Final draft ETSI ES 202 196-10 V1.2.1 (2004-12)

ETSI Standard

**Open Service Access (OSA);
Application Programming Interface (API);
Test Suite Structure and Test Purposes (TSS&TP);
Part 10: Connectivity manager SCF
(Parlay 3)**

Reference

RES/TISPAN-06005-10-OSA

Keywords

API, OSA, TSS&TP

Contents

Intellectual Property Rights

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 10 of a multi-part deliverable covering Open Service Access (OSA); Application Programming Interface (API); Test Suite Structure and Test Purposes (TSS&TP) for Parlay 3, as identified below:

Part 1: "Overview";

Part 2: "Common data definitions";

Part 3: "Framework";

Part 4: "Call control SCF";

Part 5: "User interaction SCF";

Part 6: "Mobility SCF";

Part 7: "Terminal capabilities SCF";

Part 8: "Data session control SCF";

Part 9: "Generic messaging SCF";

Part 10: "Connectivity manager SCF";

Part 11: "Account management SCF";

Part 12: "Charging SCF".

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 Connectivity Manager SCF of the Application Programming Interface (API) for Open Service Access (OSA) defined in ES 201 915-10 [1] in compliance with the relevant requirements, and in accordance with the relevant guidance given in ISO/IEC 9646-2 [4] and ETS 300 406 [5].

2 References

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

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

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

[1] ETSI ES 201 915-10: "Open Service Access (OSA); Application Programming Interface (API); Part 10: Connectivity Manager SCF (Parlay 3)".

[2] ETSI ES 202 170: "Open Service Access (OSA); Application Programming Interface (API); Implementation Conformance Statement (ICS) proforma specification (Parlay 3)".

[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".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in ES 201 915-10 [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].

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

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

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

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

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

Lower Tester (LT): 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	Connectivity Manager
ICS	Implementation Conformance Statement
IUT	Implementation Under Test
IXIT	Implementation eXtra Information for Testing
LT	Lower Tester
OSA	Open Service Access
TP	Test Purpose
TSS	Test Suite Structure

4 Test Suite Structure (TSS)

- Connectivity Manager.

5 Test Purposes (TP)

5.1 Introduction

For each test requirement a TP is defined.

5.1.1 TP naming convention

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

Table 1: TP identifier naming convention scheme

Identifier: <suite_id>_<group>_<nnn>		
<suite_id>	= SCG name:	"CM" for **C**onnectivity **M**anager part of Connectivity Manager 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 201 915-10 [1].

5.1.3 Test strategy

As the base standard ES 201 915-10 [1] contains no explicit requirements for testing, the TPs were generated as a result of an analysis of the base standard and the ICS specification ES 202 170 [2].

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

5.2 TPs for the Connectivity Manager SCF

All ICS items referred to in this clause are as specified in ES 202 170 [2] unless indicated otherwise by another numbered reference.

All parameters specified in method calls are valid unless specified.

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

5.2.1 Connectivity Manager

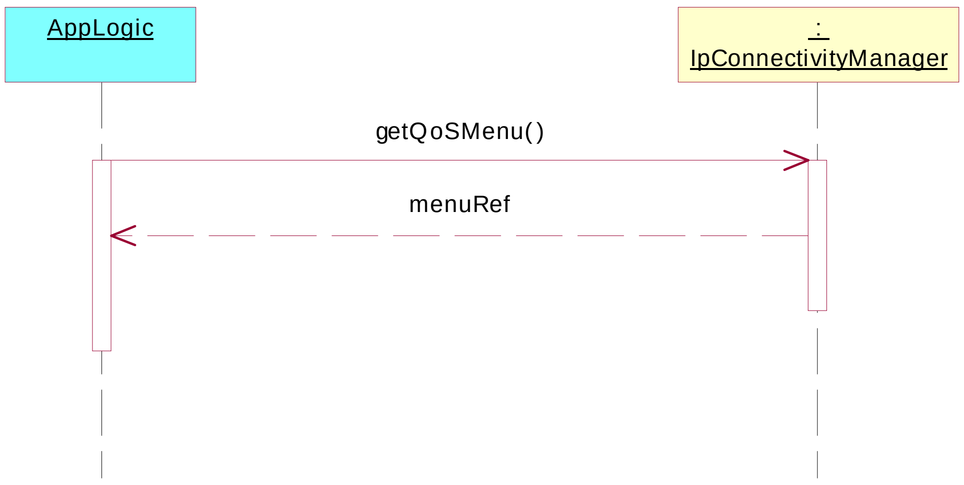
Test CM_01

Summary: getQoSMenu, successful

Reference: ES 201 915-10 [1], clause 8.1

Test Sequence:

1. Method call **getQoSMenu()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned



Test CM_02

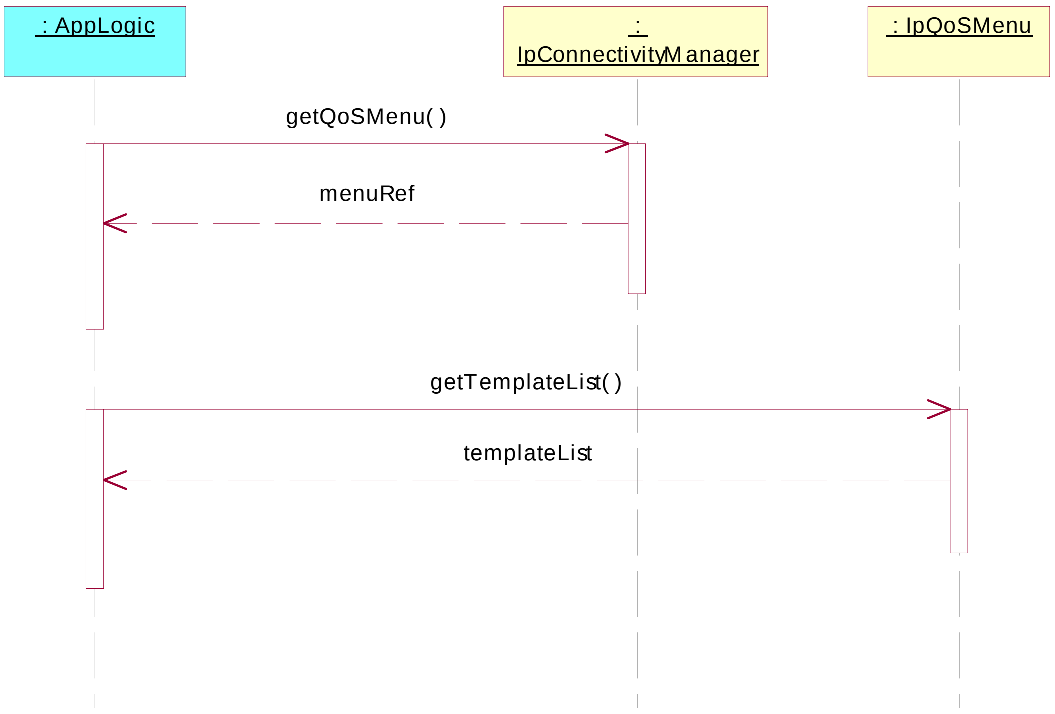
Summary: getTemplateList, successful

Reference: ES 201 915-10 [1], clauses 8.1 and 8.4

Test Sequence:

1. Method call **getQoSMenu()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getTemplateList()** on IpQoSMenu
   Parameters: None
   Check: valid value of TpStringList is returned



Test CM_03

Summary: getTemplate, successful

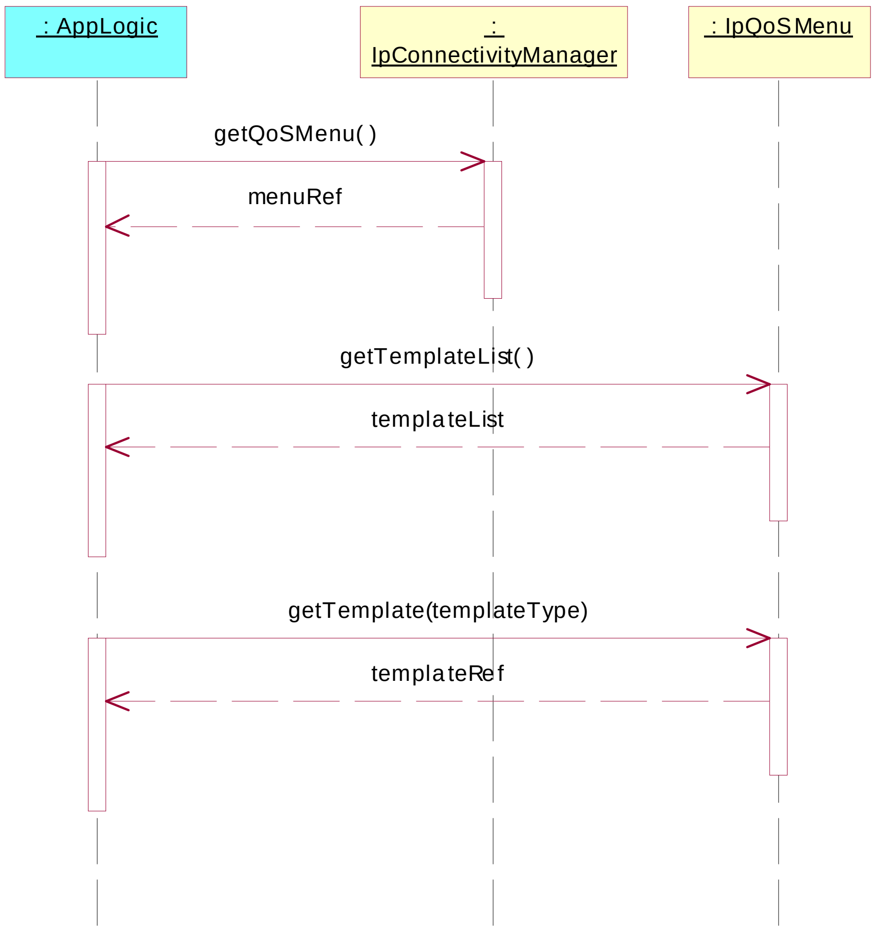
Reference: ES 201 915-10 [1], clauses 8.1 and 8.4

