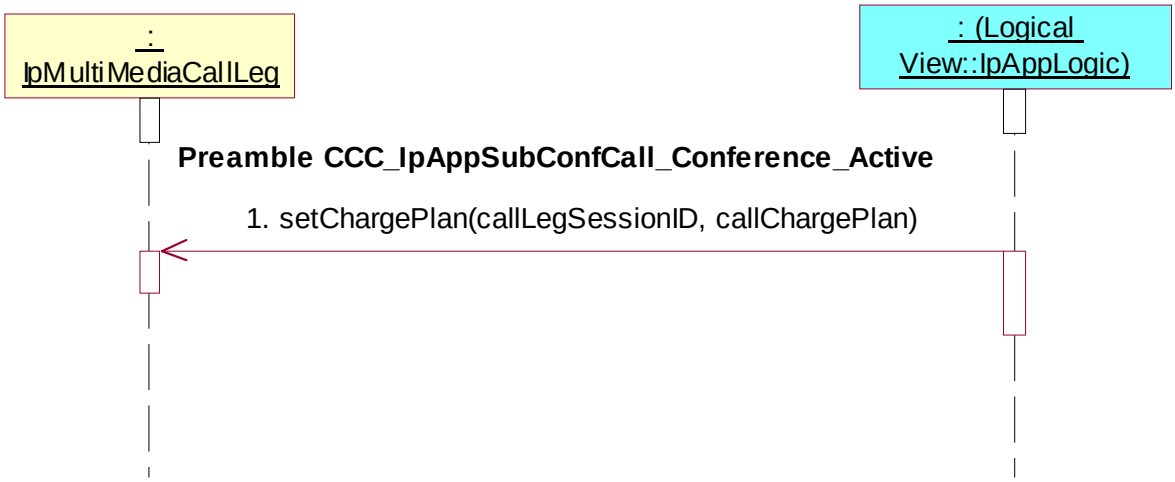
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **setChargePlan()**

Preamble: **CCC_IpAppSubConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **setChargePlan()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, callChargePlan



Test CCC_IpAppMultiMediaCallLeg_15

Summary: allow advice of charge information

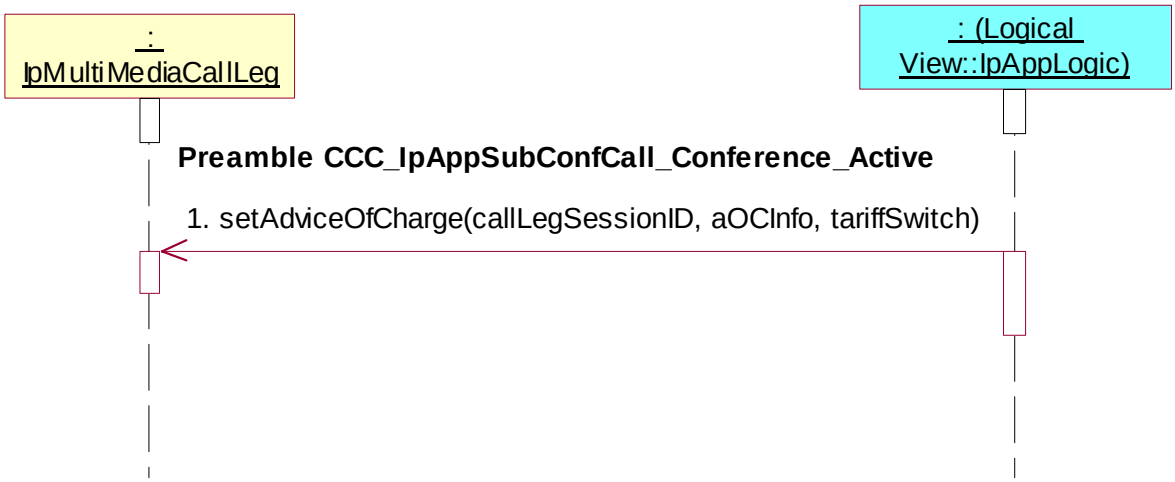
Reference: ES 202 915-4-3 [3], clause 7.3.2

Precondition: IUT capable of invoking **setAdviceOfCharge()**

Preamble: **CCC_IpAppSubConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **setAdviceOfCharge()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, aOCInfo, tariffSwitch



