Final draft ETSI ES 202 388-8 V1.1.1 (2005-01)

ETSI Standard

Open Service Access (OSA);
Application Programming Interface (API);
Test Suite Structure and Test Purposes (TSS&TP);
Part 8: Data Session Control SCF;
(Parlay 4)

ETSI

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

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

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

Important notice

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

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

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

If you find errors in the present document, please send your comment to one of the following services:
http://portal.etsi.org/chaircor/ETSI_support.asp

Copyright Notification

Contents

Intellectual Property Rights

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 8 of a multi-part deliverable. Full details of the entire series can be found in part 1 [6].

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

1 Scope

The present document provides the Test Suite Structure and Test Purposes (TSS&TP) specification for the Data Session Control SCF of the Application Programming Interface (API) for Open Service Access (OSA) defined in ES 202 915-8 [1] in compliance with the relevant requirements, and in accordance with the relevant guidance given in ISO/IEC 9646-2 [4] and ETS 300 406 [5].

2 References

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

- References are either specific (identified by date of publication and/or edition number or version number) or non-specific.

- For a specific reference, subsequent revisions do not apply.

- For a non-specific reference, the latest version applies.

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

[1] ETSI ES 202 915-8: "Open Service Access (OSA); Application Programming Interface (API); Part 8: Data Session Control SCF (Parlay 4)".

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

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

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

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

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

3 Definitions and abbreviations

3.1 Definitions

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

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

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

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

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

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

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

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

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

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

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

3.2   Abbreviations

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

API	Application Programming Interface
ATM	Abstract Test Method
ATS	Abstract Test Suite
CM	Control Manager
DS	Data Session
DSC	Data Session Control
ICS	Implementation Conformance Statement
IUT	Implementation Under Test
IXIT	Implementation eXtra Information for Testing
LT	Lower Tester
OSA	Open Service Access
SCF	Service Capability Feature
TP	Test Purpose
TSS	Test Suite Structure

4   Test Suite Structure (TSS)

Data Session Control SCF

- IpDataSessionControlManager (CM)

- IpDataSession (DS)

5   Test Purposes (TP)

5.1   Introduction

For each test requirement a TP is defined.

5.1.1   TP naming convention

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

Table 1: TP identifier naming convention scheme

Identifier: <suite_id>_<group>_<nnn>		
<suite_id>	= SCF name:	"DSC" for **D**ata **S**ession **C**ontrol SCF
<group>	= group number:	two character field representing the group reference according to TSS
<nn>	= sequential number:	(01 to 99)

5.1.2   Source of TP definition

The TPs are based on ES 202 915-8 [1].

5.1.3   Test strategy

As the base standard ES 202 915-8 [1] contains no explicit requirements for testing, the TPs were generated as a result of an analysis of the base standard and the ICS specification ES 202 363 [2].

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

5.2   TPs for the Data Session Control SCF

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

All parameters specified in method calls are valid unless specified.

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

5.2.1   Data Session Control, SCF side

5.2.1.1   IpDataSessionControlManager

Test DSC_CM_01

Summary: **IpDataSessionControlManager**, mandatory methods, successful.

Reference: ES 202 915-8 [1], clause 8.4.

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

Test Sequence:

1.   Method call **createNotifications()**
     Parameters: appDataSessionControlManager (non-NULL), eventCriteria
     Check: valid value of TpAssignmentID is returned

2.   Triggered Action: cause IUT to call **reportNotification()** method on the tester's (application's) IpAppDataSessionControlManager interface.
     Parameters: dataSessionReference, eventInfo, assignmentID

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

Test DSC_CM_02

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

Reference: ES 202 915-8 [1], clause 8.4.

Precondition: **changeNotification()** and **getNotifications()** implemented.

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

Test Sequence:

1.   Method call **createNotifications()**
     Parameters: appDataSessionControlManager (non-NULL), eventCriteria
     Check: valid value of TpAssignmentID is returned

2.   Method call **getNotifications()**
     Parameters: none
     Check: value of TpDataSessionEventCriteriaResultSet containing event criteria as specified in 1. is
                   returned

3.   Method call **changeNotification()**
     Parameters: assignmentID, eventCriteria (different to 1.)
     Check: no exception is returned

4.   Method call **getNotifications()**
     Parameters: none
     Check: value of TpDataSessionEventCriteriaResultSet containing event criteria as specified in 3. is
                   returned

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

6.   Method call **destroyNotification()**
     Parameters: assignmentID as received in1. and destroyed in 5.
     Check: P_INVALID_ASSIGNMENT_ID is returned

Test DSC_CM_03

Summary: **IpDataSessionControlManager**, createNotification(), P_INVALID_CRITERIA exception.

Reference: ES 202 915-8 [1], clauses 7.4.1 and 8.3.

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

Test Sequence:

1. Method call **createNotification()**
   Parameters: appDataSessionControlManager (non-NULL), invalid eventCriteria
   Check: P_INVALID_CRITERIA is returned

Test DSC_CM_04

Summary: **IpDataSessionControlManager**, destroyNotification(), P_INVALID_ASSIGMENT_ID exception.

Reference: ES 202 915-8 [1], clauses 7.4.1 and 8.3.

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

Test Sequence:

1. Method call **createNotification()**
    Parameters: appDataSessionControlManager (non-NULL), eventCriteria
    Check: valid value of TpAssignmentID is returned

2. Method call **destroyNotification()**
    Parameters: invalid assignmentID
    Check: P_INVALID_ASSIGMENT_ID is returned

Test DSC_CM_05

Summary: **IpDataSessionControlManager**, changeNotification(), P_INVALID_ASSIGMENT_ID exception.

Reference: ES 202 915-8 [1], clauses 7.4.1 and 8.3.

Precondition: **changeNotification()** implemented.

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

Test Sequence:

   1.   Method call **createNotification()**
        Parameters: appDataSessionControlManager (non-NULL), eventCriteria
        Check: valid value of TpAssignmentID is returned

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

```
        IpAppDataSessionControlManager                    IpDataSessionControlManager

              1. createNotification(appDataSessionControlManager, eventCriteria)

                              assignmentID


              2. changeNotification(invalid assignmentID)

                         P_INVALID_ASSIGNMENT exception
```
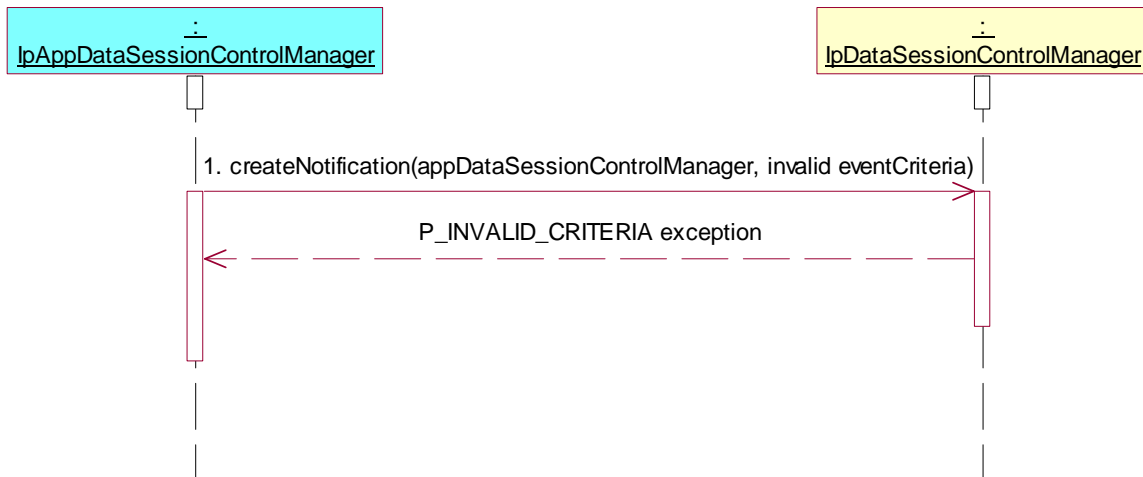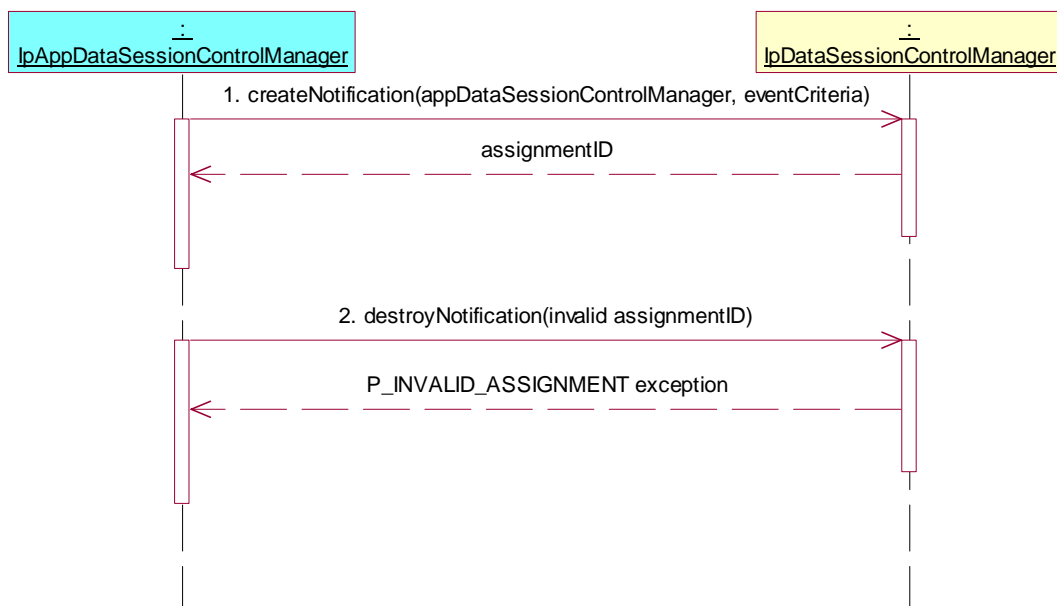
Test DSC_CM_06

Summary: **IpDataSessionControlManager**, changeNotification(), P_INVALID_CRITERIA exception.
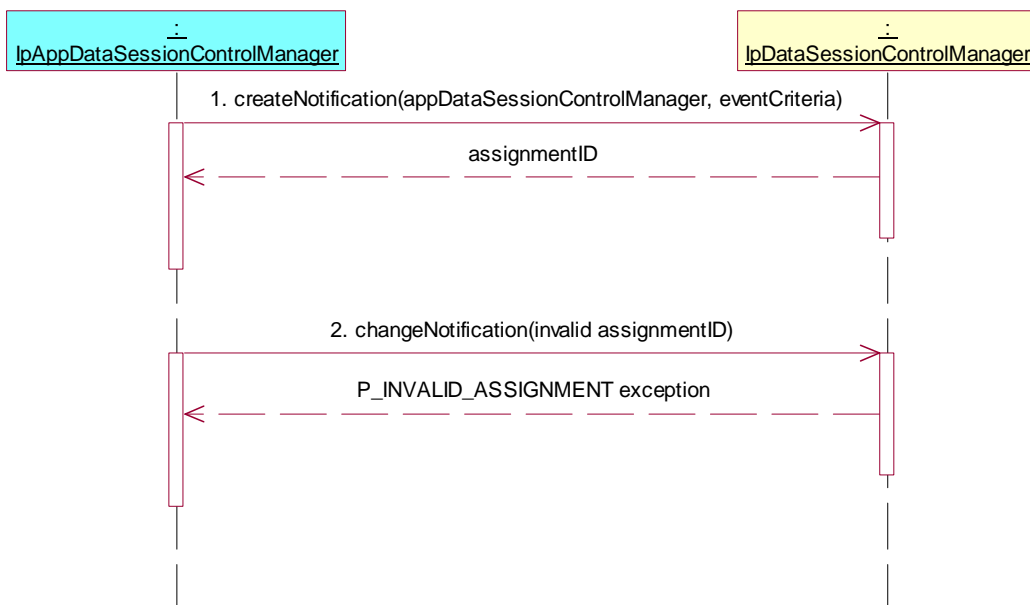
Reference: ES 202 915-8 [1], clauses 7.4.1 and 8.3.

Precondition: **changeNotification()** implemented.

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

Test Sequence:

   1.   Method call **createNotification()**
        Parameters: appDataSessionControlManager (non-NULL), eventCriteria
        Check: valid value of TpAssignmentID is returned

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

```
        :                                              :
IpAppDataSessionControlManager          IpDataSessionControlManager
```

1. createNotification(appDataSessionControlManager, eventCriteria)

assignmentID

2. changeNotification(invalid eventCriteria)

P_INVALID_CRITERIA exception

Test DSC_CM_07

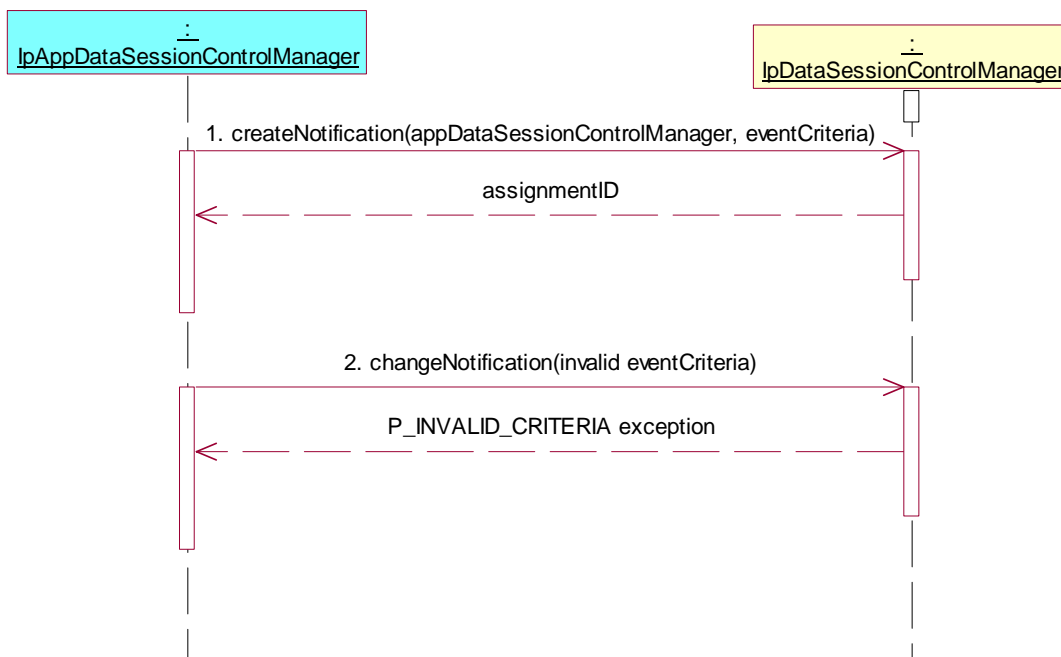Summary: All methods, successful, two createNotification() method calls.

Reference: ES 202 915-8 [1], clause 8.4.

Precondition: **getNotifications()** implemented.

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

Test Sequence:

1.   Method call **createNotifications()**
      Parameters: appDataSessionControlManager (non-NULL), eventCriteria
      Check: valid value of TpAssignmentID is returned

2.   Method call **getNotifications()**
      Parameters: none
      Check: value of TpDataSessionEventCriteriaResultSet containing event criteria as specified in 1. is
                     returned

3.   Method call **createNotifications()**
      Parameters: appDataSessionControlManager (non-NULL) different from 1., eventCriteria same as in 1.
      Check: same value of TpAssignmentID as in 1. is returned

4.   Triggered Action: cause IUT to call **reportNotification()** method on the tester's (application's)
      IpAppDataSessionControlManager interface
      Parameters: dataSessionReference, eventInfo, assignmentID
      Check: The interface given in 3. receives the event

5.   Triggered action: Kill the appDataSessionControlManager item referenced in 3.

6.   Triggered Action: cause IUT to call **reportNotification()** method on the tester's (application's)
      IpAppDataSessionControlManager interface.
      Parameters: dataSessionReference, eventInfo, assignmentID
      Check: The interface given in 1. receives the event.