Test Sequence:

1. Method call **getQoSMenu()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getTemplateList()** on IpQoSMenu
   Parameters: None
   Check: valid value of TpStringList is returned

3. Method call **getTemplate()** on IpQoSMenu
   Parameters: Valid value of templateType, which is an item of the TpStringList, returned in 2.
   Check: valid value of IpInterfaceRef is returned



Test CM_04

Summary: getTemplateType, successful

Reference: ES 201 915-10 [1], clauses 8.1, 8.4 and 8.5

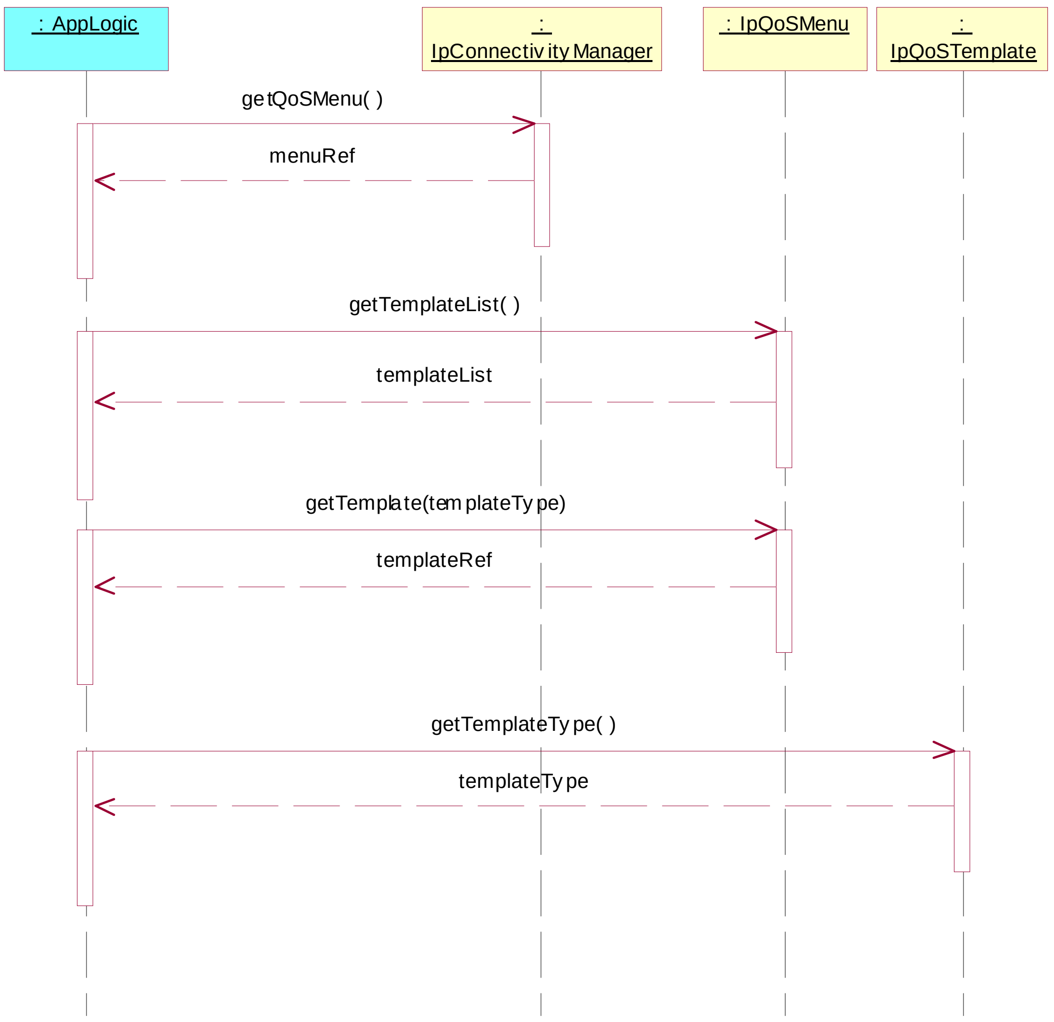
Test Sequence:

1. Method call **getQoSMenu()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getTemplateList()** on IpQoSMenu
   Parameters: None
   Check: valid value of TpStringList is returned

3. Method call **getTemplate()** on IpQoSMenu
   Parameters: Valid value of templateType, which is an item of the TpStringList, returned in 2.
   Check: valid value of IpInterfaceRef is returned

4. Method call **getTemplateType()** on IpQoSTemplate
   Parameters: None
   Check: valid value of TpString is returned



Test CM_05

Summary: getDescription, successful

Reference: ES 201 915-10 [1], clauses 8.1, 8.4 and 8.5

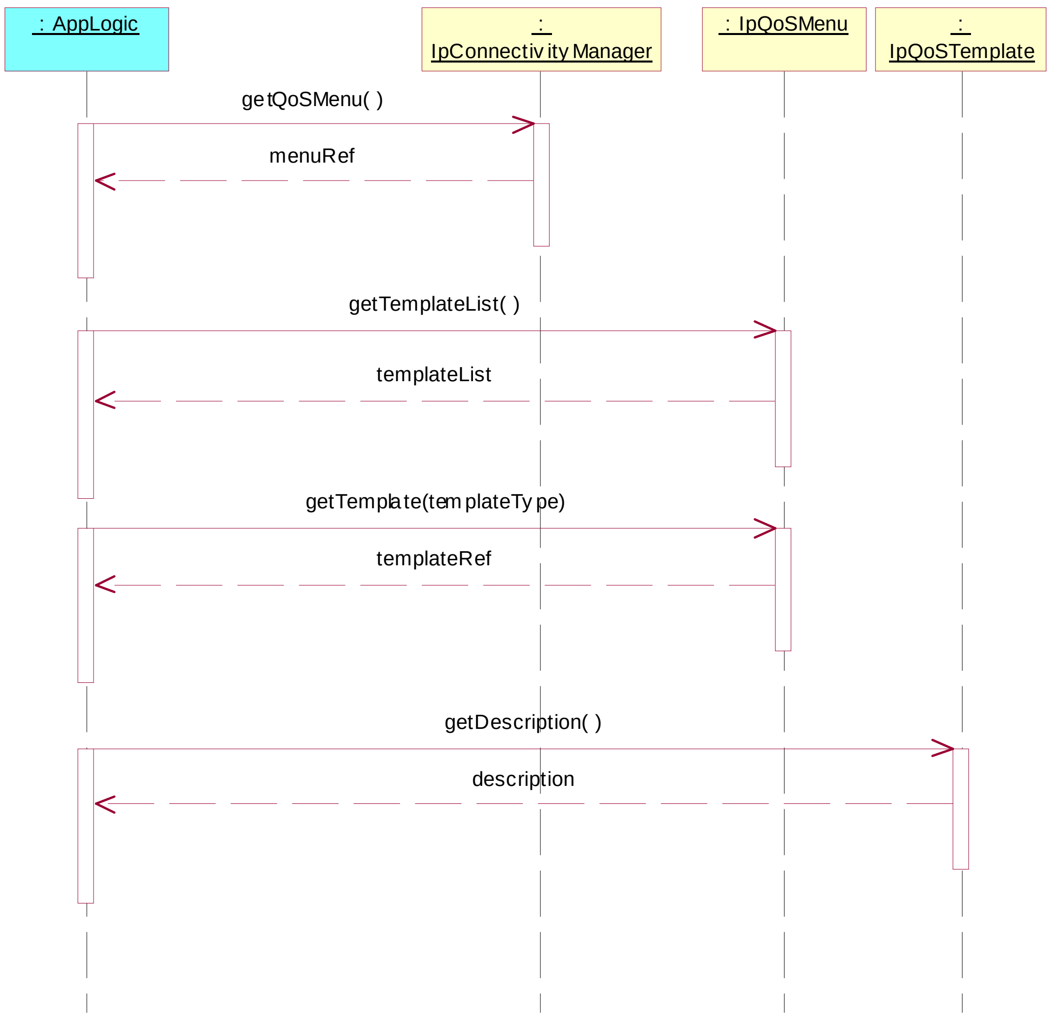
Test Sequence:

1. Method call **getQoSMenu()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getTemplateList()** on IpQoSMenu
   Parameters: None
   Check: valid value of TpStringList is returned

3. Method call **getTemplate()** on IpQoSMenu
   Parameters: Valid value of templateType, which is an item of the TpStringList, returned in 2.
   Check: valid value of IpInterfaceRef is returned