Test CCC_IpAppMultiMediaCallLeg_16

Summary: supervise call leg, successful

Reference: ES 202 915-4-3 [3], clause 7.3.2

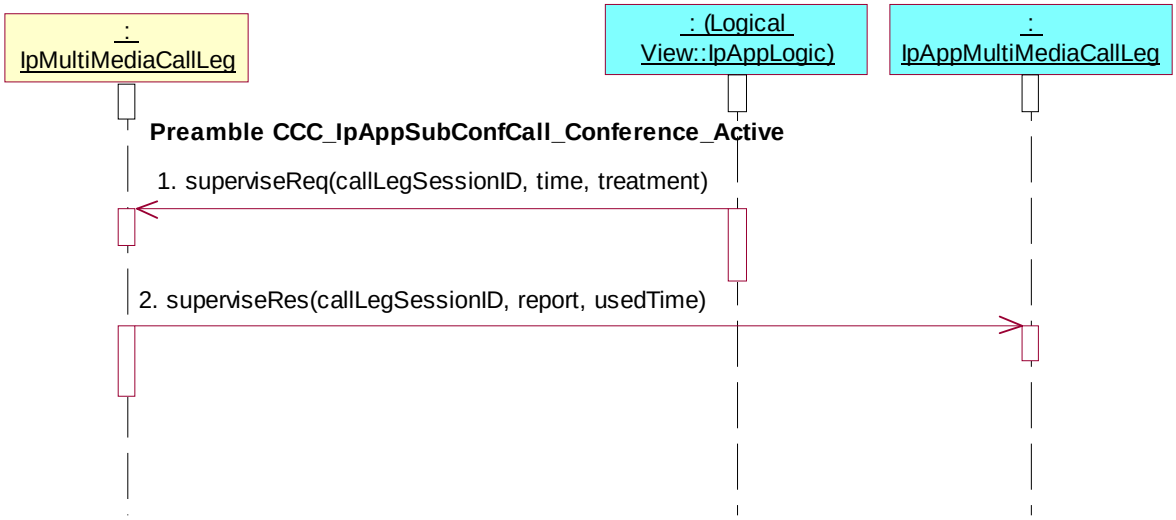
Precondition: IUT capable of invoking **superviseReq()**

Preamble: **CCC_IpAppSubConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, time, treatment

2. Method call **superviseRes()**
   Parameters: callLegSessionID, report, usedTime
   Check: no exception is returned



Test CCC_IpAppMultiMediaCallLeg_17

Summary: supervise call leg, unsuccessful

Reference: ES 202 915-4-3 [3], clause 7.3.2

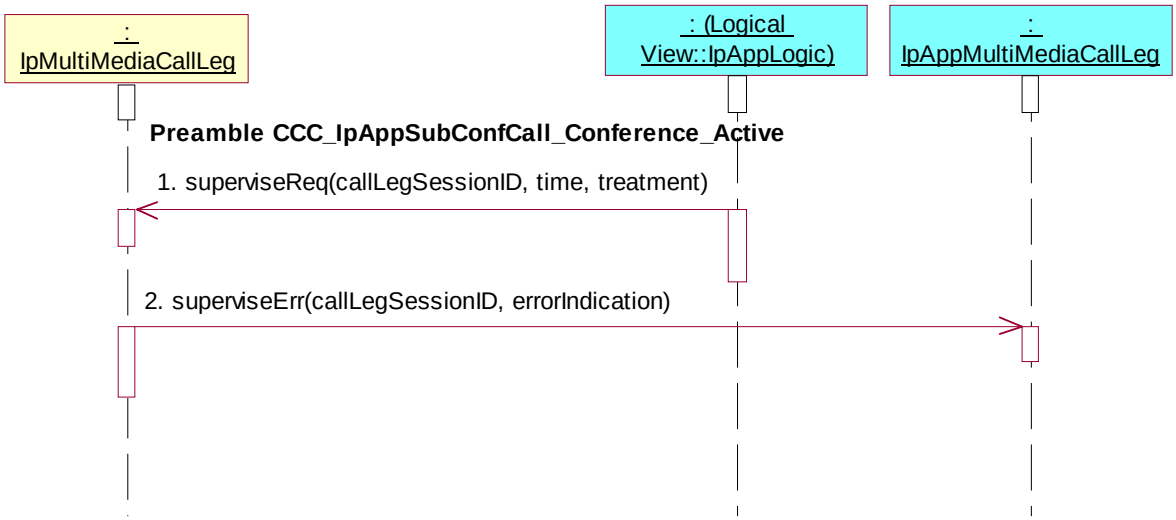
Precondition: IUT capable of invoking **superviseReq()**