Test DSC_CM_08

Summary: All methods, successful, two createNotifications() method calls.
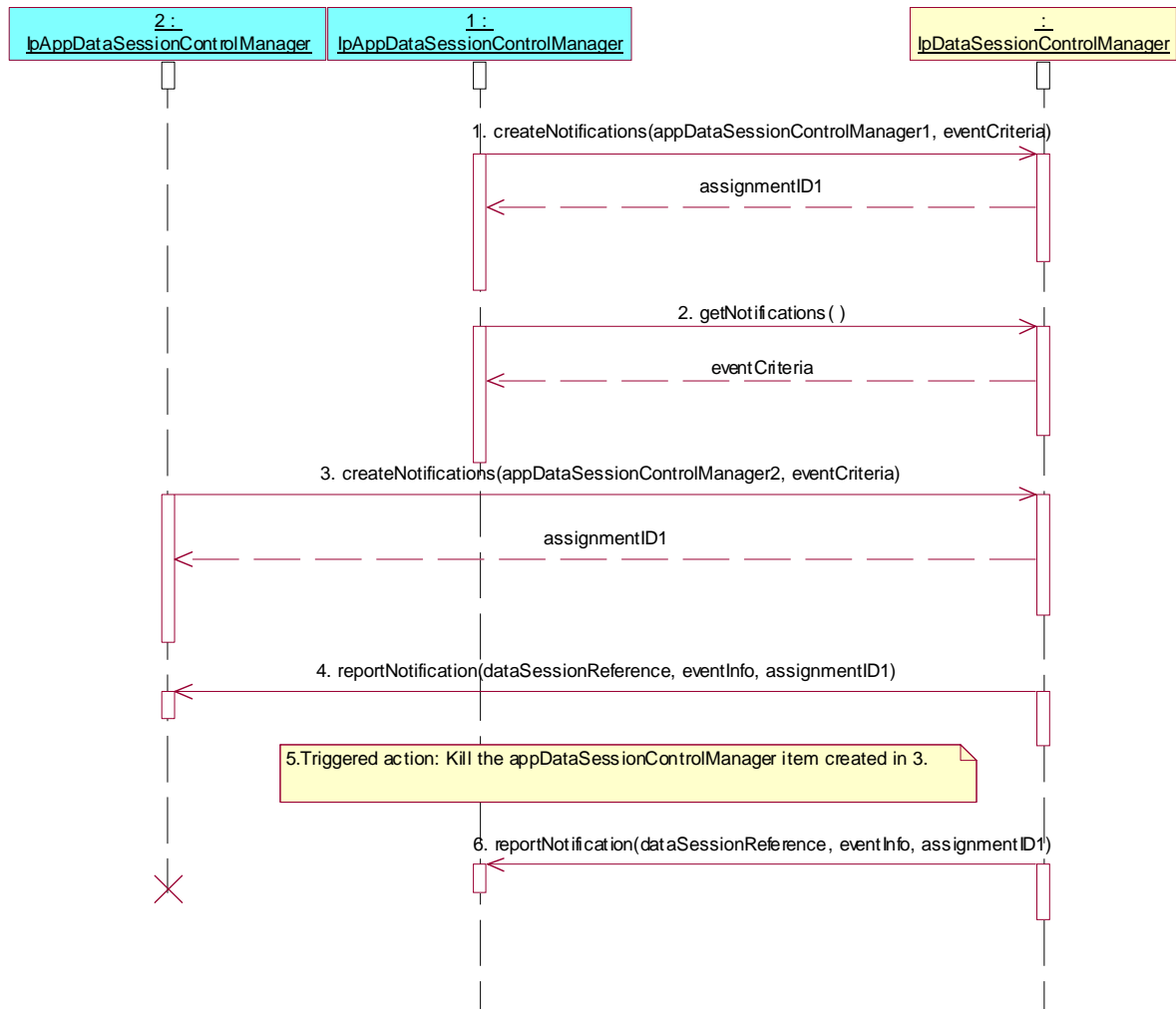
Reference: ES 202 915-8 [1], clause 8.4

Precondition: **enableNotifications()** implemented

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

Test Sequence:

1. Method call **enableNotifications()**
   Parameters: None
   Check: valid value of TpDataSessionEventCriteriaResultSet is returned

2. Triggered Action: cause IUT to call **reportNotification()** method on the tester's (application's) IpAppDataSessionControlManager interface
   Parameters: dataSessionReference, eventInfo, assignmentID

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

5.2.1.2   IpDataSession

Test DSC_DS_01

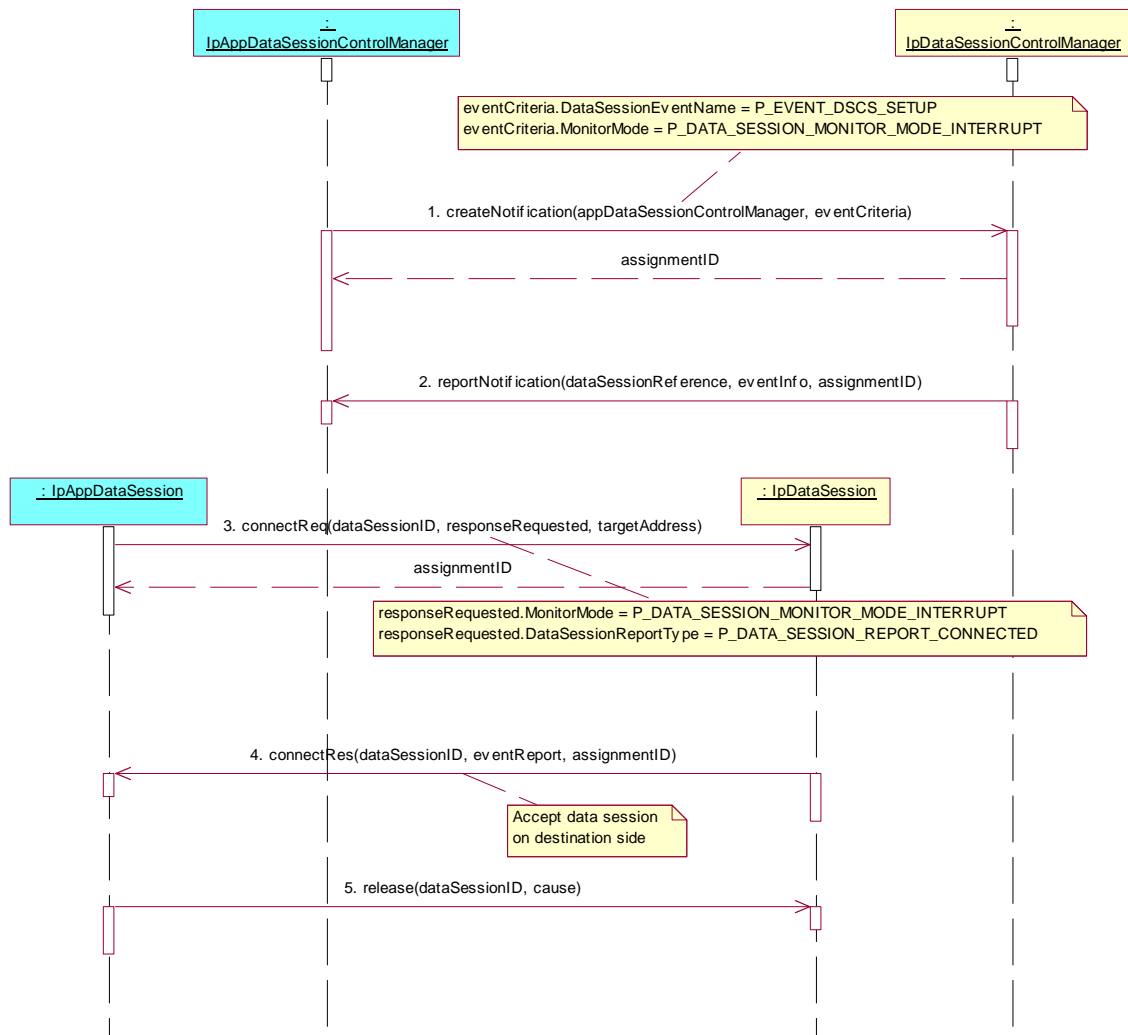Summary: **IpDataSession**, release after connectReq(), successful with interrupt after reportNotification().

Reference: ES 202 915-8 [1], clauses 7.4.1 and 8.3.

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

Test Sequence:

1.   Method call **createNotifications()**
     Parameters: appDataSessionControlManager (non-NULL), eventCriteria
     Parameter values:
                     eventCriteria.DataSessionEventName = P_EVENT_DSCS_SETUP
                     eventCriteria.MonitorMode = P_DATA_SESSION_MONITOR_MODE_INTERRUPT
     Check: valid value of TpAssignmentID is returned

2.   Triggered action: cause IUT to call **reportNotification()** method on the tester's (application's)
     IpAppDataSessionControlManager interface, i.e. start to set-up data session
     Parameters: dataSessionReference, eventInfo, assignmentID

3.   Method call **connectReq()**
     Parameters: dataSessionID, responseRequested, targetAddress
     Parameter values:
                     responseRequested.MonitorMode = P_DATA_SESSION_MONITOR_MODE_INTERRUPT
                     responseRequested.DataSessionReportType = P_DATA_SESSION_REPORT_CONNECTED
     Check: valid value of TpAssignmentID is returned

4.   Triggered action: cause IUT to call **connectRes()** method on the tester's (application's) **IpAppDataSession**
     interface, i.e. accept the data session on the destination side
     Parameters: dataSessionID, eventReport, assignmentID

5.   Method call **release()**
     Parameters: dataSessionID, cause
     Check: no exception is returned

Test DSC_DS_02

Summary: **IpDataSession**, deassignDataSession() after connectReq(), successful.

Reference: ES 202 915-8 [1], clauses 7.4.1 and 8.3

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

Test Sequence:

1.   Method call **createNotifications()**
     Parameters: appDataSessionControlManager (non-NULL), eventCriteria
     Parameter values:
              eventCriteria.DataSessionEventName = P_EVENT_DSCS_SETUP
              eventCriteria.MonitorMode = P_DATA_SESSION_MONITOR_MODE_INTERRUPT
     Check: valid value of TpAssignmentID is returned

2.   Triggered action: cause IUT to call **reportNotification()** method on the tester's (application's)
     IpAppDataSessionControlManager interface, i.e. start to set-up data session
     Parameters: dataSessionReference, eventInfo, assignmentID

3. Method call **connectReq()**
   Parameters: dataSessionID, responseRequested, targetAddress
   Parameter values:
           responseRequested.MonitorMode = P_DATA_SESSION_MONITOR_MODE_NOTIFY
           responseRequested.DataSessionReportType = P_DATA_SESSION_REPORT_CONNECTED
   Check: valid value of TpAssignmentID is returned

4. Triggered action: cause IUT to call **connectRes()** method on the tester's (application's) **IpAppDataSession**
   interface, i.e. accept the data session on the destination side
   Parameters: dataSessionID, eventReport, assignmentID

5. Method call **deassignDataSession()**
   Parameters: dataSessionID
   Check: no exception is returned

Test DSC_DS_03

Summary: **IpDataSession**, connectReq() with two trigger events, successful with interrupt after
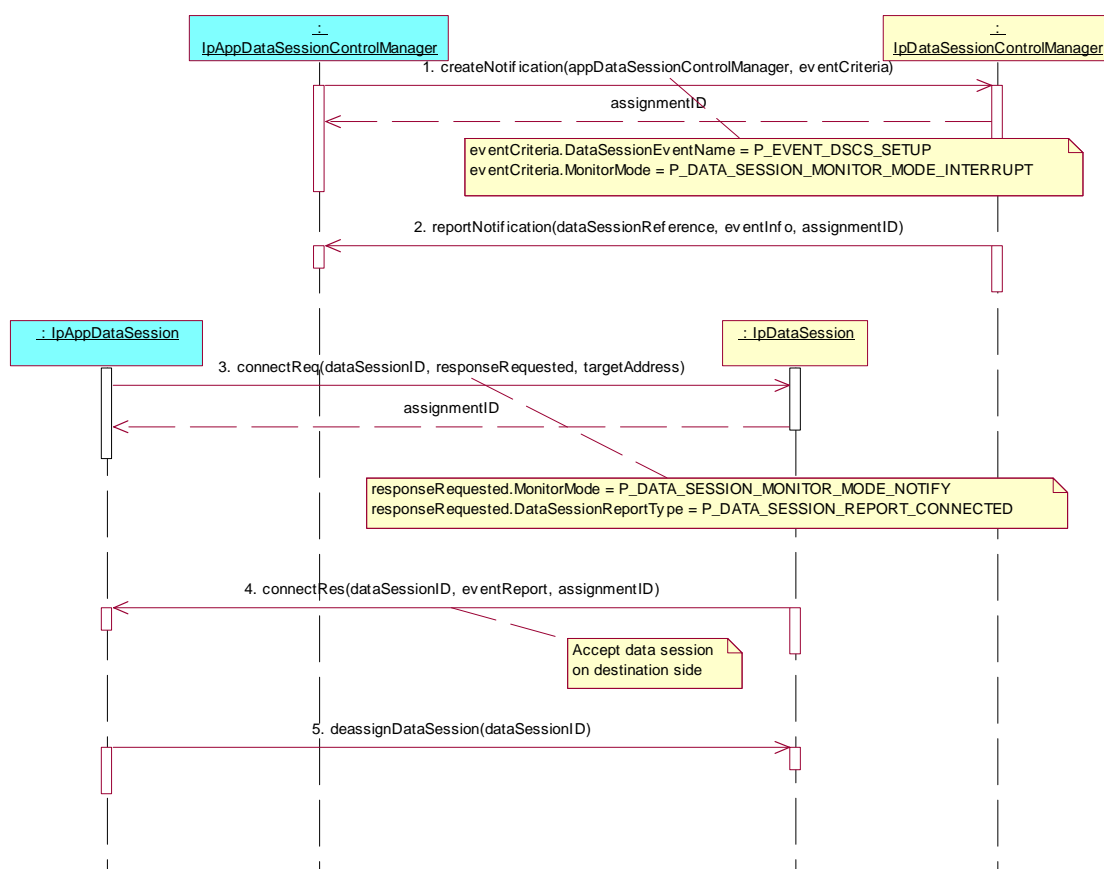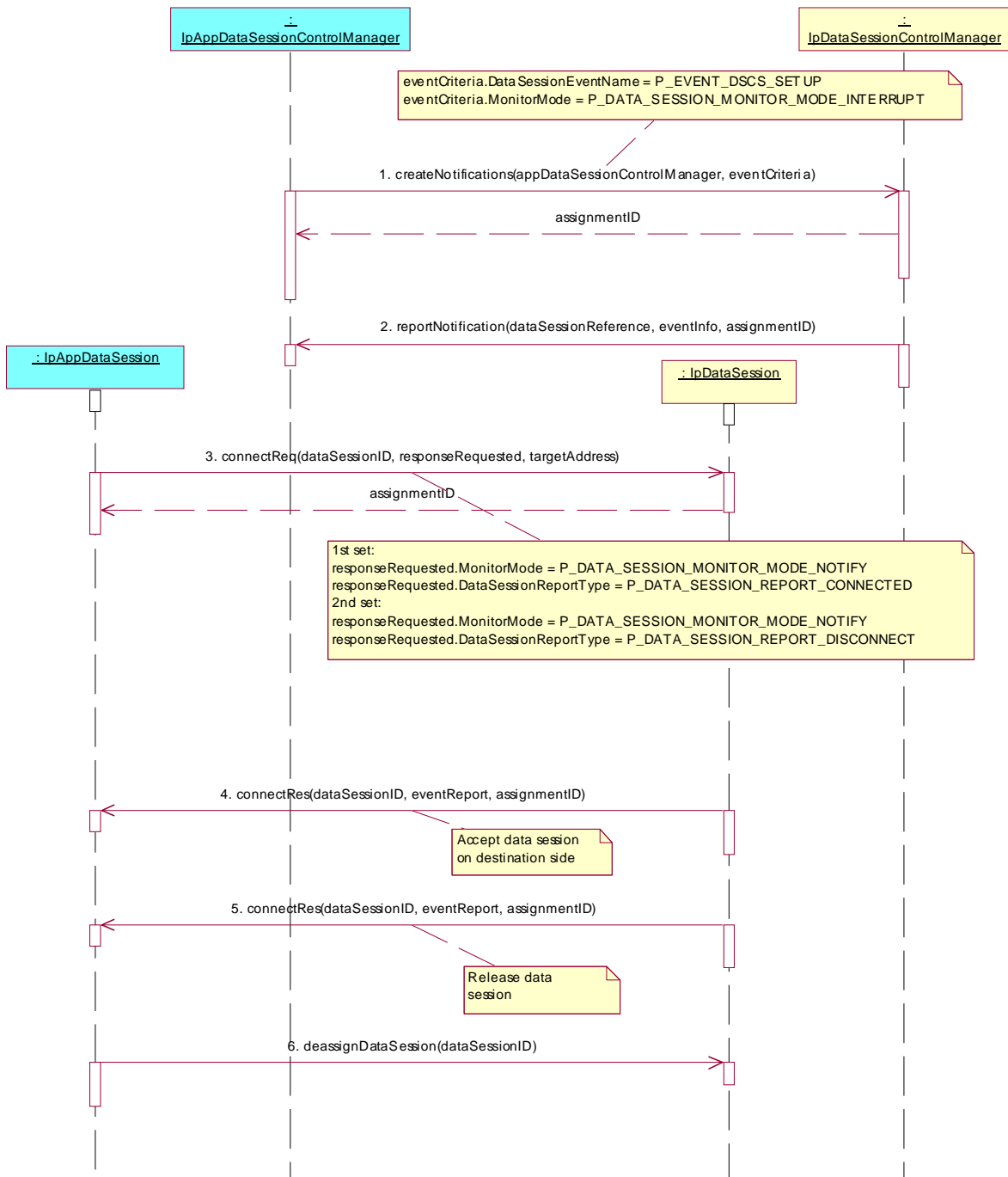   reportNotification().