4. Method call **getDescription()** on IpQoSTemplate
   Parameters: None
   Check: valid value of TpString is returned



Test CM_06

Summary: setSlaID, successful

Reference: ES 201 915-10 [1], clauses 8.1, 8.4 and 8.5

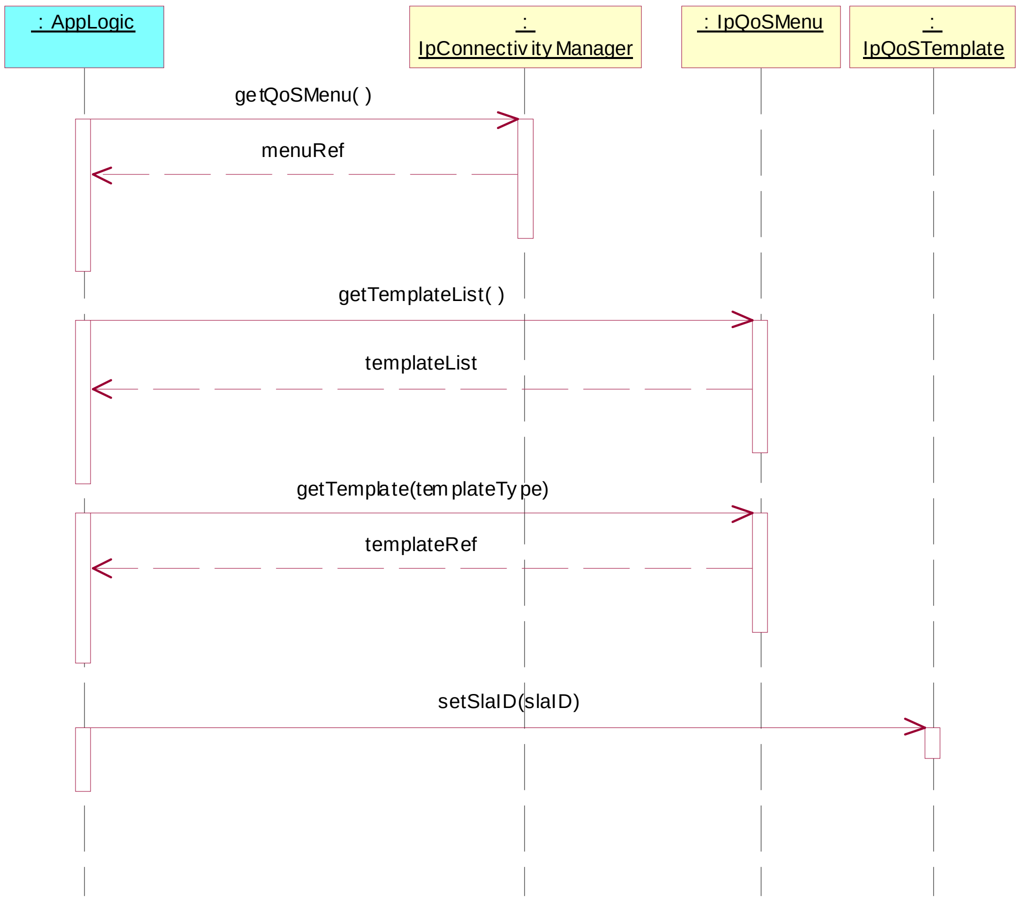
Test Sequence:

1. Method call **getQoSMenu()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getTemplateList()** on IpQoSMenu
   Parameters: None
   Check: valid value of TpStringList is returned

3. Method call **getTemplate()** on IpQoSMenu
   Parameters: Valid value of templateType, which is an item of the TpStringList, returned in 2.
   Check: valid value of IpInterfaceRef is returned

4. Method call **setSlaID()** on IpQoSTemplate
   Parameters: Valid SlaID
   Check: no exception is returned



Test CM_07

Summary: setSlaID, P_ILLEGAL_SLA_ID

Reference: ES 201 915-10 [1], clauses 8.1, 8.4 and 8.5

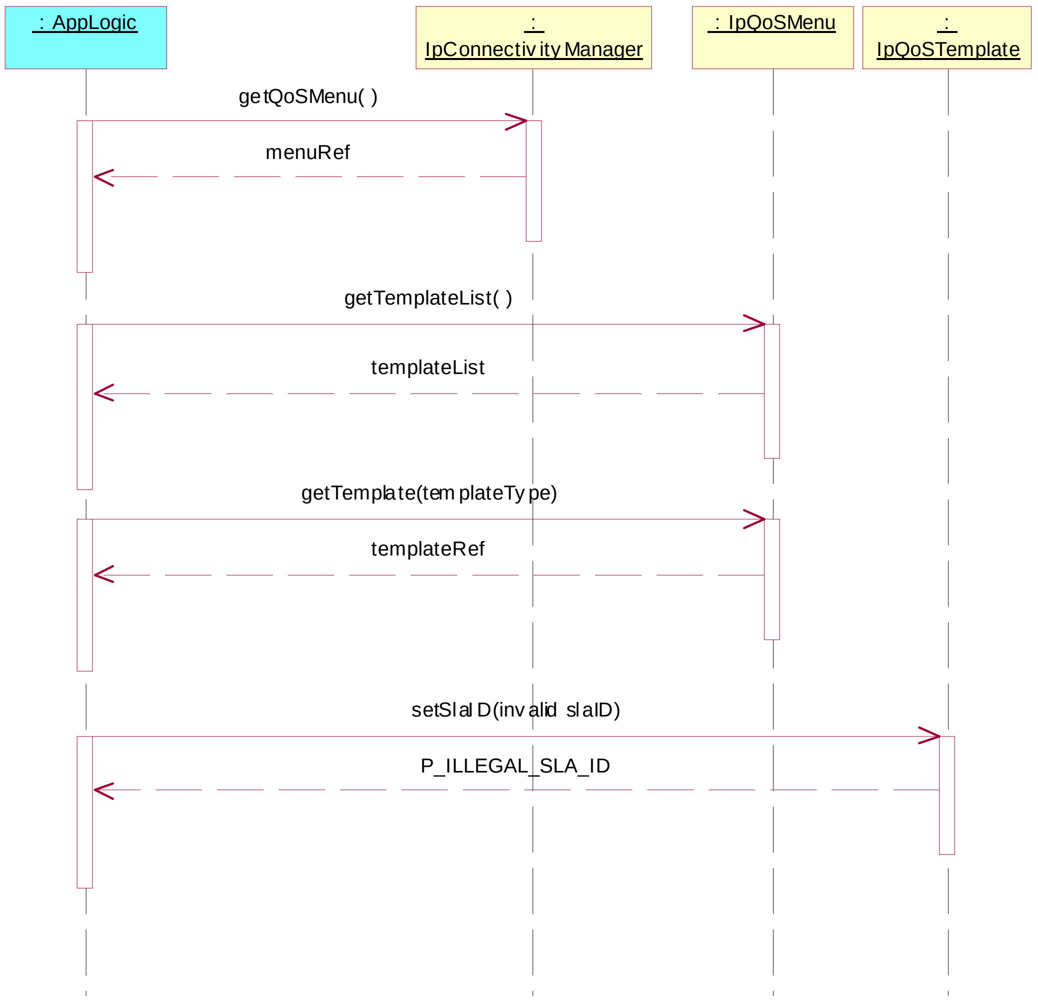
Test Sequence:

1. Method call **getQoSMenu()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getTemplateList()** on IpQoSMenu
   Parameters: None
   Check: valid value of TpStringList is returned

3. Method call **getTemplate()** on IpQoSMenu
   Parameters: Valid value of templateType, which is an item of the TpStringList, returned in 2.
   Check: valid value of IpInterfaceRef is returned

4. Method call **setSlaID()** on IpQoSTemplate
   Parameters: Invalid SlaID
   Check: P_ILLEGAL_SLA_ID exception is returned



Test CM_08

Summary: getPipeQoSInfo, setPipeQoSInfo, successful

Reference: ES 201 915-10 [1], clauses 8.1, 8.4 and 8.5

Test Sequence:

1. Method call **getQoSMenu()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

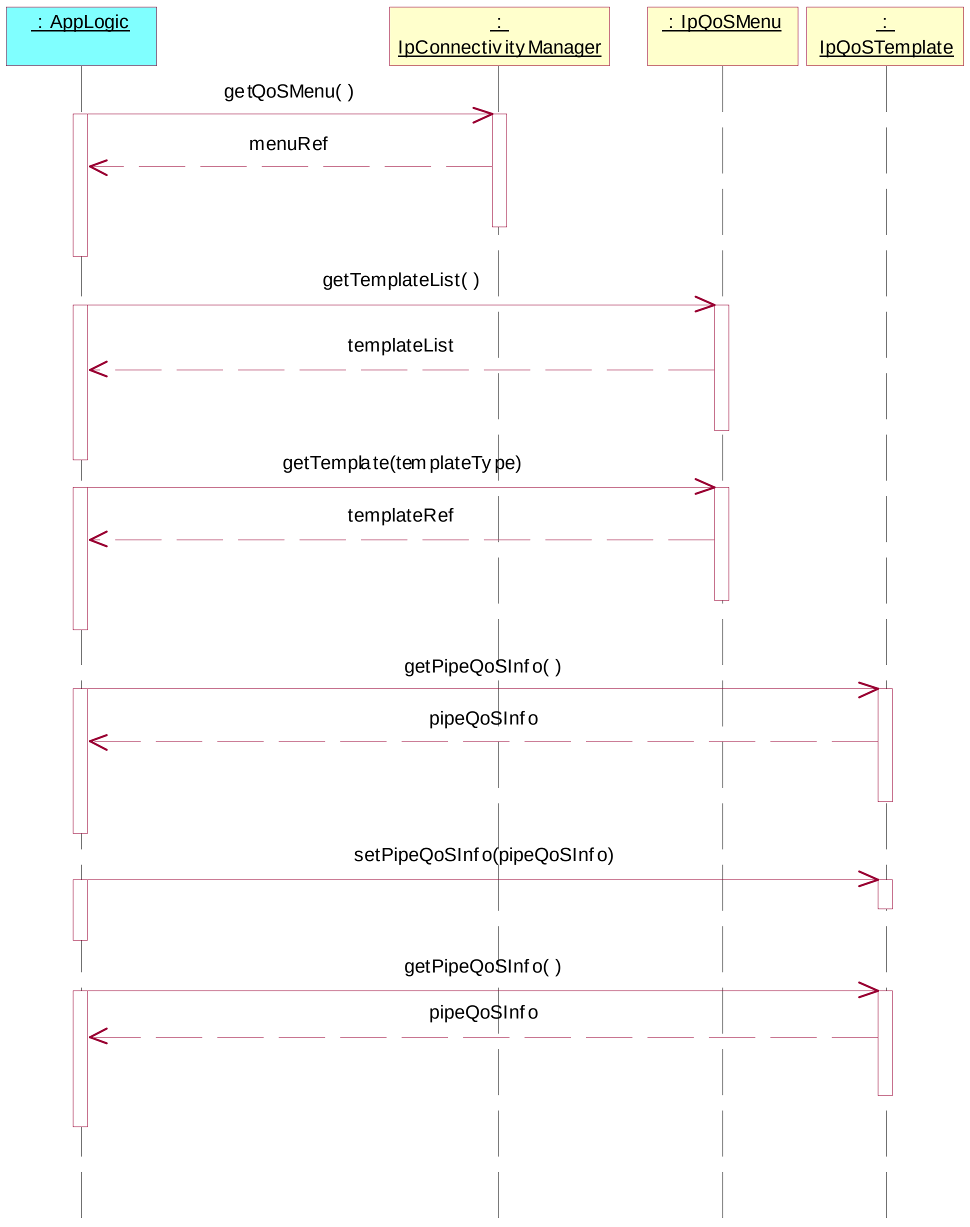
2. Method call **getTemplateList()** on IpQoSMenu
   Parameters: None
   Check: valid value of TpStringList is returned

3. Method call **getTemplate()** on IpQoSMenu
   Parameters: Valid value of templateType, which is an item of the TpStringList, returned in 2.
   Check: valid value of IpInterfaceRef is returned

4. Method call **getPipeQoSInfo()** on IpQoSTemplate
   Parameters: None
   Check: valid TpPipeQoSInfo is returned

5. Method call **setPipeQoSInfo()** on IpQoSTemplate
   Parameters: valid pipeQoSInfo with different values of the structure returned in 4.
   Check: no exception is returned

6. Method call **getPipeQoSInfo()** on IpQoSTemplate
   Parameters: None
   Check: TpPipeQoSInfo given in 5. is returned


: AppLogic
: IpConnectivityManager
: IpQoSMenu
: IpQoSTemplate
getQoSMenu( )
menuRef
getTemplateList( )
templateList
getTemplate(templateType)
templateRef
getPipeQoSInfo( )
pipeQoSInfo
setPipeQoSInfo(pipeQoSInfo)
getPipeQoSInfo( )
pipeQoSInfo

Test CM_09

Summary: setPipeQoSInfo, P_ILLEGAL_TAG

Reference: ES 201 915-10 [1], clauses 8.1, 8.4 and 8.5

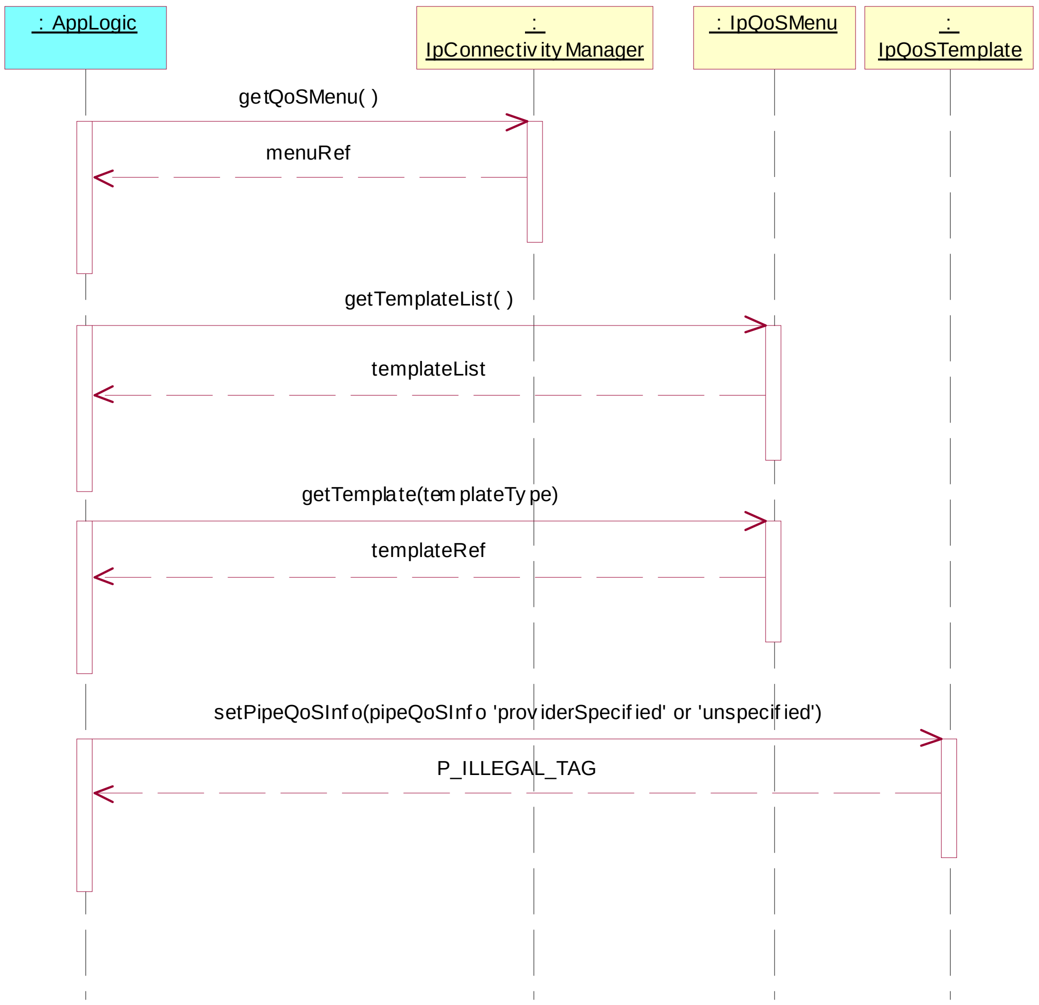
Test Sequence:

1. Method call **getQoSMenu()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getTemplateList()** on IpQoSMenu
   Parameters: None
   Check: valid value of TpStringList is returned

3. Method call **getTemplate()** on IpQoSMenu
   Parameters: Valid value of templateType, which is an item of the TpStringList, returned in 2.
   Check: valid value of IpInterfaceRef is returned

4. Method call **setPipeQoSInfo()** on IpQoSTemplate
   Parameters: pipeQoSInfo with a value tagged with "providerSpecified" or "unspecified".
   Check: P_ILLEGAL_TAG exception is returned



Test CM_10

Summary: setPipeQoSInfo, P_ILLEGAL_VALUE

Reference: ES 201 915-10 [1], clauses 8.1, 8.4 and 8.5

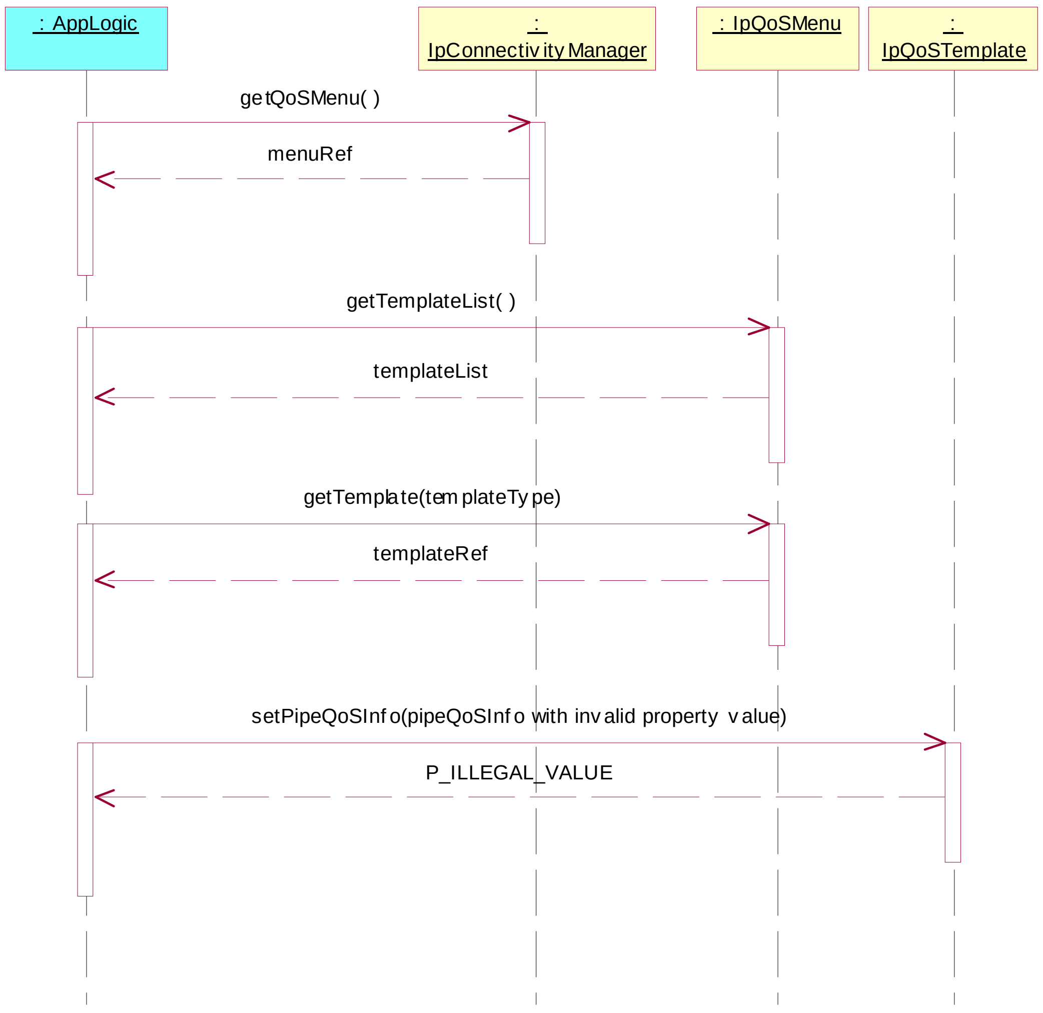
Test Sequence:

1. Method call **getQoSMenu()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getTemplateList()** on IpQoSMenu
   Parameters: None
   Check: valid value of TpStringList is returned

3. Method call **getTemplate()** on IpQoSMenu
   Parameters: Valid value of templateType, which is an item of the TpStringList, returned in 2.
   Check: valid value of IpInterfaceRef is returned

4. Method call **setPipeQoSInfo()** on IpQoSTemplate
   Parameters: pipeQoSInfo with an invalid value in a property.
   Check: P_ILLEGAL_VALUE exception is returned



Test CM_11

Summary: setPipeQoSInfo, P_ILLEGAL_COMBINATION

Reference: ES 201 915-10 [1], clauses 8.1, 8.4 and 8.5

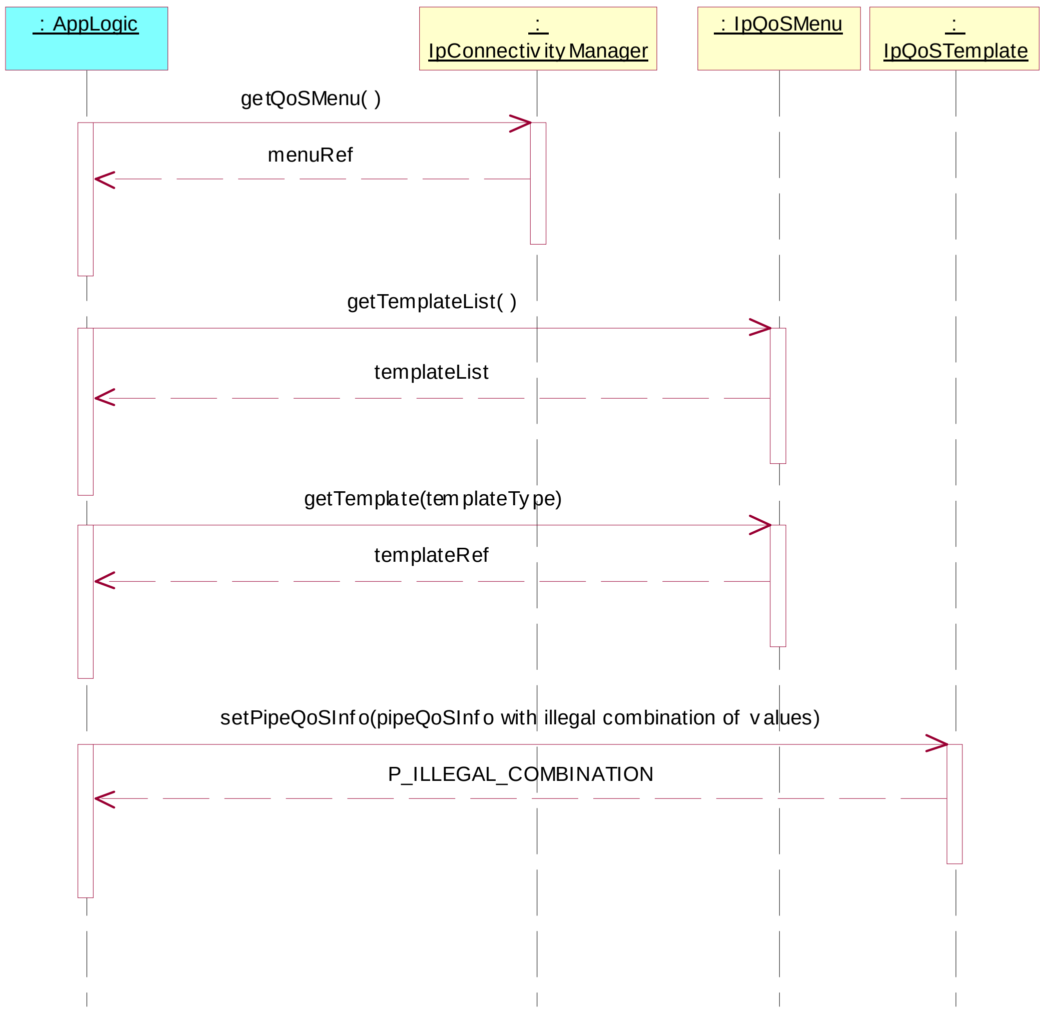
Test Sequence:

1. Method call **getQoSMenu()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getTemplateList()** on IpQoSMenu
   Parameters: None
   Check: valid value of TpStringList is returned

3. Method call **getTemplate()** on IpQoSMenu
   Parameters: Valid value of templateType, which is an item of the TpStringList, returned in 2.
   Check: valid value of IpInterfaceRef is returned

4. Method call **setPipeQoSInfo()** on IpQoSTemplate
   Parameters: pipeQoSInfo with an invalid combination of values.
   Check: P_ILLEGAL_COMBINATION exception is returned



Test CM_12

Summary: getValidityInfo, setValidityInfo, successful

Reference: ES 201 915-10 [1], clauses 8.1, 8.4 and 8.5

Test Sequence:

1. Method call **getQoSMenu()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

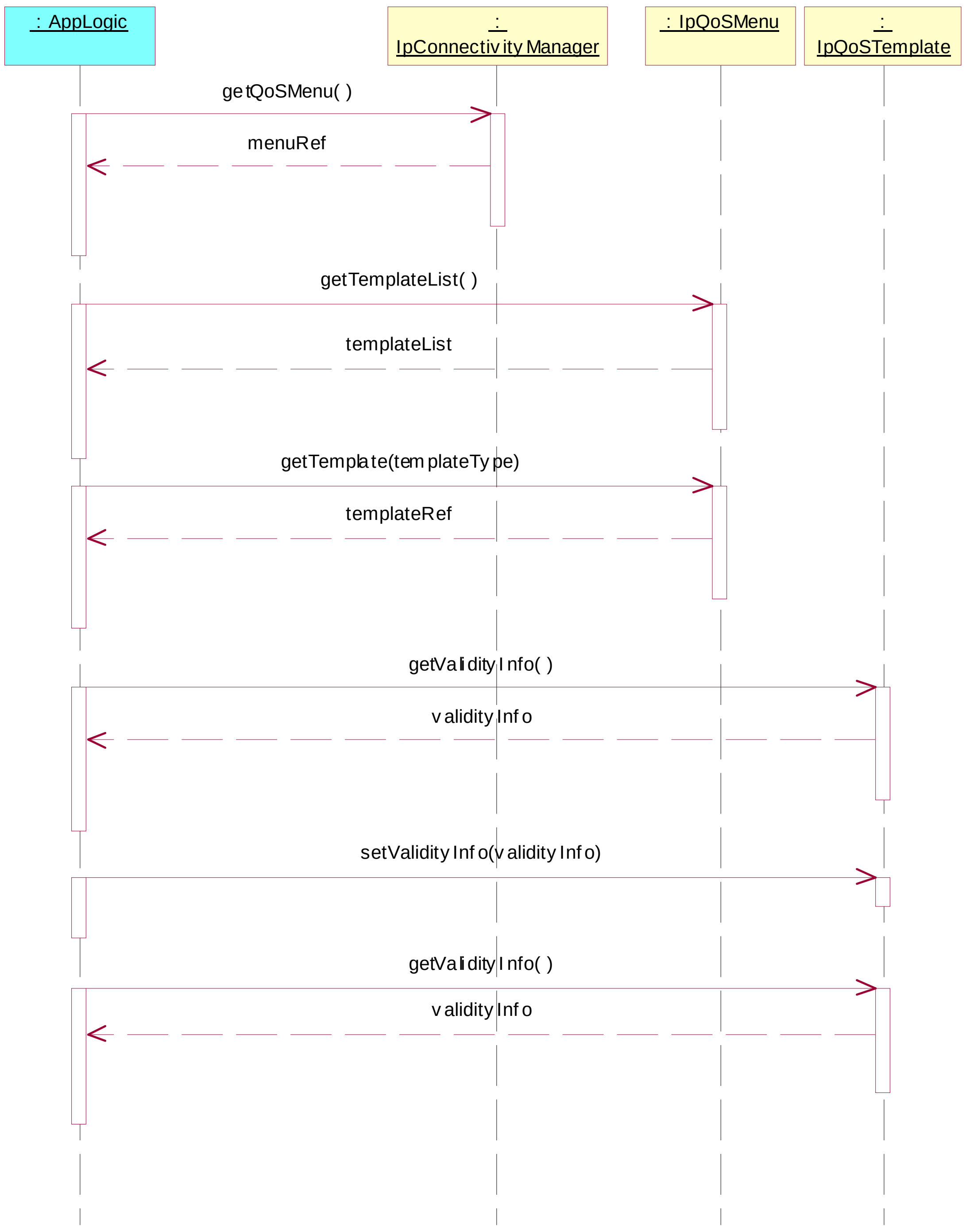
2. Method call **getTemplateList()** on IpQoSMenu
   Parameters: None
   Check: valid value of TpStringList is returned