Preamble: **CCC_IpAppSubConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, time, treatment

2. Method call **superviseErr()**
   Parameters: callLegSessionID, errorIndication
   Check: no exception is returned



Test CCC_IpAppMultiMediaCallLeg_18

Summary: set monitor on media streams

Reference: ES 202 915-4-3 [3], clause 7.3.2 and ES 202 915-4-4 [4], clauses 6.5 and 6.6

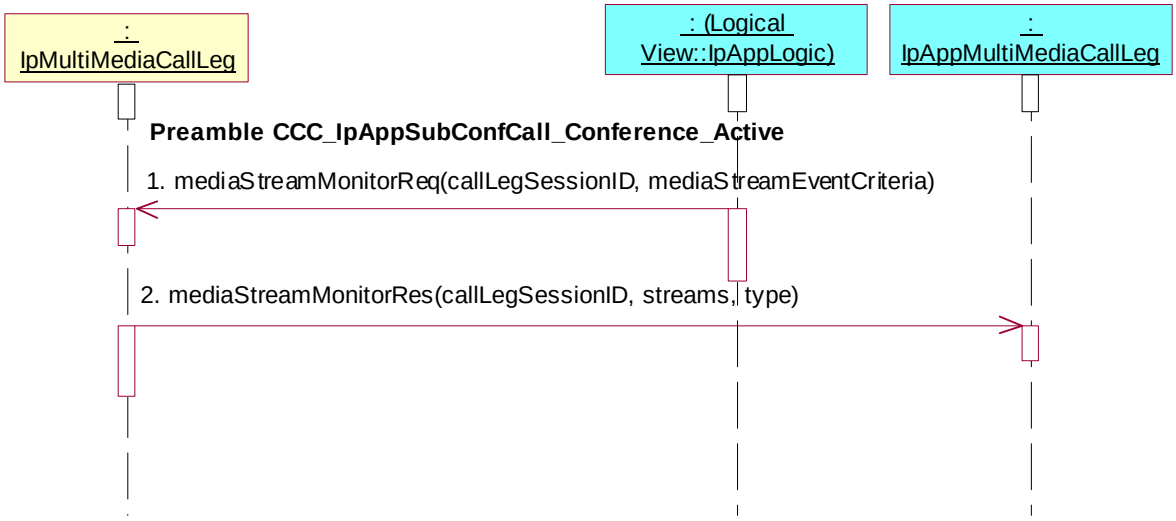
Precondition: IUT capable of invoking **mediaStreamMonitorReq()**

Preamble: **CCC_IpAppSubConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **mediaStreamMonitorReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, mediaStreamEventCriteria

2. Method call **mediaStreamMonitorRes()**
   Parameters: callLegSessionID, streams, type
   Check: no exception is returned



Test CCC_IpAppMultiMediaCallLeg_19

Summary: set monitor on media streams and allow setup

Reference: ES 202 915-4-3 [3], clause 7.3.2 and ES 202 915-4-4 [4], clauses 6.5 and 6.6

Precondition: IUT capable of invoking **mediaStreamMonitorReq()** and **mediaStreamAllow()**