Reference: ES 202 915-8 [1], clauses 7.4.1 and 8.3.

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

Test Sequence:

1.   Method call **createNotifications()**
      Parameters: appDataSessionControlManager (non-NULL), eventCriteria
      Parameter values:
                  eventCriteria.DataSessionEventName = P_EVENT_DSCS_SETUP
                  eventCriteria.MonitorMode = P_DATA_SESSION_MONITOR_MODE_INTERRUPT
      Check: valid value of TpAssignmentID is returned

2.   Triggered action: cause IUT to call **reportNotification()** method on the tester's (application's)
      IpAppDataSessionControlManager interface, i.e. start to set-up data session
      Parameters: dataSessionReference, eventInfo, assignmentID

3.   Method call **connectReq()**
      Parameters: dataSessionID, responseRequested, targetAddress
      Parameter values:
            1st set
                  responseRequested.MonitorMode = P_DATA_SESSION_MONITOR_MODE_NOTIFY
                  responseRequested.DataSessionReportType = P_DATA_SESSION_REPORT_CONNECTED
            2nd set
                  responseRequested.MonitorMode = P_DATA_SESSION_MONITOR_MODE_NOTIFY
                  responseRequested.DataSessionReportType = P_DATA_SESSION_REPORT_DISCONNECT
      Check: valid value of TpAssignmentID is returned

4.   Triggered action: cause IUT to call **connectRes()** method on the tester's (application's) **IpAppDataSession**
      interface, i.e. accept the data session on the destination side
      Parameters: dataSessionID, eventReport, assignmentID

5.   Triggered action: cause IUT to call **connectRes()** method on the tester's (application's) **IpAppDataSession**
      interface, i.e. release the data session
      Parameters: dataSessionID, eventReport, assignmentID

6.   Method call **deassignDataSession()**
      Parameters: dataSessionID
      Check: no exception is returned

```
          : IpAppDataSessionControlManager                                          : IpDataSessionControlManager
```

> eventCriteria.DataSessionEventName = P_EVENT_DSCS_SETUP
> eventCriteria.MonitorMode = P_DATA_SESSION_MONITOR_MODE_INTERRUPT

1. createNotifications(appDataSessionControlManager, eventCriteria)

assignmentID

2. reportNotification(dataSessionReference, eventInfo, assignmentID)

```
      : IpAppDataSession                               : IpDataSession
```

3. connectReq(dataSessionID, responseRequested, targetAddress)

assignmentID

> 1st set:
> responseRequested.MonitorMode = P_DATA_SESSION_MONITOR_MODE_NOTIFY
> responseRequested.DataSessionReportType = P_DATA_SESSION_REPORT_CONNECTED
> 2nd set:
> responseRequested.MonitorMode = P_DATA_SESSION_MONITOR_MODE_NOTIFY
> responseRequested.DataSessionReportType = P_DATA_SESSION_REPORT_DISCONNECT

4. connectRes(dataSessionID, eventReport, assignmentID)

> Accept data session
> on destination side

5. connectRes(dataSessionID, eventReport, assignmentID)

> Release data
> session

6. deassignDataSession(dataSessionID)

Test DSC_DS_04

Summary: **IpDataSession**, superviseDataSessionReq().
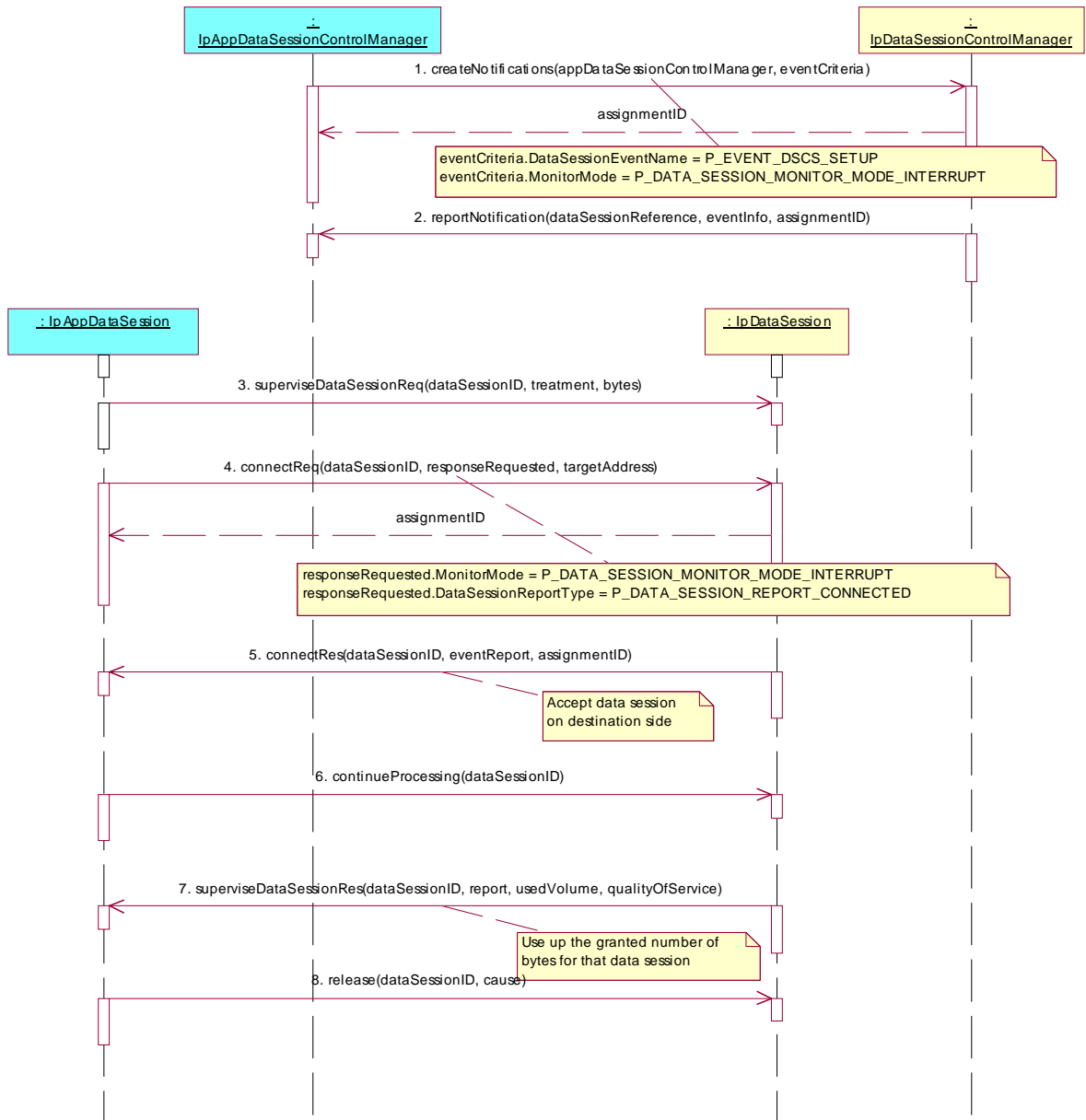
Reference: ES 202 915-8 [1], clauses 7.4.1 and 8.3.

Precondition: **superviseDataSessionReq()** implemented.

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

Test Sequence:

1. Method call **createNotifications()**
    Parameters: appDataSessionControlManager (non-NULL), eventCriteria
    Parameter values:
                    eventCriteria.DataSessionEventName = P_EVENT_DSCS_SETUP
                    eventCriteria.MonitorMode = P_DATA_SESSION_MONITOR_MODE_INTERRUPT
    Check: valid value of TpAssignmentID is returned

2. Triggered action: cause IUT to call **reportNotification()** method on the tester's (application's) **IpAppDataSessionControlManager** interface, i.e. start to set-up data session
    Parameters: dataSessionReference, eventInfo, assignmentID

3. Method call **superviseDataSessionReq()**
    Parameters: dataSessionID, treatment, bytes
    Check: no exception is returned

4. Method call **connectReq()**
    Parameters: dataSessionID, responseRequested, targetAddress
    Parameter values:
                    responseRequested.MonitorMode = P_DATA_SESSION_MONITOR_MODE_INTERRUPT
                    responseRequested.DataSessionReportType = P_DATA_SESSION_REPORT_CONNECTED
    Check: valid value of TpAssignmentID is returned

5. Triggered action: cause IUT to call **connectRes()** method on the tester's (application's) **IpAppDataSession** interface, i.e. accept the data session on the destination side
    Parameters: dataSessionID, eventReport, assignmentID

6. Method call **continueProcessing()**
    Parameters: dataSessionID
    Check: no exception is returned

7. Triggered action: cause IUT to call **superviseDataSessionRes()** method on the tester's (application's) **IpAppDataSession** interface, i.e. use up the granted number of bytes for that data session
    Parameters: dataSessionID, report, usedVolume, qualityOfService

8. Method call **release()**
    Parameters: dataSessionID, cause
    Check: no exception is returned

Test DSC_DS_05

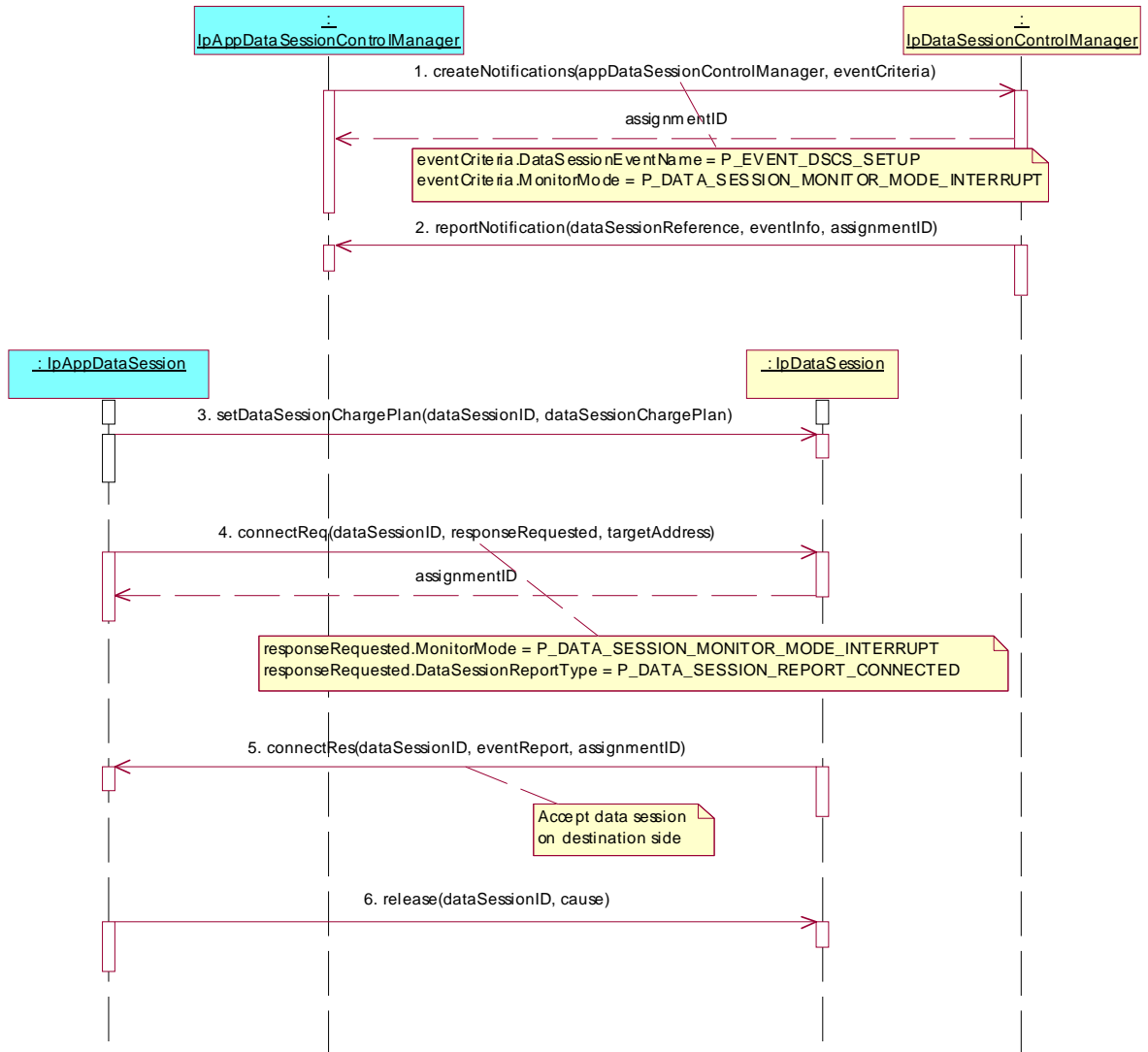Summary: **IpDataSession**, setDataSessionChargePlan().