3. Method call **getTemplate()** on IpQoSMenu
   Parameters: Valid value of templateType, which is an item of the TpStringList, returned in 2.
   Check: valid value of IpInterfaceRef is returned

4. Method call **getValidityInfo()** on IpQoSTemplate
   Parameters: None
   Check: valid TpValidityInfo is returned

5. Method call **setValidityInfo()** on IpQoSTemplate
   Parameters: valid validityInfo with different values of the structure returned in 4.
   Check: no exception is returned

6. Method call **getValidityInfo()** on IpQoSTemplate
   Parameters: None
   Check: TpValidityInfo given in 5. is returned


: AppLogic
: IpConnectivityManager
: IpQoSMenu
: IpQoSTemplate
getQoSMenu( )
menuRef
getTemplateList( )
templateList
getTemplate(templateType)
templateRef
getValidityInfo( )
validityInfo
setValidityInfo(validityInfo)
getValidityInfo( )
validityInfo

Test CM_13

Summary: setValidityInfo, P_ILLEGAL_TAG

Reference: ES 201 915-10 [1], clauses 8.1, 8.4 and 8.5

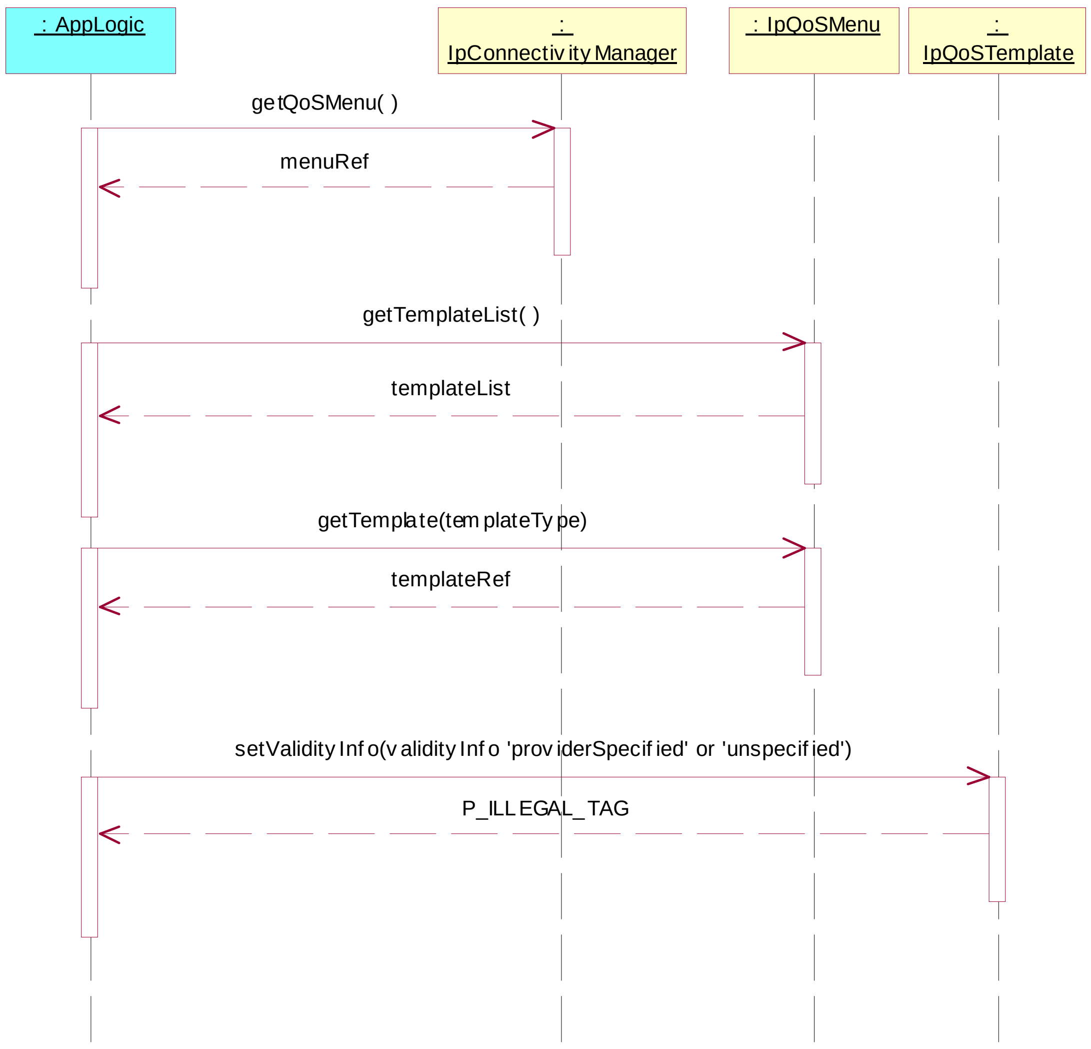
Test Sequence:

1. Method call **getQoSMenu()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getTemplateList()** on IpQoSMenu
   Parameters: None
   Check: valid value of TpStringList is returned

3. Method call **getTemplate()** on IpQoSMenu
   Parameters: Valid value of templateType, which is an item of the TpStringList, returned in 2.
   Check: valid value of IpInterfaceRef is returned

4. Method call **setValidityInfo()** on IpQoSTemplate
   Parameters: validityInfo with a value tagged with "providerSpecified" or "unspecified".
   Check: P_ILLEGAL_TAG exception is returned



Test CM_14

Summary: setValidityInfo, P_ILLEGAL_VALUE

Reference: ES 201 915-10 [1], clauses 8.1, 8.4 and 8.5

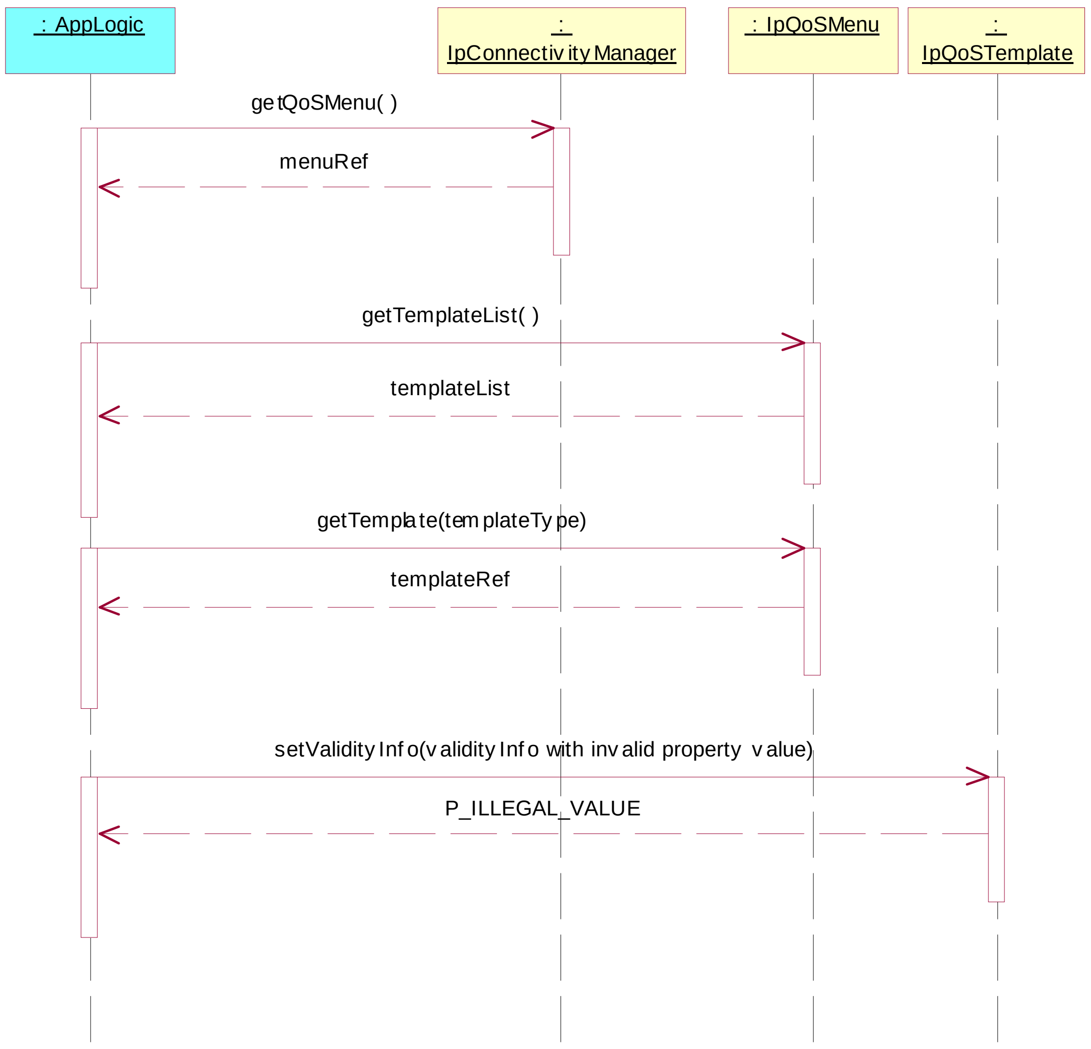
Test Sequence:

1. Method call **getQoSMenu()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getTemplateList()** on IpQoSMenu
   Parameters: None
   Check: valid value of TpStringList is returned

3. Method call **getTemplate()** on IpQoSMenuParameters: Valid value of templateType, which is an item of the TpStringList, returned in 2.
   Check: valid value of IpInterfaceRef is returned

4. Method call **setValidityInfo()** on IpQoSTemplate
   Parameters: validityInfo with an invalid value in a property.
   Check: P_ILLEGAL_VALUE exception is returned



Test CM_15

Summary: setValidityInfo, P_ILLEGAL_COMBINATION

Reference: ES 201 915-10 [1], clauses 8.1, 8.4 and 8.5

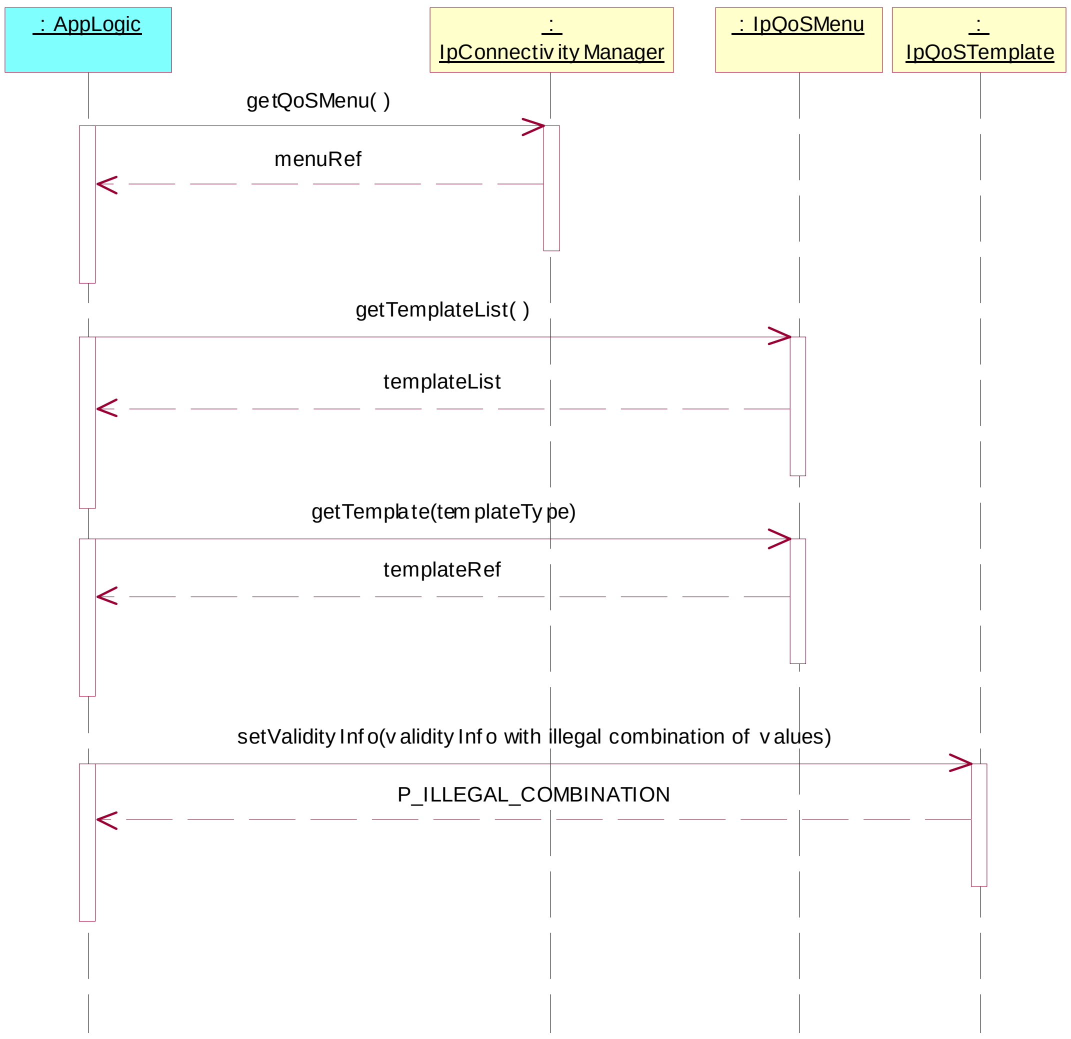
Test Sequence:

1. Method call **getQoSMenu()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getTemplateList()** on IpQoSMenu
   Parameters: None
   Check: valid value of TpStringList is returned

3. Method call **getTemplate()** on IpQoSMenu
   Parameters: Valid value of templateType, which is an item of the TpStringList, returned in 2.
   Check: valid value of IpInterfaceRef is returned

4. Method call **setValidityInfo()** on IpQoSTemplate
   Parameters: validityInfo with an invalid combination of values.
   Check: P_ILLEGAL_COMBINATION exception is returned



Test CM_16

Summary: getProvisionedQoSInfo, setProvisionedQoSInfo, successful

Reference: ES 201 915-10 [1], clauses 8.1, 8.4 and 8.5

Test Sequence:

1. Method call **getQoSMenu()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

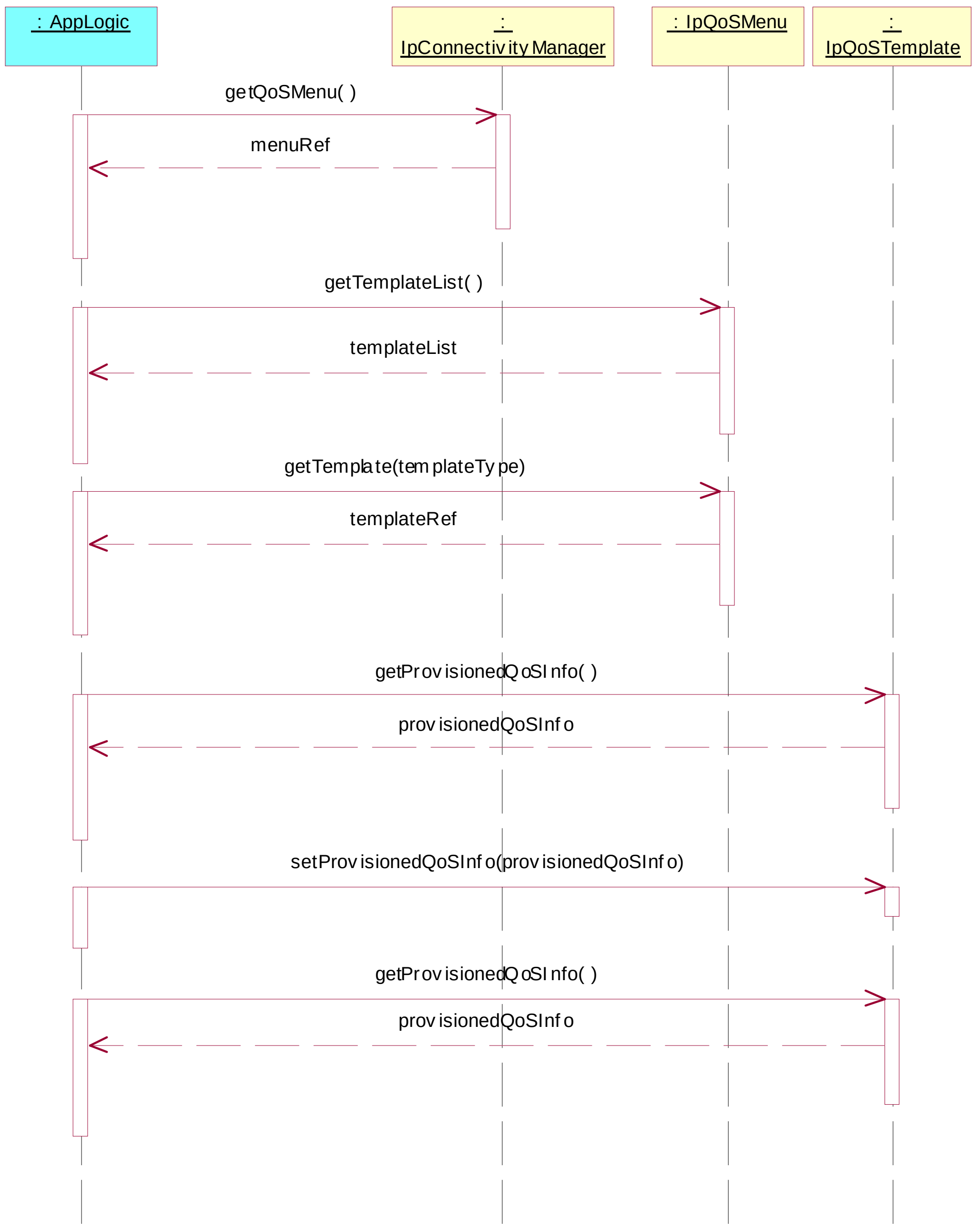
2. Method call **getTemplateList()** on IpQoSMenu
   Parameters: None
   Check: valid value of TpStringList is returned

3. Method call **getTemplate()** on IpQoSMenu
   Parameters: Valid value of templateType, which is an item of the TpStringList, returned in 2.
   Check: valid value of IpInterfaceRef is returned