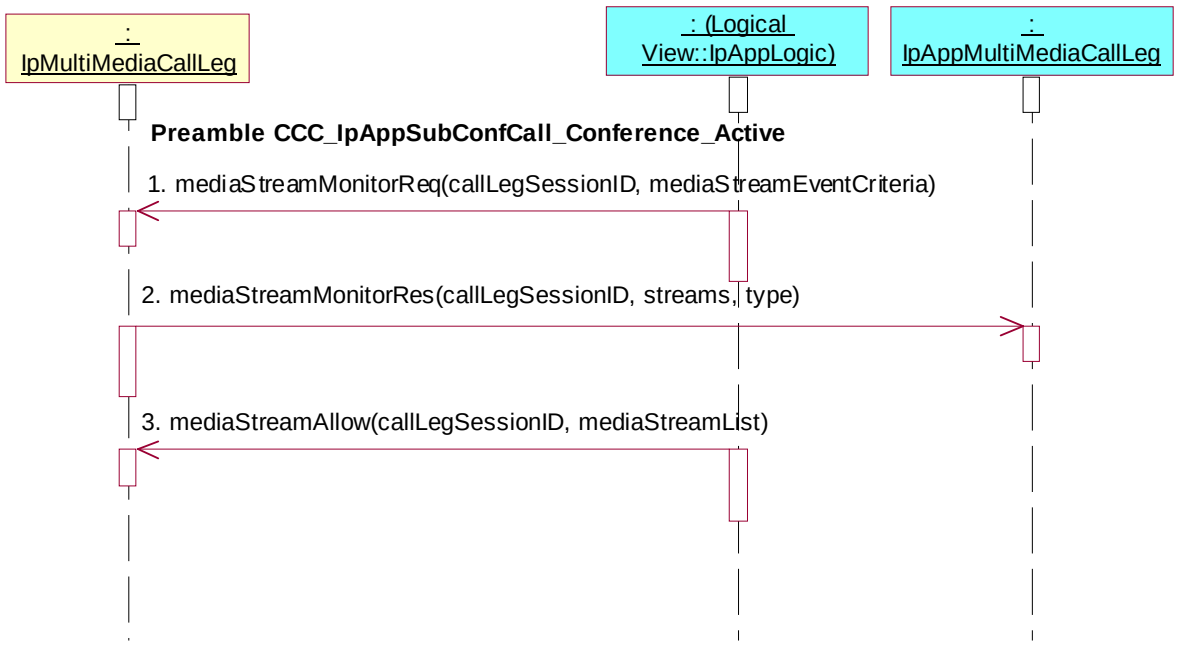
Preamble: **CCC_IpAppSubConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **mediaStreamMonitorReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, mediaStreamEventCriteria
   mediaStreamEventCriteria.CallMonitorMode= P_CALL_MONITOR_MODE_INTERRUPT

2. Method call **mediaStreamMonitorRes()**
   Parameters: callLegSessionID, streams, type
   Check: no exception is returned

3. Triggered Action: cause IUT to call **mediaStreamAllow()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, mediaStreamList



Test CCC_IpAppMultiMediaCallLeg_20

Summary: get media streams

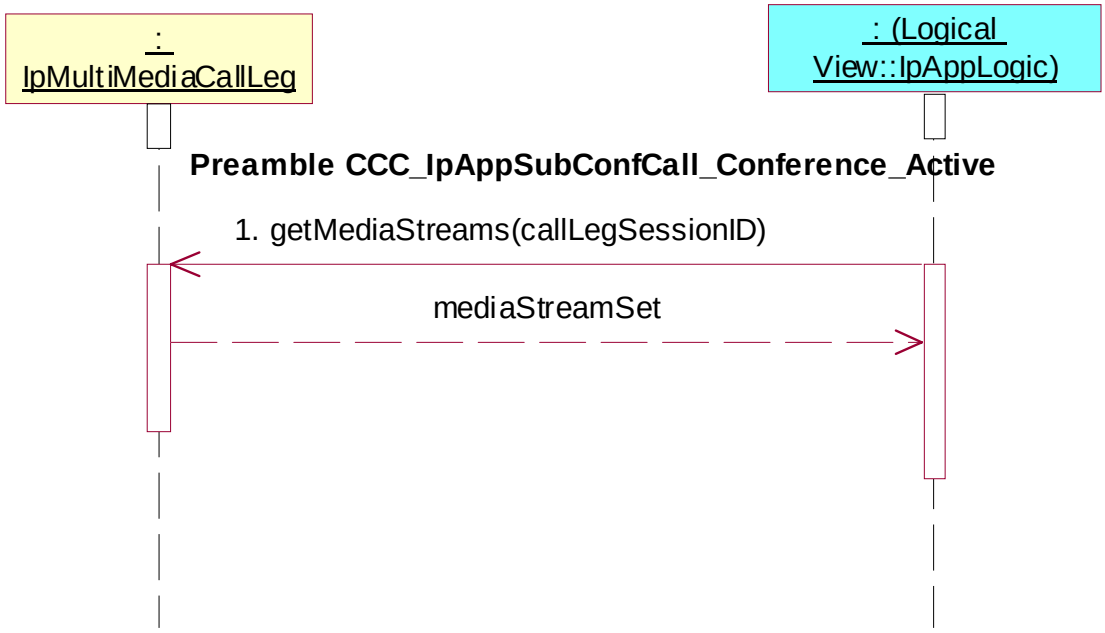
Reference: ES 202 915-4-3 [3], clause 7.3.2 and ES 202 915-4-4 [4], clauses 6.5 and 6.6