Reference: ES 202 915-8 [1], clauses 7.4.1 and 8.3.

Precondition: **setDataSessionChargePlan()** implemented.

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

Test Sequence:

1.   Method call **createNotifications()**
     Parameters: appDataSessionControlManager (non-NULL), eventCriteria
     Parameter values:
                     eventCriteria.DataSessionEventName = P_EVENT_DSCS_SETUP
                     eventCriteria.MonitorMode = P_DATA_SESSION_MONITOR_MODE_INTERRUPT
     Check: valid value of TpAssignmentID is returned

2.   Triggered action: cause IUT to call **reportNotification()** method on the tester's (application's) **IpAppDataSessionControlManager** interface, i.e. start to set-up data session
     Parameters: dataSessionReference, eventInfo, assignmentID

3.   Method call **setDataSessionChargePlan()**
     Parameters: dataSessionID, dataSessionChargePlan
     Check: no exception is returned

4.   Method call **connectReq()**
     Parameters: dataSessionID, responseRequested, targetAddress
     Parameter values:
                     responseRequested.MonitorMode = P_DATA_SESSION_MONITOR_MODE_INTERRUPT
                     responseRequested.DataSessionReportType = P_DATA_SESSION_REPORT_CONNECTED
     Check: valid value of TpAssignmentID is returned

5.   Triggered action: cause IUT to call **connectRes()** method on the tester's (application's) **IpAppDataSession** interface, i.e. accept the data session on the destination side
     Parameters: dataSessionID, eventReport, assignmentID

6.   Method call **release()**
     Parameters: dataSessionID, cause
     Check: no exception is returned

Test DSC_DS_06

Summary: **IpDataSession**, setAdviceOfCharge().
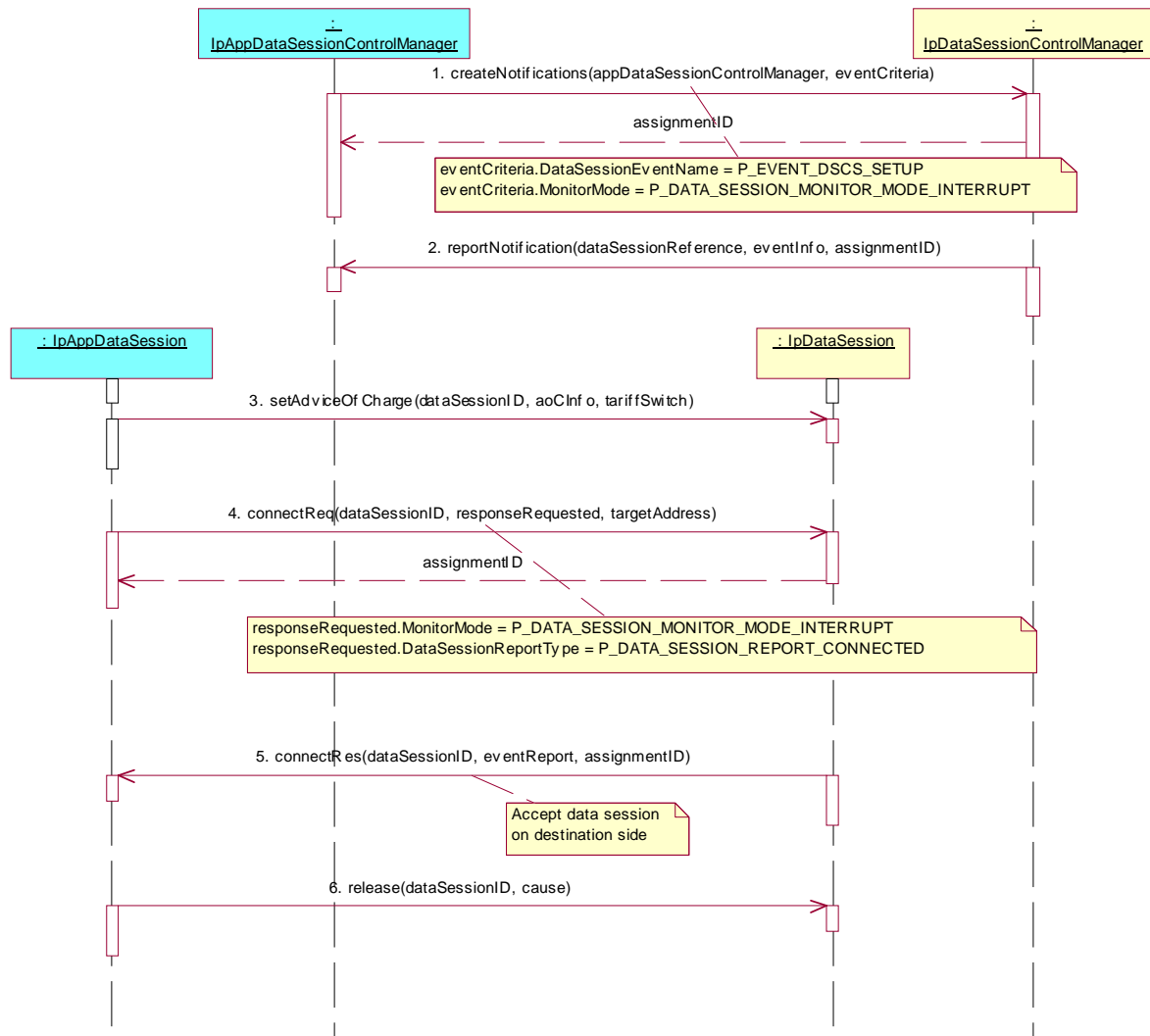
Reference: ES 202 915-8 [1], clauses 7.4.1 and 8.3.

Precondition: **setAdviceOfCharge()** implemented.

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

Test Sequence:

1.   Method call **createNotifications()**
     Parameters: appDataSessionControlManager (non-NULL), eventCriteria
     Parameter values:
                     eventCriteria.DataSessionEventName = P_EVENT_DSCS_SETUP
                     eventCriteria.MonitorMode = P_DATA_SESSION_MONITOR_MODE_INTERRUPT
     Check: valid value of TpAssignmentID is returned

2.   Triggered action: cause IUT to call **reportNotification()** method on the tester's (application's) **IpAppDataSessionControlManager** interface, i.e. start to set-up data session
     Parameters: dataSessionReference, eventInfo, assignmentID

3.   Method call **setAdviceOfCharge()**
     Parameters: dataSessionID, aoCInfo, tariffswitch
     Check: no exception is returned

4.   Method call **connectReq()**
     Parameters: dataSessionID, responseRequested, targetAddress
     Parameter values:
                     responseRequested.MonitorMode = P_DATA_SESSION_MONITOR_MODE_INTERRUPT
                     responseRequested.DataSessionReportType = P_DATA_SESSION_REPORT_CONNECTED
     Check: valid value of TpAssignmentID is returned

5.   Triggered action: cause IUT to call **connectRes()** method on the tester's (application's) **IpAppDataSession** interface, i.e. accept the data session on the destination side
     Parameters: dataSessionID, eventReport, assignmentID

6.   Method call **release()**
     Parameters: dataSessionID, cause
     Check: no exception is returned

5.2.2   Data Session Control, application side

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.

5.2.2.1   IpAppDataSessionControlManager

Test DSC_APP_CM_01

Summary: **IpAppDataSessionControlManager**, mandatory methods, successful.

Reference: ES 202 915-8 [1], clauses 8.2 and 8.4.

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

Instead of providing a valid value of its IpAppDataSessionControlManager interface reference in the createNotifications() method, the application is permitted to provide the IpAppDataSessionControlManager interface reference in a setCallback() method which it calls prior to invoking createNotifications().

Test Sequence:

1. Triggered Action: cause IUT to call **createNotifications()** method on the tester's (SCF's)
   IpDataSessionControlManager interface.
   Parameters: appDataSessionControlManager (non-NULL), eventCriteria

2. Method call **reportNotification()**
   Parameters: dataSessionReference, value depending on monitor mode of notification request, eventInfo,
   assignmentID
   Check: valid value of IpAppDataSessionRef is returned, depending on monitor mode of notification
   request

```
        ·                                                    ·
IpDataSessionControlManager              IpAppDataSessionControlManager
        |                                                    |
        |  1. createNotifications(appDataSessionControlManager, eventCriteria)  |
        |<───────────────────────────────────────────────── |
        |                                                    |
        |                  assignmentID                      |
        | ─────────────────────────────────────────────────>|
        |                                                    |
        |                                                    |
        |  2. reportNotification(dataSessionReference, eventInfo, assignmentID)  |
        | ─────────────────────────────────────────────────>|
        |                                                    |
        |                  appDataSession                    |
        |<───────────────────────────────────────────────── |
        |                                                    |
        |                                                    |
        |                                                    |
```

Test DSC_APP_CM_02

Summary: **IpAppDataSessionControlManager**, interrupt and continue data session notification, successful.

Reference: ES 202 915-8 [1], clause 8.2.

Precondition: **dataSessionNotificationInterrupted()** and **dataSessionNotificationContinued()** implemented.

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

                Instead of providing a valid value of its IpAppDataSessionControlManager interface reference in the
                createNotifications() method, the application is permitted to provide the
                IpAppDataSessionControlManager interface reference in a setCallback() method which it calls prior to
                invoking createNotifications().
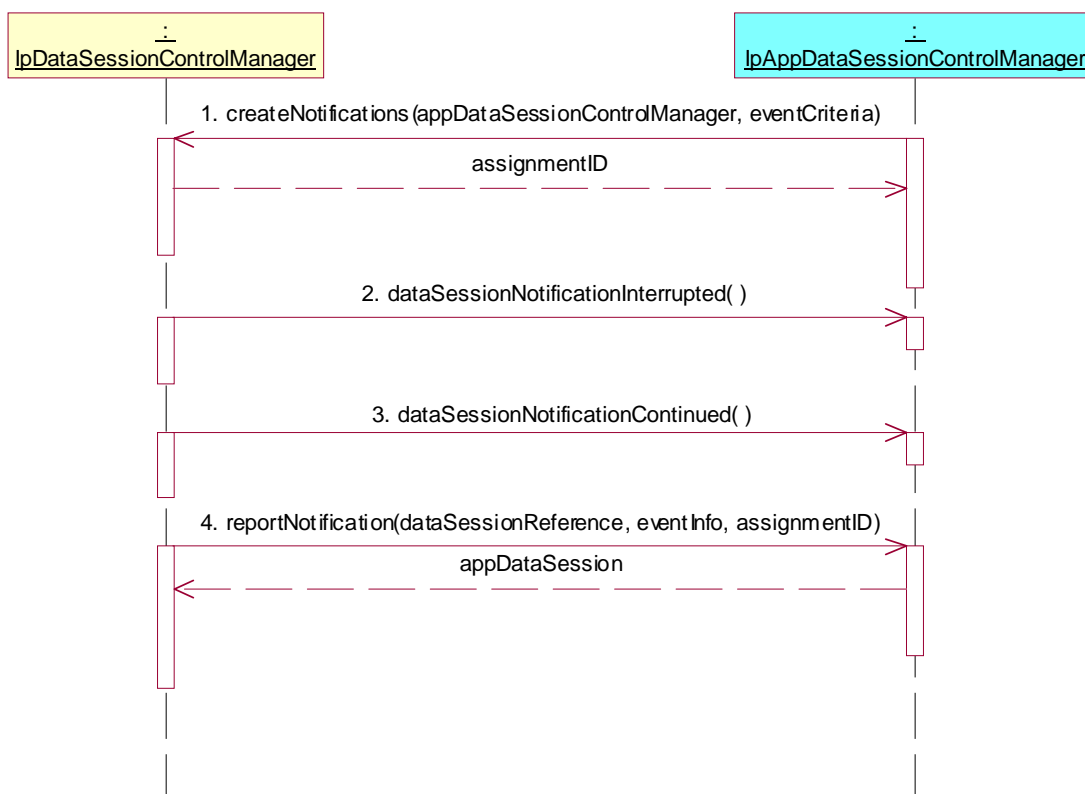
Test Sequence:

1. Triggered Action: cause IUT to call **createNotifications()** method on the tester's (SCF's)
   IpDataSessionControlManager interface.
   Parameters: appDataSessionControlManager (non-NULL), eventCriteria

Delay between createNotifications and Interrupted

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

Delay between Interrupted and Continued

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

4.   Method call **reportNotification()**
      Parameters: dataSessionReference, value depending on monitor mode of notification request, eventInfo,
      assignmentID
      Check: valid value of IpAppDataSessionRef is returned, depending on monitor mode of notification
      request

Test DSC_APP_CM_03

Summary: **IpAppDataSessionControlManager**, abort data session, successful.

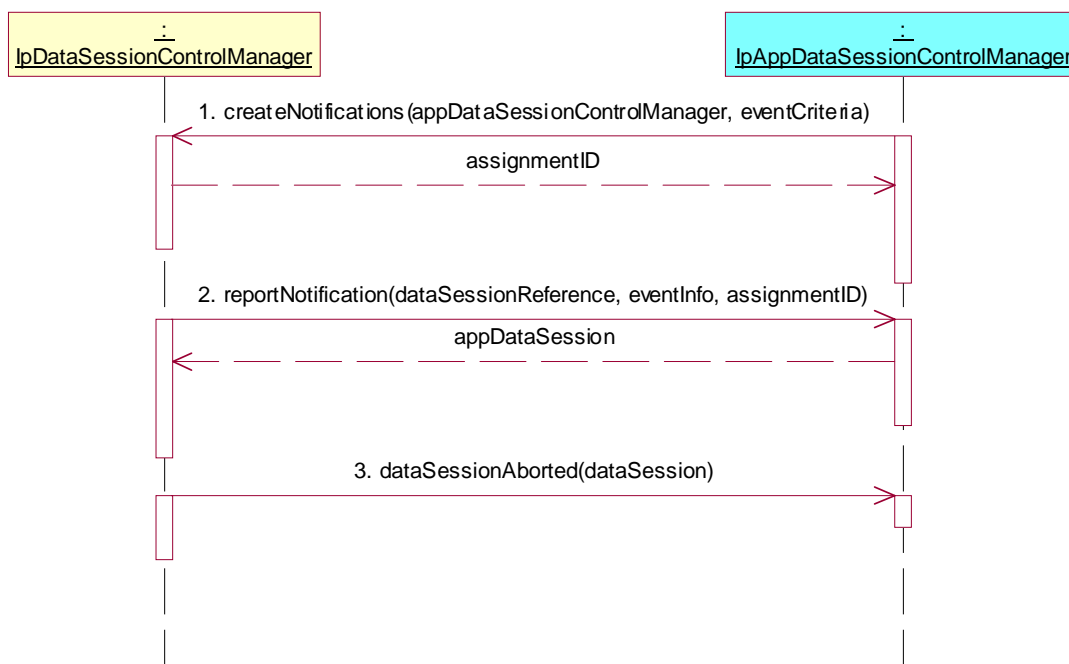Reference: ES 202 915-8 [1], clause 8.2.

Precondition: **dataSessionAborted()** implemented.

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

                Instead of providing a valid value of its IpAppDataSessionControlManager interface reference in the
                createNotifications() method, the application is permitted to provide the
                IpAppDataSessionControlManager interface reference in a setCallback() method which it calls prior to
                invoking createNotifications().

Test Sequence:

1. Triggered Action: cause IUT to call **createNotifications()** method on the tester's (SCF's)
    IpDataSessionControlManager interface.
    Parameters: appDataSessionControlManager (non-NULL), eventCriteria
                    eventCriteria.MonitorMode = P_DATA_SESSION_MONITOR_MODE_INTERRUPT

2. Method call **reportNotification()**
    Parameters: dataSessionReference, eventInfo, assignmentID
    Check: valid value of IpAppDataSessionRef is returned

3. Method call **dataSessionAborted()**
    Parameters: dataSessionID with same value as supplied in reportNotification.
    Check: no exception is returned

Test DSC_APP_CM_04

Summary: **IpAppDataSessionControlManager**, change event criteria, successful.

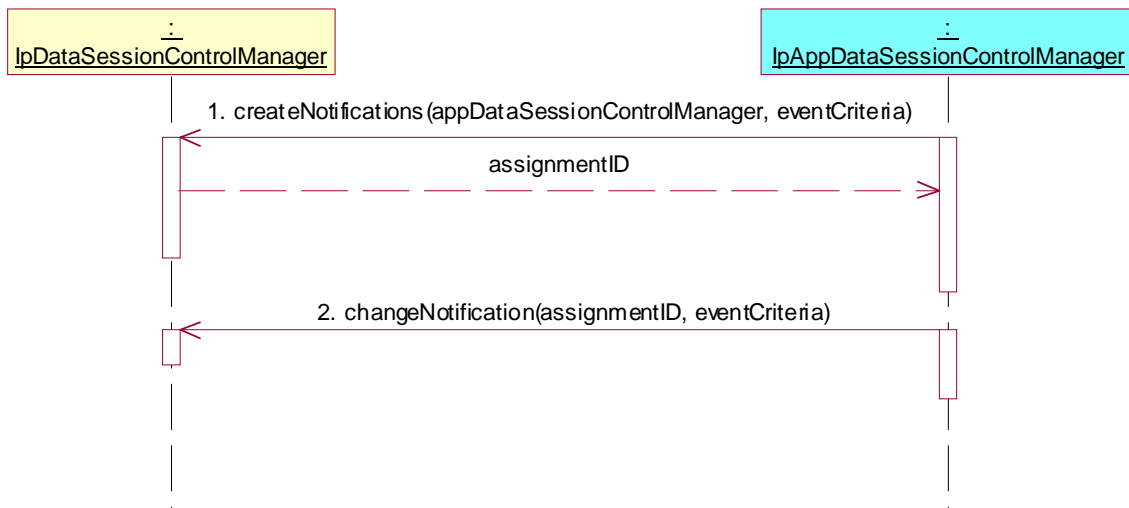Reference: ES 202 915-8 [1], clause 8.4.

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

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

                Instead of providing a valid value of its IpAppDataSessionControlManager interface reference in the
                createNotifications() method, the application is permitted to provide the
                IpAppDataSessionControlManager interface reference in a setCallback() method which it calls prior to
                invoking createNotifications().

Test Sequence:

1. Triggered Action: cause IUT to call **createNotifications()** method on the tester's (SCF's)
    IpDataSessionControlManager interface.
    Parameters: appDataSessionControlManager (non-NULL), eventCriteria

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

```
        : IpDataSessionControlManager                              : IpAppDataSessionControlManager
```

1. createNotifications(appDataSessionControlManager, eventCriteria)

assignmentID

2. changeNotification(assignmentID, eventCriteria)

Test DSC_APP_CM_05

Summary: **IpAppDataSessionControlManager**, disable data session notifications, successful.

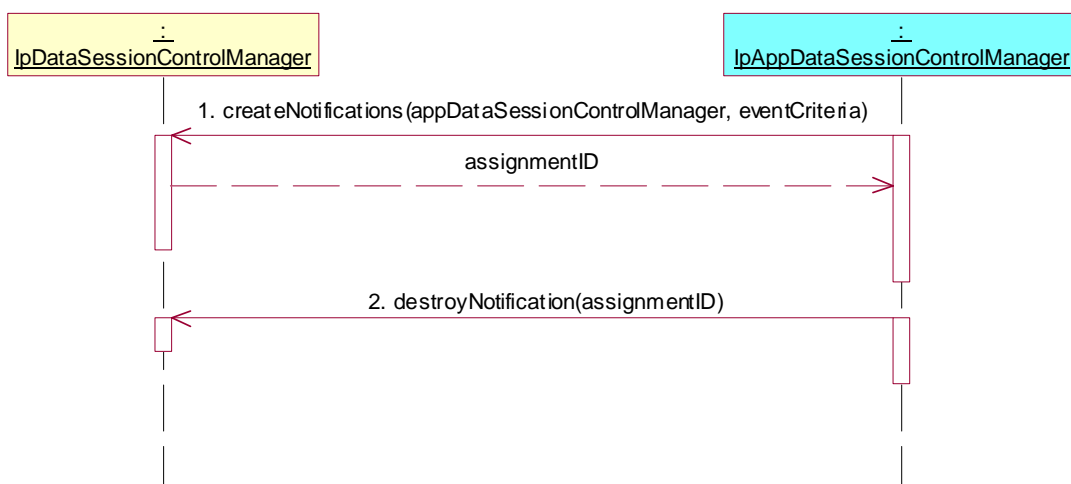Reference: ES 202 915-8 [1], clause 8.4.

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

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

             Instead of providing a valid value of its IpAppDataSessionControlManager interface reference in the
             createNotifications() method, the application is permitted to provide the
             IpAppDataSessionControlManager interface reference in a setCallback() method which it calls prior to
             invoking createNotifications().

Test Sequence:

1.   Triggered Action: cause IUT to call **createNotifications()** method on the tester's (SCF's)
     IpDataSessionControlManager interface.
     Parameters: appDataSessionControlManager (non-NULL), eventCriteria

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

```
        : IpDataSessionControlManager                              : IpAppDataSessionControlManager
```

1. createNotifications(appDataSessionControlManager, eventCriteria)

assignmentID

2. destroyNotification(assignmentID)

ETSI

Test DSC_APP_CM_06

Summary: **IpAppDataSessionControlManager**, query event criteria set, successful.
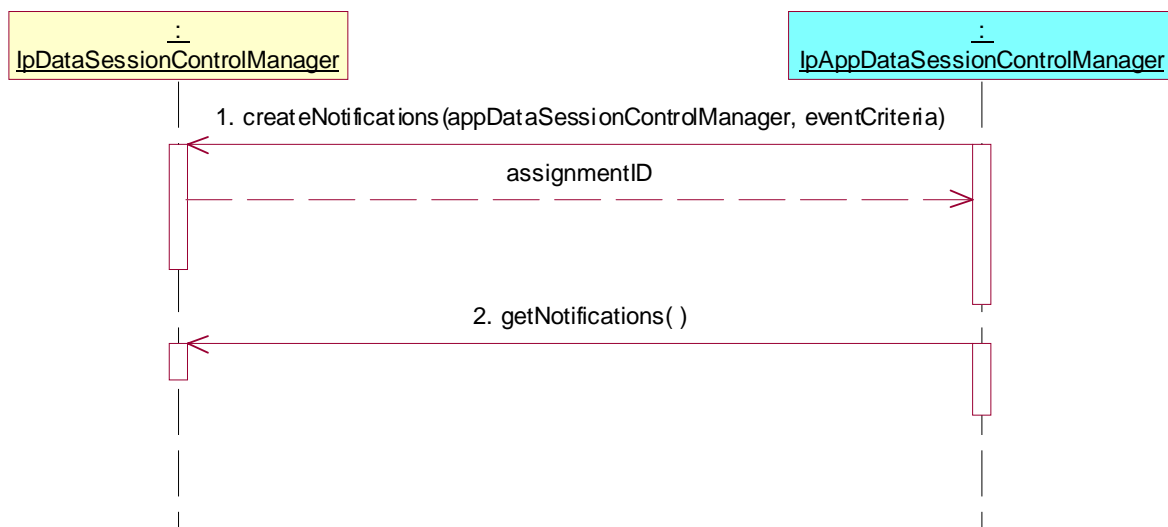
Reference: ES 202 915-8 [1], clause 8.4.

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

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

                Instead of providing a valid value of its IpAppDataSessionControlManager interface reference in the
                createNotifications() method, the application is permitted to provide the
                IpAppDataSessionControlManager interface reference in a setCallback() method which it calls prior to
                invoking createNotifications().

Test Sequence:

1. Triggered Action: cause IUT to call **createNotifications()** method on the tester's (SCF's)
   IpDataSessionControlManager interface.
   Parameters: appDataSessionControlManager (non-NULL), eventCriteria

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

Test DSC_APP_CM_07

Summary: **IpAppDataSessionControlManager**, query event criteria set, successful.

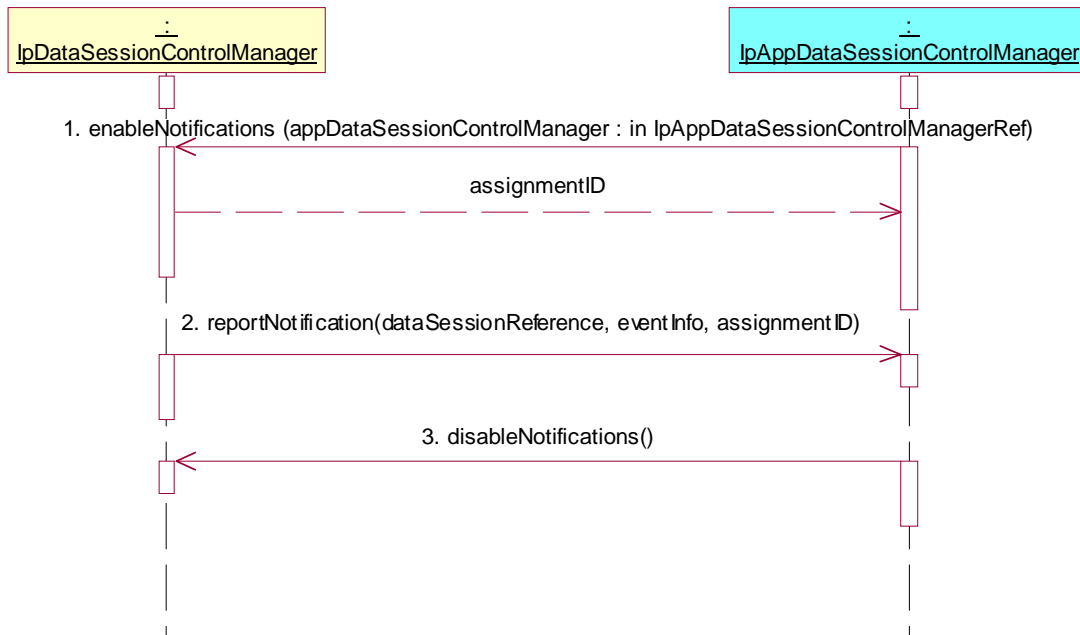Reference: ES 202 915-8 [1], clause 8.4.

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

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

                Instead of providing a valid value of its IpAppDataSessionControlManager interface reference in the
                createNotifications() method, the application is permitted to provide the
                IpAppDataSessionControlManager interface reference in a setCallback() method which it calls prior to
                invoking createNotifications().

Test Sequence:

1.   Triggered Action: cause IUT to call **enableNotifications ()** method on the tester's (application's)
      IpAppDataSessionControlManager interface
      Parameters: None

2.   Method call **reportNotification()**
      Parameters: dataSessionReference, eventInfo, assignmentID
      Check: no exception is returned

3.   Triggered Action: cause IUT to call **disableNotifications()** method on the tester's (application's)
      IpAppDataSessionControlManager interface
      Parameters: None

5.2.2.2   IpAppDataSession

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

Reference: ES 202 915-8 [1], clauses 9.1.

Precondition: **IpAppDataSession** interface implemented.

5.2.2.2.1   General tests, Active state

NOTE: Tests DSC_APP_DS_01 to 05 do not specify the values of DataSessionEventName field in the
             eventCriteria parameter of the createNotifications() 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.

Test DSC_APP_DS_01

Summary: **IpAppDataSession**, supervise data session.

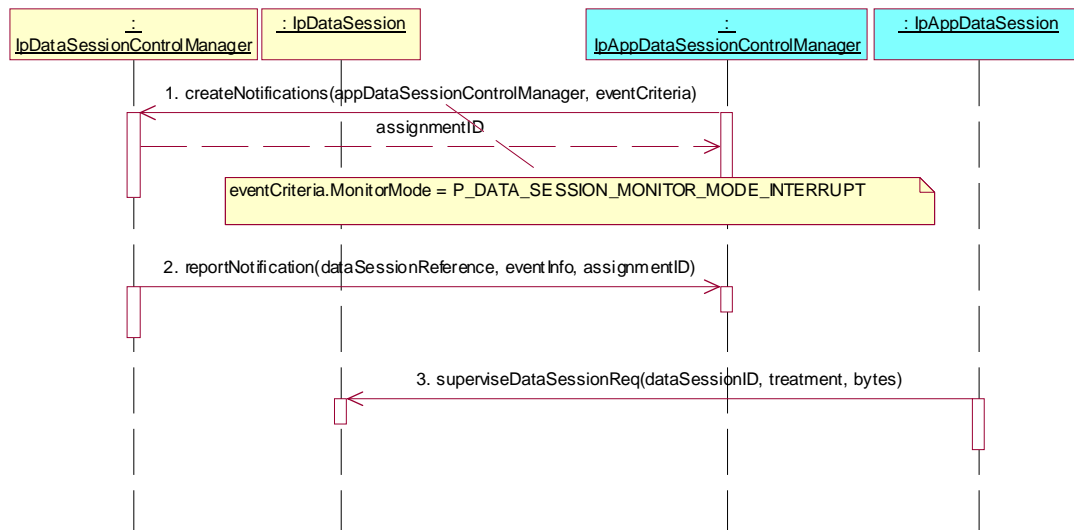Reference: ES 202 915-8 [1], clauses 8.1 and 8.3.

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

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

Instead of providing a valid value of its IpAppDataSessionControlManager interface reference in the createNotifications() method, the application is permitted to provide the IpAppDataSessionControlManager interface reference in a setCallback() method which it calls prior to invoking createNotifications().

Test Sequence:

1. Triggered Action: cause IUT to call **createNotifications()** method on the tester's (SCF's) IpDataSessionControlManager interface.
   Parameters: appDataSessionControlManager (non-NULL), eventCriteria
                   eventCriteria.MonitorMode = P_DATA_SESSION_MONITOR_MODE_INTERRUPT

2. Method call **reportNotification()**
   Parameters: dataSessionReference, eventInfo, assignmentID
   Check: valid value of IpAppDataSessionRef is returned

3. Triggered Action: cause IUT to call **superviseDataSessionReq()** method on the tester's (SCF's) IpDataSession interface.
   Parameters: dataSessionID, treatment, bytes

Test DSC_APP_DS_02

Summary: **IpAppDataSession**, indication of a data session fault.

Reference: ES 202 915-8 [1], clauses 8.1 and 8.3.
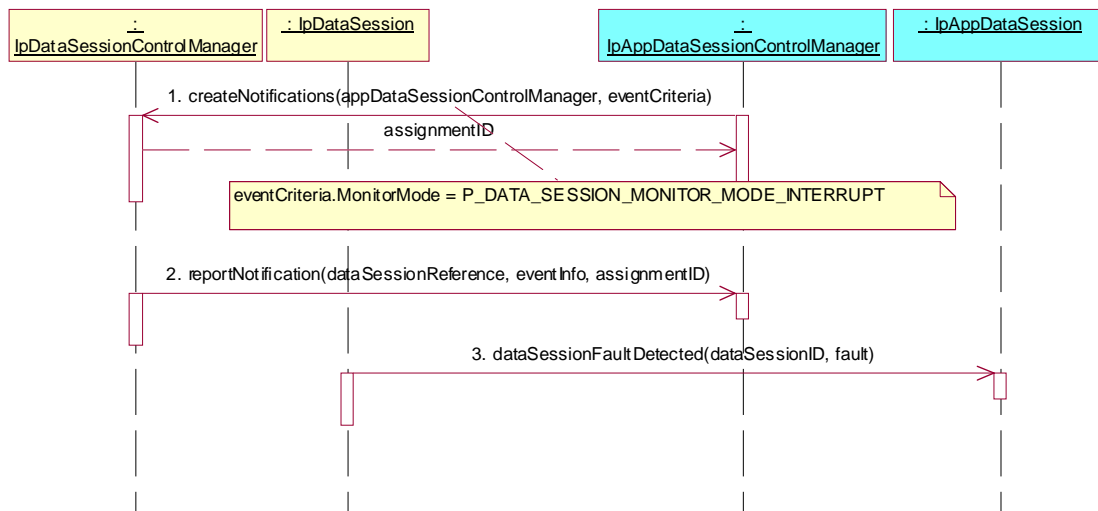
Precondition: **dataSessionFaultDetected()** implemented.