Precondition: IUT capable of invoking **getMediaStreams()**

Preamble: **CCC_IpAppSubConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **getMediaStreams()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID



7.2.4.5 IpMultiMediaStream

Test CCC_IpAppMultiMediaStream_01

Summary: substract media stream

Reference: ES 202 915-4-4 [4], clause 6.7

Precondition: IUT capable of invoking **mediaStreamMonitorReq()** and **substract()**

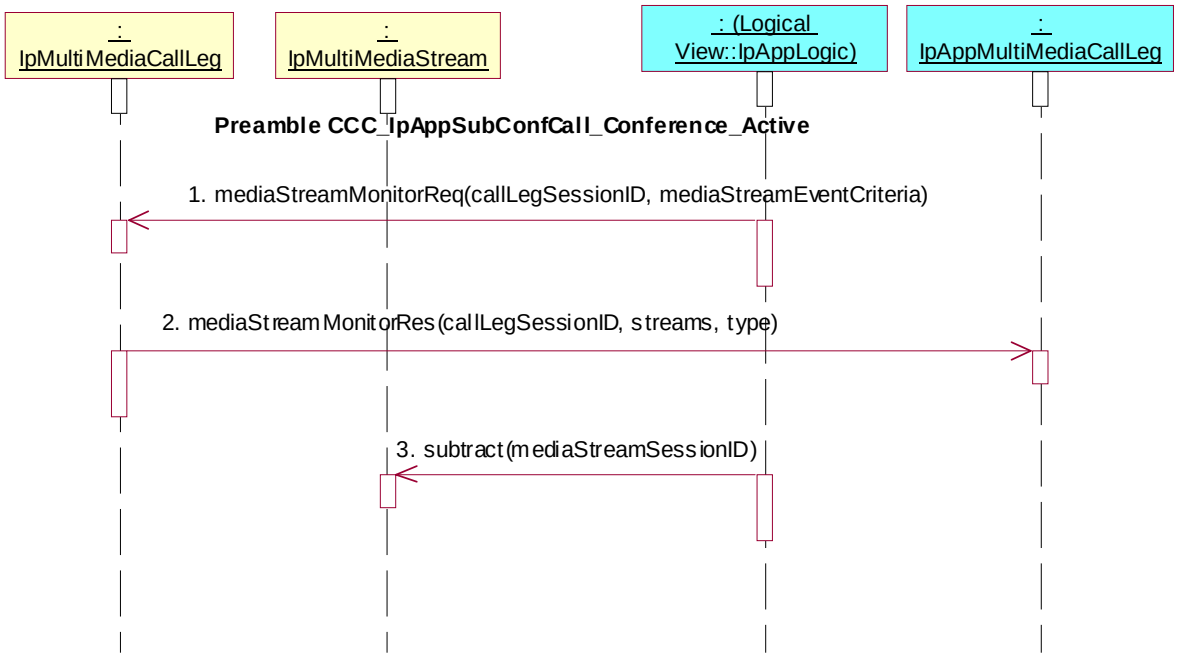
Preamble: **CCC_IpAppSubConfCall_Conference_Active**

Test Sequence:

1. Triggered Action: cause IUT to call **mediaStreamMonitorReq()** method on the tester's (SCF's) IpMultiMediaCallLeg interface.
   Parameters: callLegSessionID, mediaStreamEventCriteria

2. Method call **mediaStreamMonitorRes()**
   Parameters: callLegSessionID, streams, type
   Check: no exception is returned

3. Triggered Action: cause IUT to call **substract()** method on the tester's (SCF's) IpMultiMediaStream interface.
   Parameters: mediaStreamSessionID



History

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