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

                Instead of providing a valid value of its IpAppDataSessionControlManager interface reference in the createNotifications() method, the application is permitted to provide the IpAppDataSessionControlManager interface reference in a setCallback() method which it calls prior to invoking createNotifications().

Test Sequence:

1. Triggered Action: cause IUT to call **createNotifications()** method on the tester's (SCF's)
   IpDataSessionControlManager interface.
   Parameters: appDataSessionControlManager (non-NULL), eventCriteria
                   eventCriteria.MonitorMode = P_DATA_SESSION_MONITOR_MODE_INTERRUPT

2. Method call **reportNotification()**
   Parameters: dataSessionReference, eventInfo, assignmentID
   Check: valid value of IpAppDataSessionRef is returned

3. Method call **dataSessionFaultDetected()**
   Parameters: dataSessionID, fault
   Check: no exception is returned

Test DSC_APP_DS_03

Summary: **IpAppDataSession**, determine charging information.

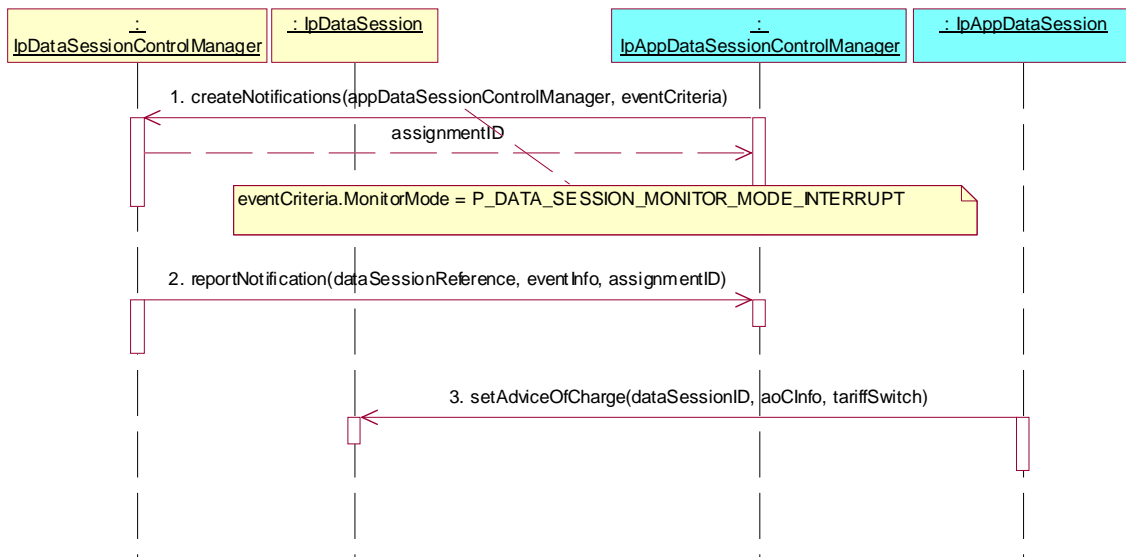Reference: ES 202 915-8 [1], clauses 8.1 and 8.3.

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

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

                Instead of providing a valid value of its IpAppDataSessionControlManager interface reference in the
                createNotifications() method, the application is permitted to provide the
                IpAppDataSessionControlManager interface reference in a setCallback() method which it calls prior to
                invoking createNotifications().

Test Sequence:

1. Triggered Action: cause IUT to call **createNotifications()** method on the tester's (SCF's)
   IpDataSessionControlManager interface.
   Parameters: appDataSessionControlManager (non-NULL), eventCriteria
                   eventCriteria.MonitorMode = P_DATA_SESSION_MONITOR_MODE_INTERRUPT

2. Method call **reportNotification()**
   Parameters: dataSessionReference, eventInfo, assignmentID
   Check: valid value of IpAppDataSessionRef is returned

3. Triggered Action: cause IUT to call **setAdviceOfCharge()** method on the tester's (SCF's) IpDataSession
   interface.
   Parameters: dataSessionID, aoCInfo, tariffSwitch

Test DSC_APP_DS_04

Summary: **IpAppDataSession**, include charging information.

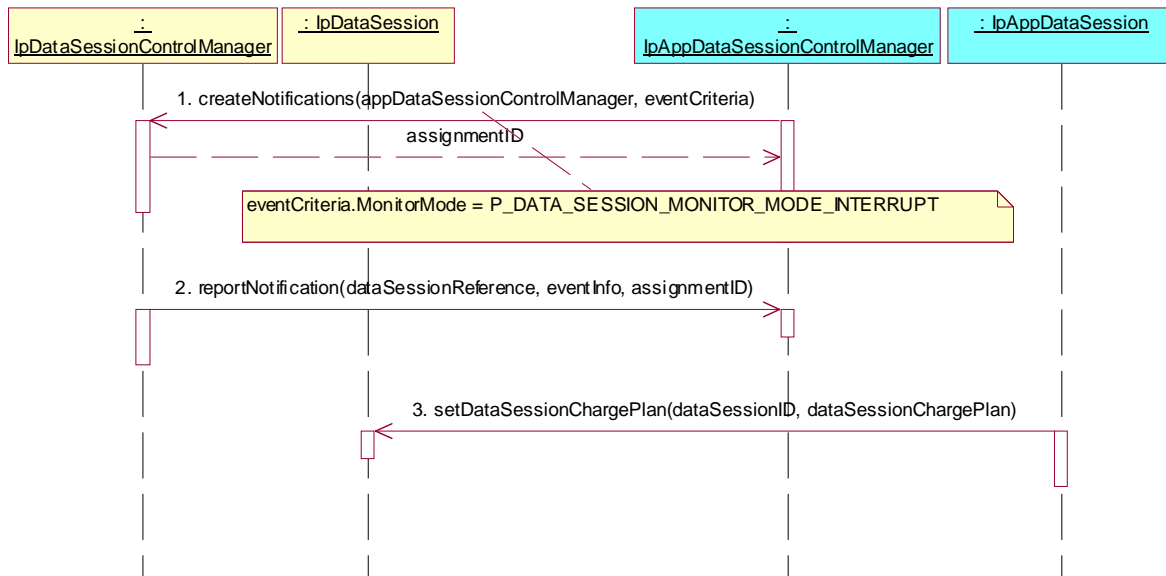Reference: ES 202 915-8 [1], clauses 8.1 and 8.3.

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

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

                Instead of providing a valid value of its IpAppDataSessionControlManager interface reference in the createNotifications() method, the application is permitted to provide the IpAppDataSessionControlManager interface reference in a setCallback() method which it calls prior to invoking createNotifications().

Test Sequence:

1.   Triggered Action: cause IUT to call **createNotifications()** method on the tester's (SCF's) IpDataSessionControlManager interface.
      Parameters: appDataSessionControlManager (non-NULL), eventCriteria
                       eventCriteria.MonitorMode = P_DATA_SESSION_MONITOR_MODE_INTERRUPT

2.   Method call **reportNotification()**
      Parameters: dataSessionReference, eventInfo, assignmentID
      Check: valid value of IpAppDataSessionRef is returned

3.   Triggered Action: cause IUT to call **setDataSessionChargePlan()** method on the tester's (SCF's) IpDataSession interface.
      Parameters: dataSessionID, dataSessionChargePlan

Test DSC_APP_DS_05

Summary: **IpAppDataSession**, release data session.

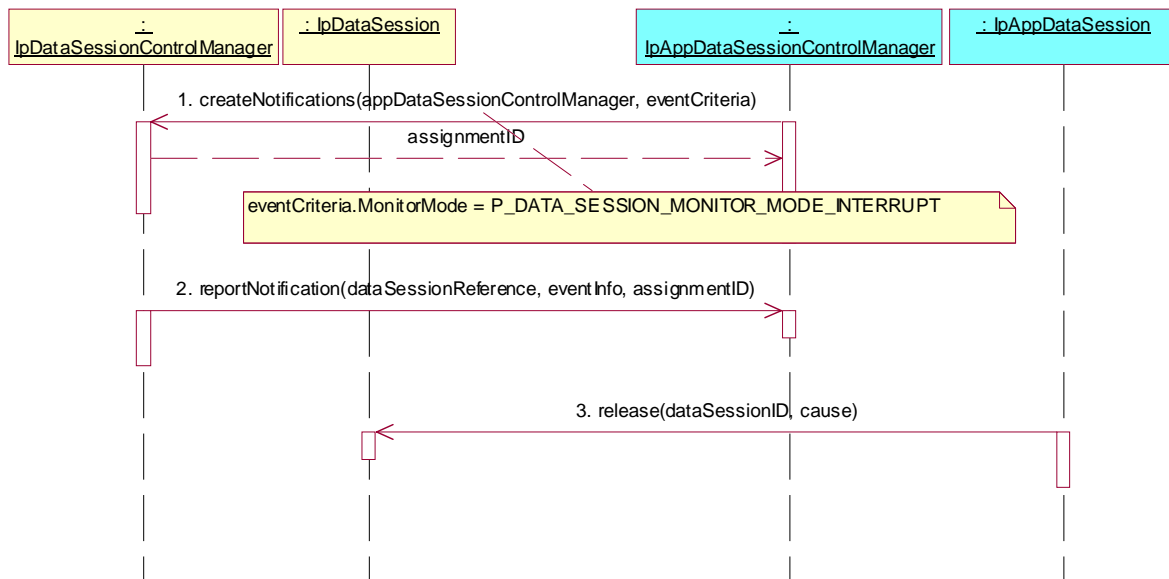Reference: ES 202 915-8 [1], clauses 8.1 and 8.3.

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

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

                Instead of providing a valid value of its IpAppDataSessionControlManager interface reference in the
                createNotifications() method, the application is permitted to provide the
                IpAppDataSessionControlManager interface reference in a setCallback() method which it calls prior to
                invoking createNotifications().

Test Sequence:

1. Triggered Action: cause IUT to call **createNotifications()** method on the tester's (SCF's)
    IpDataSessionControlManager interface.
    Parameters: appDataSessionControlManager (non-NULL), eventCriteria
                    eventCriteria.MonitorMode = P_DATA_SESSION_MONITOR_MODE_INTERRUPT

2. Method call **reportNotification()**
    Parameters: dataSessionReference, eventInfo, assignmentID
    Check: valid value of IpAppDataSessionRef is returned

3. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) IpDataSession interface.
    Parameters: dataSessionID, cause

5.2.2.2.2   Active state, Setup Sub-state

NOTE: It will be necessary to create some kind of 'PIXIT' items to check, if the sending of the distinct parameter values in the following tests is supported by the IUT.

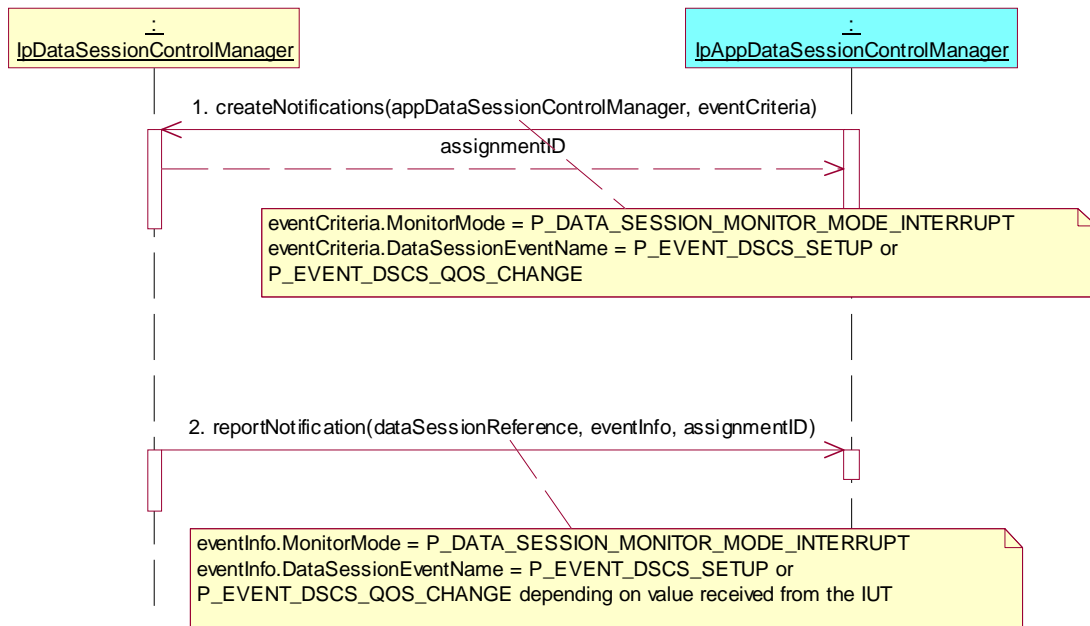**Preamble DSC_APP_DS_Active_Setup**

Reference: ES 202 915-8 [1], clauses 9.1.

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

Instead of providing a valid value of its IpAppDataSessionControlManager interface reference in the createNotifications() method, the application is permitted to provide the IpAppDataSessionControlManager interface reference in a setCallback() method which it calls prior to invoking createNotifications().

Preamble Sequence:

1.   Triggered Action: cause IUT to call **createNotifications()** method on the tester's (SCF's) IpDataSessionControlManager interface.
      Parameters: appDataSessionControlManager (non-NULL), eventCriteria
      Parameter values:
                eventCriteria.DataSessionEventName = P_EVENT_DSCS_SETUP or
                P_EVENT_DSCS_QOS_CHANGE
                eventCriteria.MonitorMode = P_DATA_SESSION_MONITOR_MODE_INTERRUPT

2.   Method call **reportNotification()**
      Parameters: dataSessionReference, eventInfo, assignmentID
      Parameter values:
                eventInfo.DataSessionEventName = P_EVENT_DSCS_SETUP or
                P_EVENT_DSCS_QOS_CHANGE depending on value received from the IUT
                eventInfo.MonitorMode = P_DATA_SESSION_MONITOR_MODE_INTERRUPT
      Check: valid value of IpAppDataSessionRef is returned

Test DSC_APP_DS_06

Summary: **IpAppDataSession**, connect interrupted data session, wait for connected, successful.
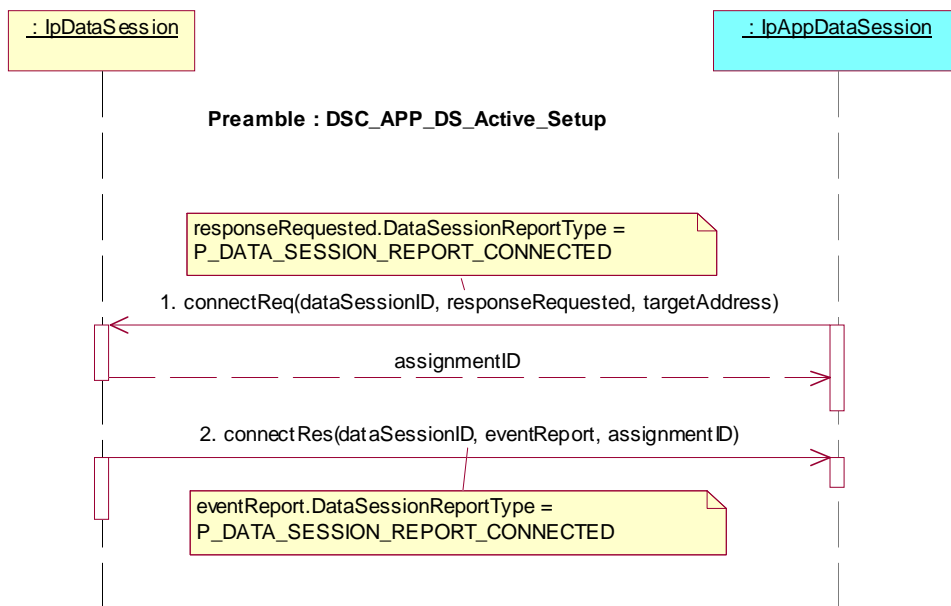
Reference: ES 202 915-8 [1], clauses 8.1 and 8.3.

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

Preamble: **DSC_APP_DS_Active_Setup.**

Test Sequence:

1.   Triggered Action: cause IUT to call **connectReq()** method on the tester's (SCF's) IpDataSession interface.
      Parameters: dataSessionID, responseRequested, targetAddress
      Parameter values:
                 responseRequested.DataSessionReportType = P_DATA_SESSION_REPORT_CONNECTED

2.   Method call **connectRes()**
      Parameters: dataSessionID, eventReport, assignmentID
      Parameter values:
                 eventReport.DataSessionReportType = P_DATA_SESSION_REPORT_CONNECTED
      Check: no exception is returned

Test DSC_APP_DS_07

Summary: **IpAppDataSession**, connect interrupted data session, wait for connected, unsuccessful.

Reference: ES 202 915-8 [1], clauses 8.1 and 8.3.

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

Preamble: **DSC_APP_DS_Active_Setup.**

Test Sequence:

1.   Triggered Action: cause IUT to call **connectReq()** method on the tester's (SCF's) IpDataSession interface.
      Parameters: dataSessionID, responseRequested, targetAddress

2.   Method call **connectErr()**
      Parameters: dataSessionID, errorIndication, assignmentID
      Check: no exception is returned

Test DSC_APP_DS_08

Summary: **IpAppDataSession**, connect interrupted data session, wait for disconnected, successful.
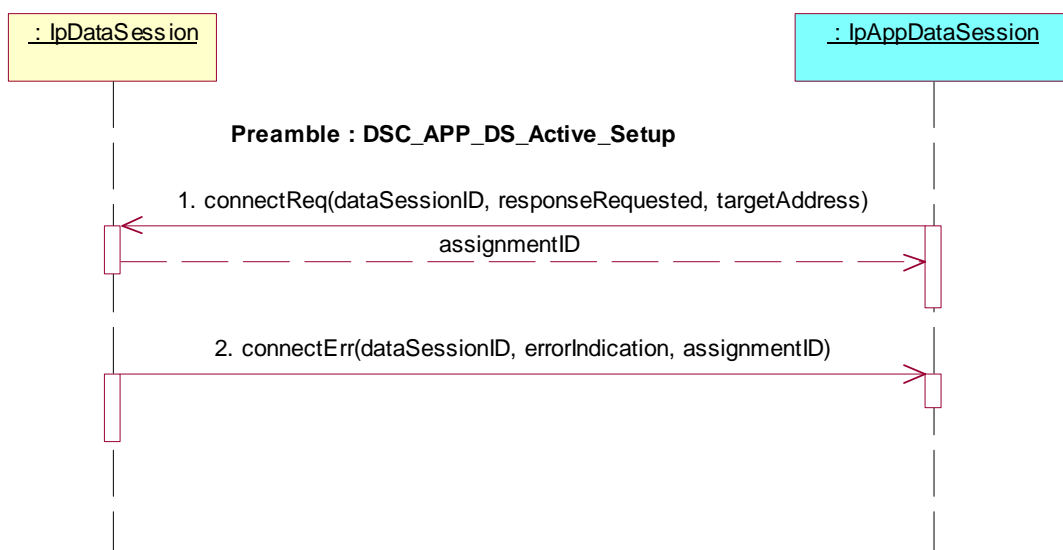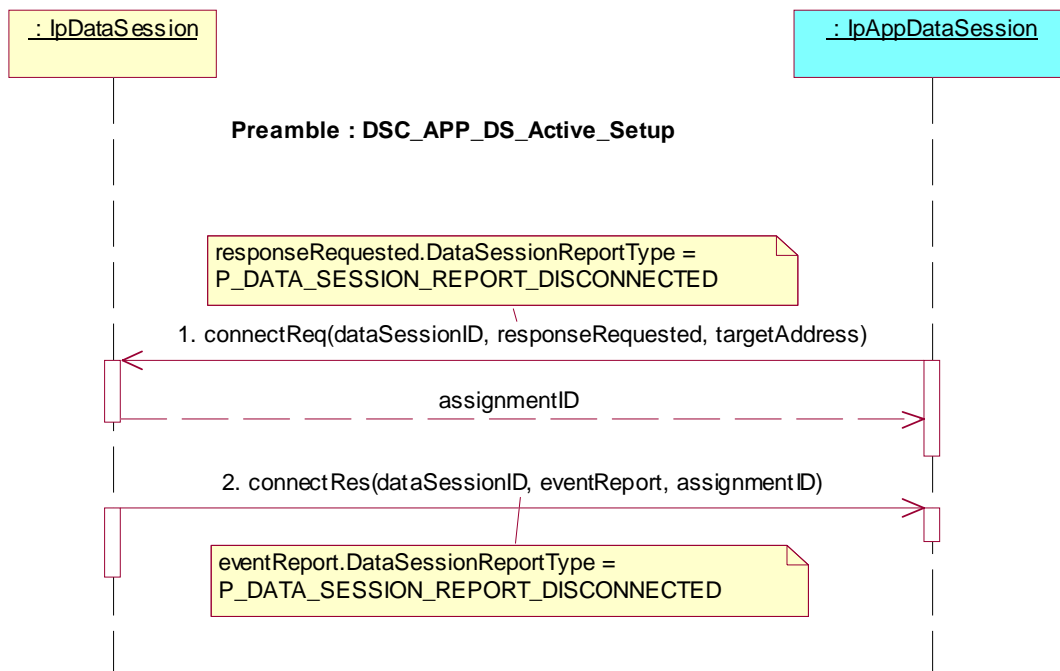
Reference: ES 202 915-8 [1], clauses 8.1 and 8.3.

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

Preamble: **DSC_APP_DS_Active_Setup.**

Test Sequence:

1. Triggered Action: cause IUT to call **connectReq()** method on the tester's (SCF's) IpDataSession interface.
    Parameters: dataSessionID, responseRequested, targetAddress
    Parameter values:
            responseRequested.DataSessionReportType = P_DATA_SESSION_REPORT_DISCONNECT

2. Method call **connectRes()**
    Parameters: dataSessionID, eventReport, assignmentID
    Parameter values:
            eventReport.DataSessionReportType = P_DATA_SESSION_REPORT_DISCONNECT
    Check: no exception is returned

Test DSC_APP_DS_09

Summary: **IpAppDataSession**, continue interrupted data session.

Reference: ES 202 915-8 [1], clauses 8.1 and 8.3.

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

Preamble: **DSC_APP_DS_Active_Setup.**

Test Sequence:

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

```
┌─────────────────┐                          ┌──────────────────┐
│ : IpDataSession │                          │ : IpAppDataSession│
└─────────────────┘                          └──────────────────┘
        │                                             │
        │       Preamble : DSC_APP_DS_Active_Setup    │
        │                                             │
        │         1. continueProcessing(dataSessionID)│
        │◄────────────────────────────────────────────│
        ▯                                             ▯
        │                                             │
        │                                             │
        │                                             │
```

5.2.2.2.2   Active state, Established Sub-state

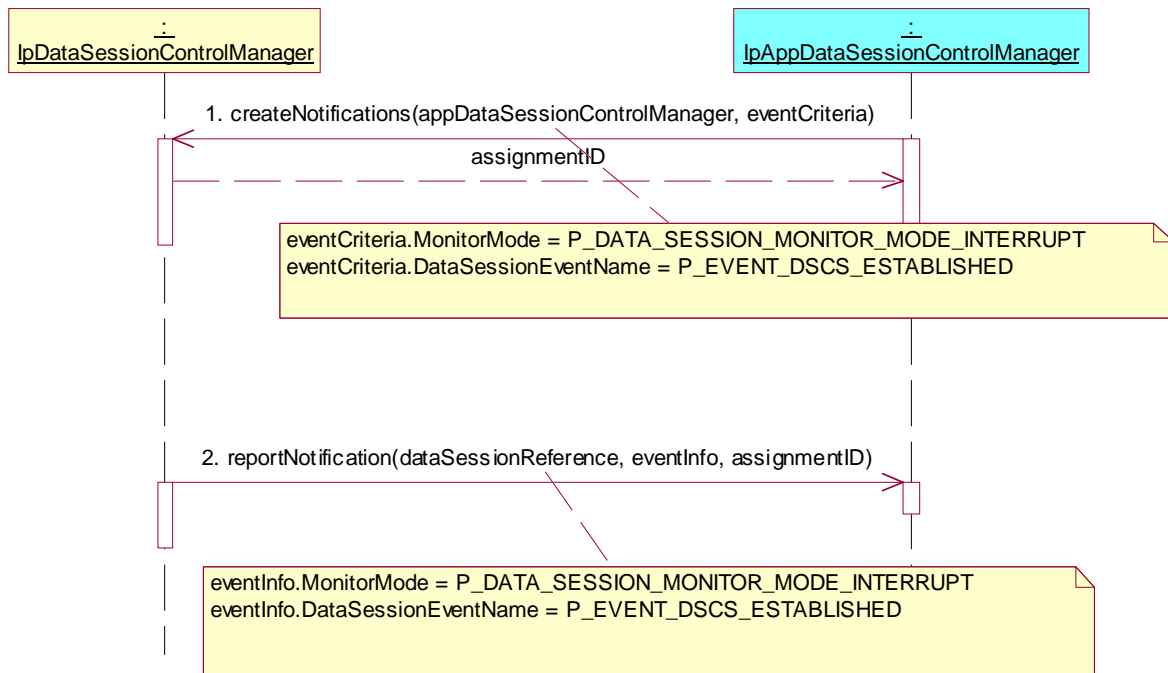**Preamble DSC_APP_DS_Active_Established**

Reference: ES 202 915-8 [1], clauses 9.1.

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

Instead of providing a valid value of its IpAppDataSessionControlManager interface reference in the createNotifications() method, the application is permitted to provide the IpAppDataSessionControlManager interface reference in a setCallback() method which it calls prior to invoking createNotifications().

Preamble Sequence:

1.   Triggered Action: cause IUT to call **createNotifications()** method on the tester's (SCF's) IpDataSessionControlManager interface.
       Parameters: appDataSessionControlManager (non-NULL), eventCriteria
       Parameter values:
                 eventCriteria.DataSessionEventName = P_EVENT_DSCS_ESTABLISHED
                 eventCriteria.MonitorMode = P_DATA_SESSION_MONITOR_MODE_INTERRUPT

2.   Method call **reportNotification()**
       Parameters: dataSessionReference, eventInfo, assignmentID
       Parameter values:
                 eventInfo.DataSessionEventName = P_EVENT_DSCS_ESTABLISHED
                 eventInfo.MonitorMode = P_DATA_SESSION_MONITOR_MODE_INTERRUPT
       Check: valid value of IpAppDataSessionRef is returned

Test DSC_APP_DS_10

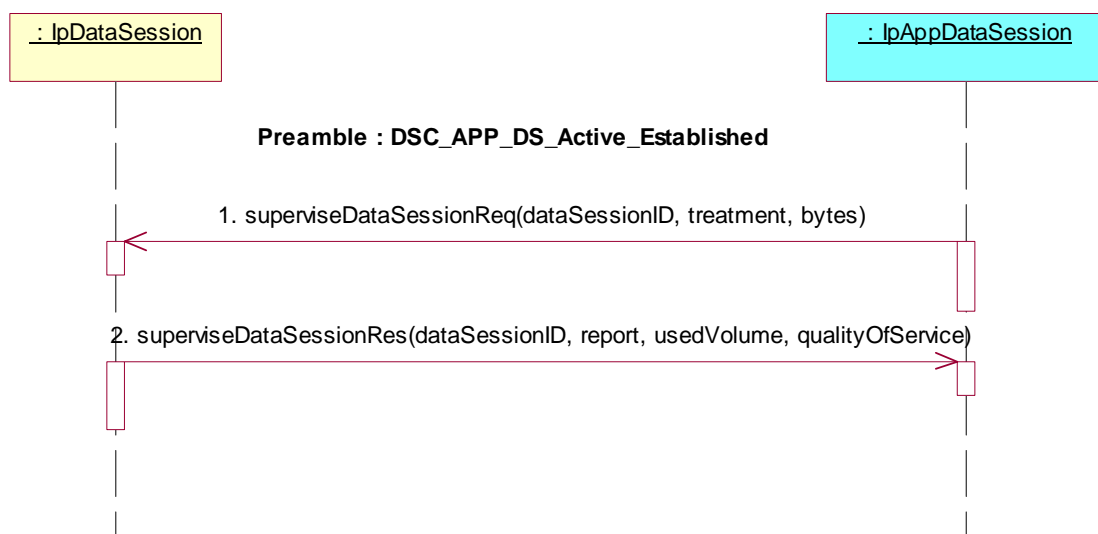Summary: **IpAppDataSession**, supervise data session, successful.

Reference: ES 202 915-8 [1], clauses 8.1 and 8.3.

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

Preamble: **DSC_APP_DS_Active_Established.**

Test Sequence:

1.   Triggered Action: cause IUT to call **superviseDataSessionReq()** method on the tester's (SCF's) IpDataSession
     interface.
     Parameters: dataSessionID, treatment, bytes

2.   Method call **superviseDataSessionRes()**
     Parameters: dataSessionID, report, usedVolume, qualityOfService
     Check: no exception returned

Test DSC_APP_DS_11

Summary: **IpAppDataSession**, supervise data session, unsuccessful.

Reference: ES 202 915-8 [1], clauses 8.1 and 8.3.

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

Preamble: **DSC_APP_DS_Active_Established.**

Test Sequence:

1. Triggered Action: cause IUT to call **superviseDataSessionReq()** method on the tester's (SCF's) IpDataSession
    interface.
    Parameters: dataSessionID, treatment, bytes

2. Method call **superviseDataSessionErr()**
    Parameters: dataSessionID, erroIndication
    Check: no exception returned


```
    : IpDataSession                                                          : IpAppDataSession

                       Preamble : DSC_APP_DS_Active_Established


                  1. superviseDataSessionReq(dataSessionID, treatment, bytes)


                  2. superviseDataSessionErr(dataSessionID, errorIndication)
```

5.2.2.2.3   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 DSC_APP_DS_12

Summary: **IpAppDataSession**, de-assign data session that has been released by the network for reasons of fault

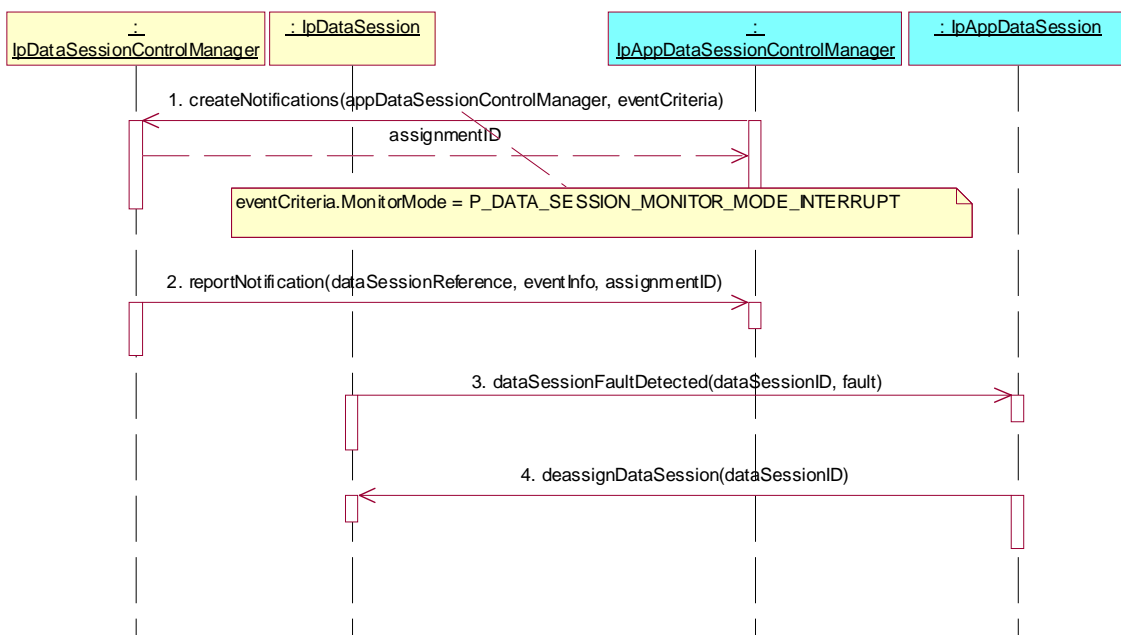Reference: ES 202 915-8 [1], clauses 8.1, 8.3 and 9.1

Precondition: **dataSessionFaultDetected()** implemented, IUT capable of invoking **deassignDataSession()**

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

               Instead of providing a valid value of its IpAppDataSessionControlManager interface reference in the
               createNotifications() method, the application is permitted to provide the
               IpAppDataSessionControlManager interface reference in a setCallback() method which it calls prior to
               invoking createNotifications().

Test Sequence:

1. Triggered Action: cause IUT to call **createNotifications()** method on the tester's (SCF's)
   IpDataSessionControlManager interface.
   Parameters: appDataSessionControlManager (non-NULL), eventCriteria

2. Method call **reportNotification()**
   Parameters: dataSessionReference, eventInfo, assignmentID
   Check: valid value of IpAppDataSessionRef is returned

3. Method call **dataSessionFaultDetected()**
   Parameters: dataSessionID, fault
   Check: no exception is returned

4. Triggered Action: cause IUT to call **deassignDataSession()** method on the tester's (SCF's) IpDataSession
   interface.
   Parameters: dataSessionID

Test DSC_APP_DS_13

Summary: **IpAppDataSession**, de-assign network released data session **after** receipt of supervision result.

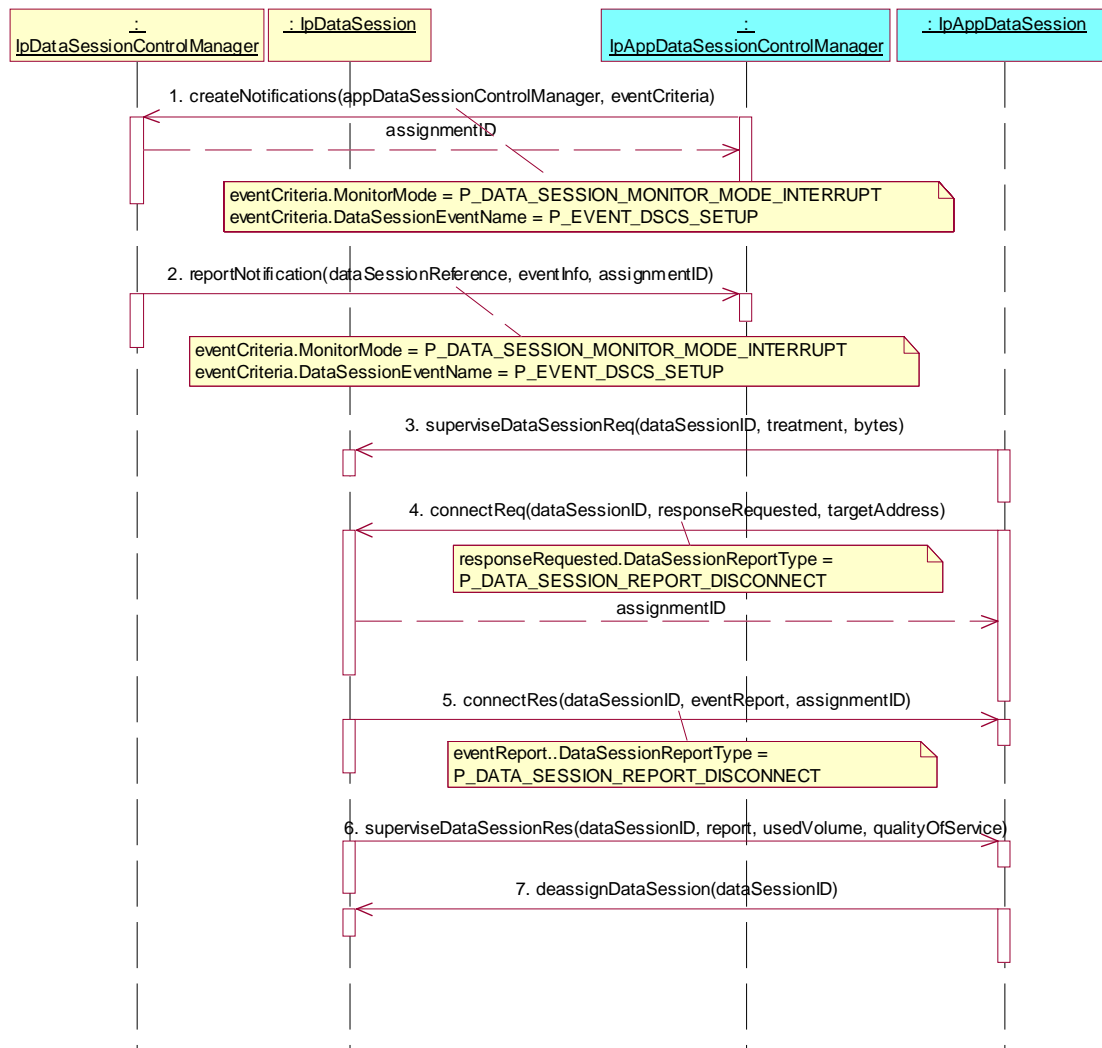Reference: ES 202 915-8 [1], clauses 8.1, 8.3 and 9.1.

Precondition: IUT capable of invoking **connectReq()**, **superviseDataSessionReq()** and **deassignDataSession().**

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

              Instead of providing a valid value of its IpAppDataSessionControlManager interface reference in the
              createNotifications() method, the application is permitted to provide the
              IpAppDataSessionControlManager interface reference in a setCallback() method which it calls prior to
              invoking createNotifications().

Preamble Sequence:

1. Triggered Action: cause IUT to call **createNotifications()** method on the tester's (SCF's)
   IpDataSessionControlManager interface.
   Parameters: appDataSessionControlManager (non-NULL), eventCriteria
   Parameter values:
      eventCriteria.DataSessionEventName = P_EVENT_DSCS_SETUP
      eventCriteria.MonitorMode = P_DATA_SESSION_MONITOR_MODE_INTERRUPT

2. Method call **reportNotification()**
   Parameters: dataSessionReference, eventInfo, assignmentID
   Parameter values:
      eventInfo.DataSessionEventName = P_EVENT_DSCS_SETUP
      eventInfo.MonitorMode = P_DATA_SESSION_MONITOR_MODE_INTERRUPT
   Check: valid value of IpAppDataSessionRef is returned

3. Triggered Action: cause IUT to call **superviseDataSessionReq()** method on the tester's (SCF's) IpDataSession
   interface.
   Parameters: dataSessionID, treatment, bytes

4. Triggered Action: cause IUT to call **connectReq()** method on the tester's (SCF's) IpDataSession interface.
   Parameters: dataSessionID, responseRequested, targetAddress
   Parameter values:
   Parameter values:
      responseRequested.DataSessionReportType = P_DATA_SESSION_REPORT_DISCONNECT

5. Method call **connectRes()**
   Parameters: dataSessionID, eventReport, assignmentID
   Parameter values:
      eventReport.DataSessionReportType = P_DATA_SESSION_REPORT_DISCONNECT
   Check: no exception is returned

6. Method call **superviseDataSessionRes()**
   Parameters: dataSessionID, report, usedVolume, qualityOfService
   Check: no exception returned

7. Triggered Action: cause IUT to call **deassignDataSession()** method on the tester's (SCF's) IpDataSession
   interface.
   Parameters: dataSessionID

5.2.2.2.4   Application Released state

Test DSC_APP_DS_14

Summary: **IpAppDataSession**, accept receipt of supervision result **after** release of data session by application.

Reference: ES 202 915-8 [1], clauses 8.1, 8.3 and 9.1.

Precondition: IUT capable of invoking **connectReq()**, **superviseDataSessionReq()** and **release().**
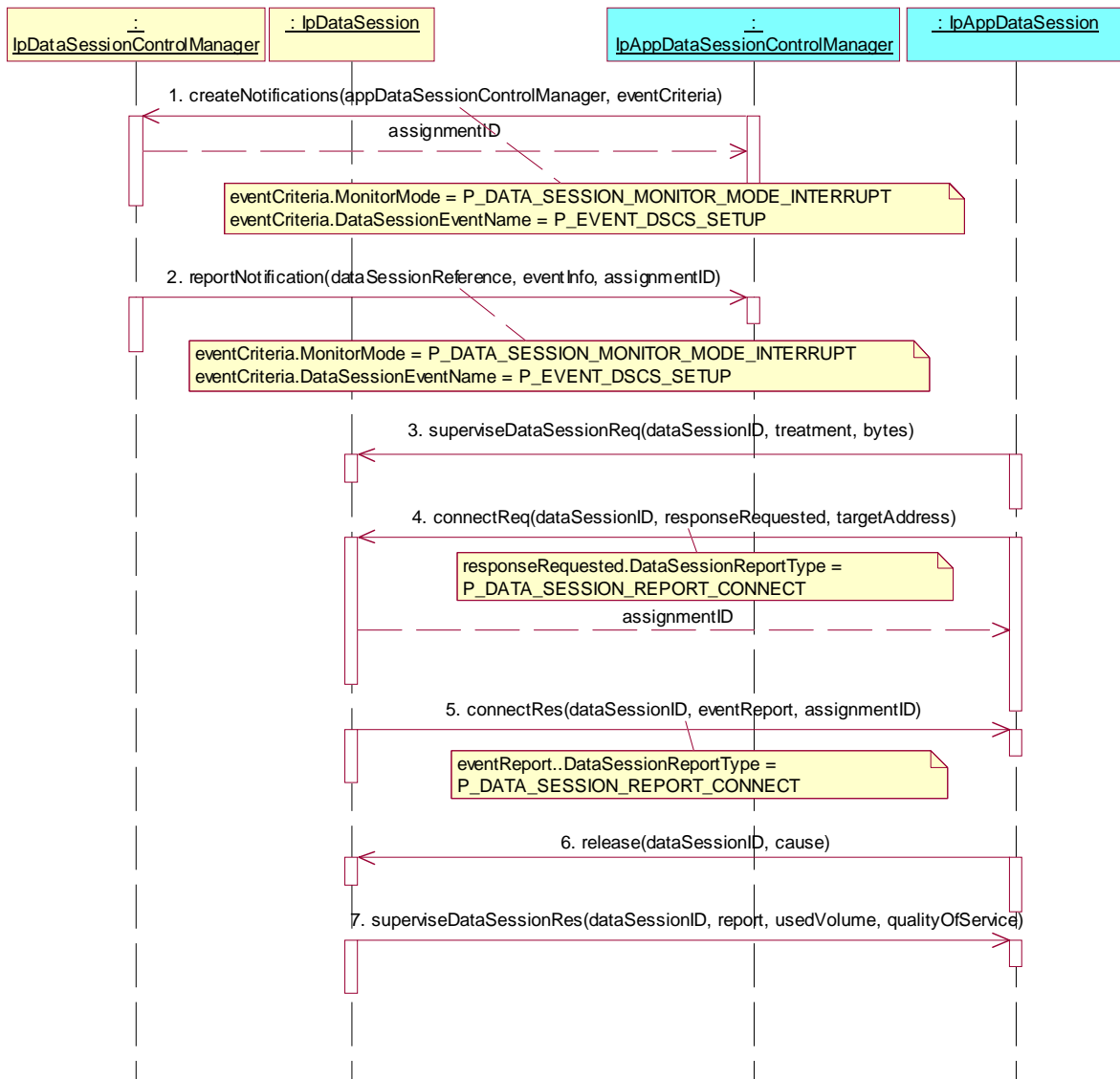
Preamble: Registration of the IUT (application) and the tester (Data Session Control SCF) to the framework. The IUT must have obtained a reference to an instance of the IpDataSessionControlManager interface through selecting that service and signing the required service agreement.

Instead of providing a valid value of its IpAppDataSessionControlManager interface reference in the createNotifications() method, the application is permitted to provide the IpAppDataSessionControlManager interface reference in a setCallback() method which it calls prior to invoking createNotifications().

Preamble Sequence:

1.   Triggered Action: cause IUT to call **createNotifications()** method on the tester's (SCF's) IpDataSessionControlManager interface.
Parameters: appDataSessionControlManager (non-NULL), eventCriteria
Parameter values:
          eventCriteria.DataSessionEventName = P_EVENT_DSCS_SETUP
          eventCriteria.MonitorMode = P_DATA_SESSION_MONITOR_MODE_INTERRUPT

2. Method call **reportNotification()**
   Parameters: dataSessionReference, eventInfo, assignmentID
   Parameter values:
           eventInfo.DataSessionEventName = P_EVENT_DSCS_SETUP
           eventInfo.MonitorMode = P_DATA_SESSION_MONITOR_MODE_INTERRUPT
   Check: valid value of IpAppDataSessionRef is returned

3. Triggered Action: cause IUT to call **superviseDataSessionReq()** method on the tester's (SCF's) IpDataSession interface.
   Parameters: dataSessionID, treatment, bytes

4. Triggered Action: cause IUT to call **connectReq()** method on the tester's (SCF's) IpDataSession interface.
   Parameters: dataSessionID, responseRequested, targetAddress
   Parameter values:
   Parameter values:
           responseRequested.DataSessionReportType = P_DATA_SESSION_REPORT_CONNECT

5. Method call **connectRes()**
   Parameters: dataSessionID, eventReport, assignmentID
   Parameter values:
           eventReport.DataSessionReportType = P_DATA_SESSION_REPORT_CONNECT
   Check: no exception is returned

6. Triggered Action: cause IUT to call **release()** method on the tester's (SCF's) IpDataSession interface.
   Parameters: dataSessionID, cause

7. Method call **superviseDataSessionRes()**
   Parameters: dataSessionID, report, usedVolume, qualityOfService
   Check: no exception returned

History

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