4. Method call **getProvisionedQoSInfo()** on IpQoSTemplate
   Parameters: None
   Check: valid TpProvisionedQoSInfo is returned

5. Method call **setProvisionedQoSInfo()** on IpQoSTemplate
   Parameters: valid ProvisionedQoSInfo with different values of the structure returned in 4.
   Check: no exception is returned

6. Method call **getProvisionedQoSInfo()** on IpQoSTemplate
   Parameters: None
   Check: TpProvisionedQoSInfo given in 5. is returned


: AppLogic
: IpConnectivityManager
: IpQoSMenu
: IpQoSTemplate
getQoSMenu( )
menuRef
getTemplateList( )
templateList
getTemplate(templateType)
templateRef
getProvisionedQoSInfo( )
provisionedQoSInfo
setProvisionedQoSInfo(provisionedQoSInfo)
getProvisionedQoSInfo( )
provisionedQoSInfo

Test CM_17

Summary: setProvisionedQoSInfo, P_ILLEGAL_TAG

Reference: ES 201 915-10 [1], clauses 8.1, 8.4 and 8.5

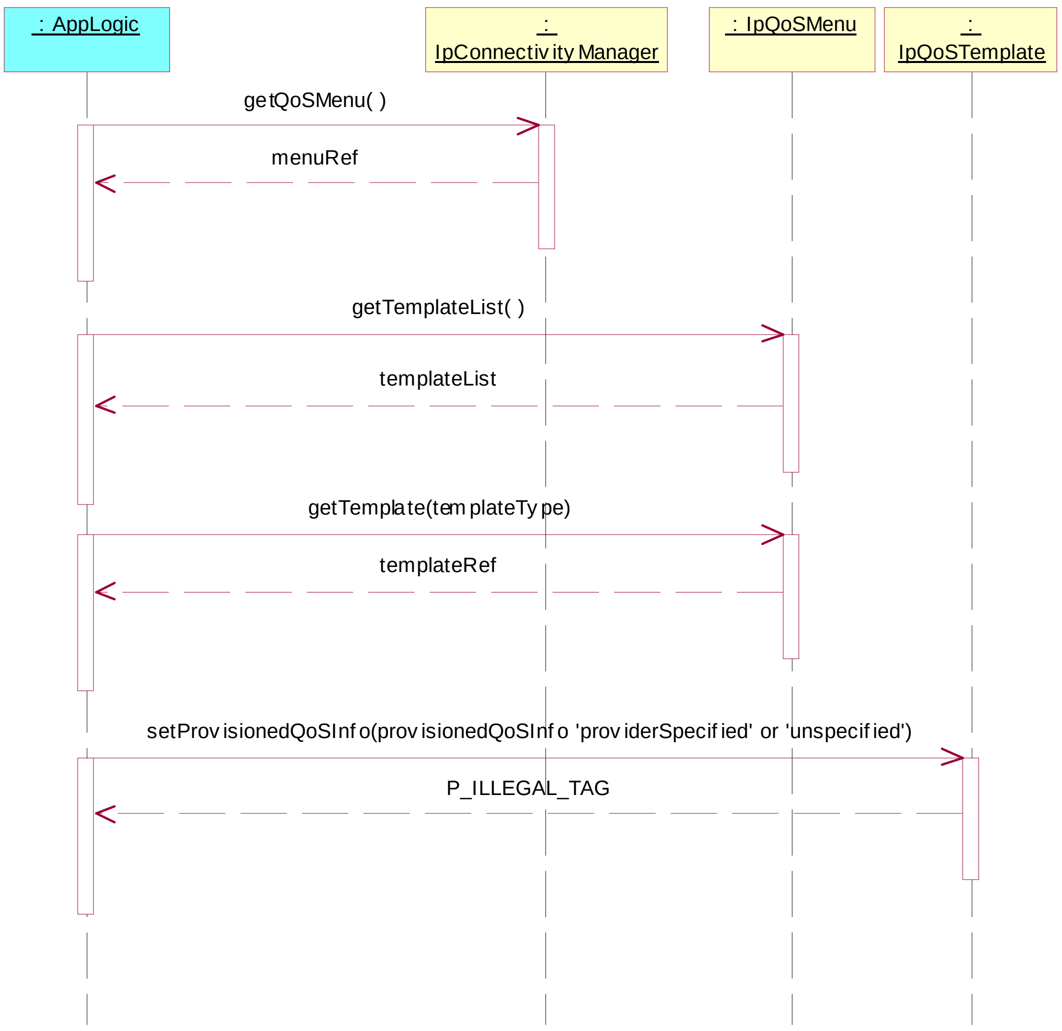
Test Sequence:

1. Method call **getQoSMenu()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getTemplateList()** on IpQoSMenu
   Parameters: None
   Check: valid value of TpStringList is returned

3. Method call **getTemplate()** on IpQoSMenu
   Parameters: Valid value of templateType, which is an item of the TpStringList, returned in 2.
   Check: valid value of IpInterfaceRef is returned

4. Method call **setProvisionedQoSInfo()** on IpQoSTemplate
   Parameters: provisionedQoSInfo with a value tagged with "providerSpecified" or "unspecified".
   Check: P_ILLEGAL_TAG exception is returned



Test CM_18

Summary: setProvisionedQoSInfo, P_ILLEGAL_VALUE

Reference: ES 201 915-10 [1], clauses 8.1, 8.4 and 8.5

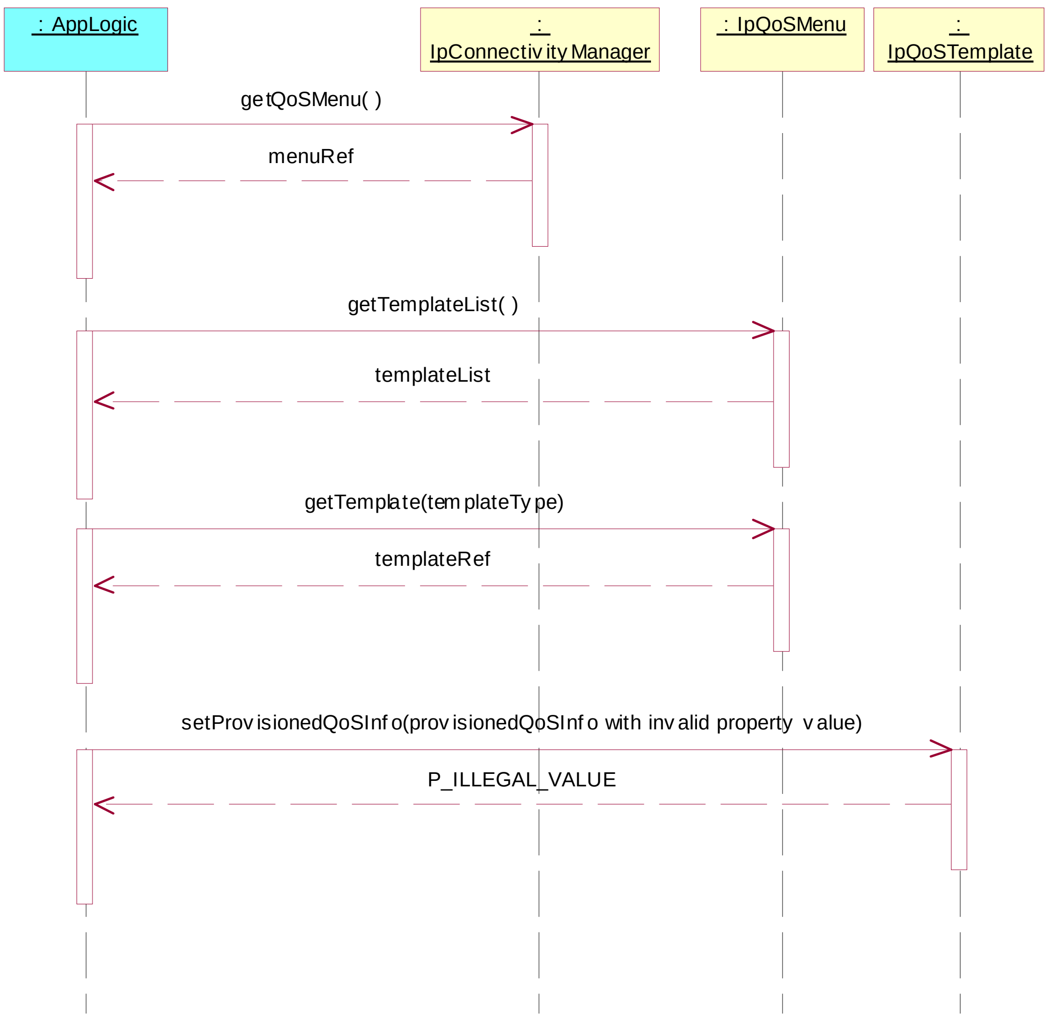
Test Sequence:

1. Method call **getQoSMenu()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getTemplateList()** on IpQoSMenu
   Parameters: None
   Check: valid value of TpStringList is returned

3. Method call **getTemplate()** on IpQoSMenu
   Parameters: Valid value of templateType, which is an item of the TpStringList, returned in 2.
   Check: valid value of IpInterfaceRef is returned

4. Method call **setProvisionedQoSInfo()** on IpQoSTemplate
   Parameters: provisionedQoSInfo with an invalid value in a property.
   Check: P_ILLEGAL_VALUE exception is returned



Test CM_19

Summary: setProvisionedQoSInfo, P_ILLEGAL_COMBINATION

Reference: ES 201 915-10 [1], clauses 8.1, 8.4 and 8.5

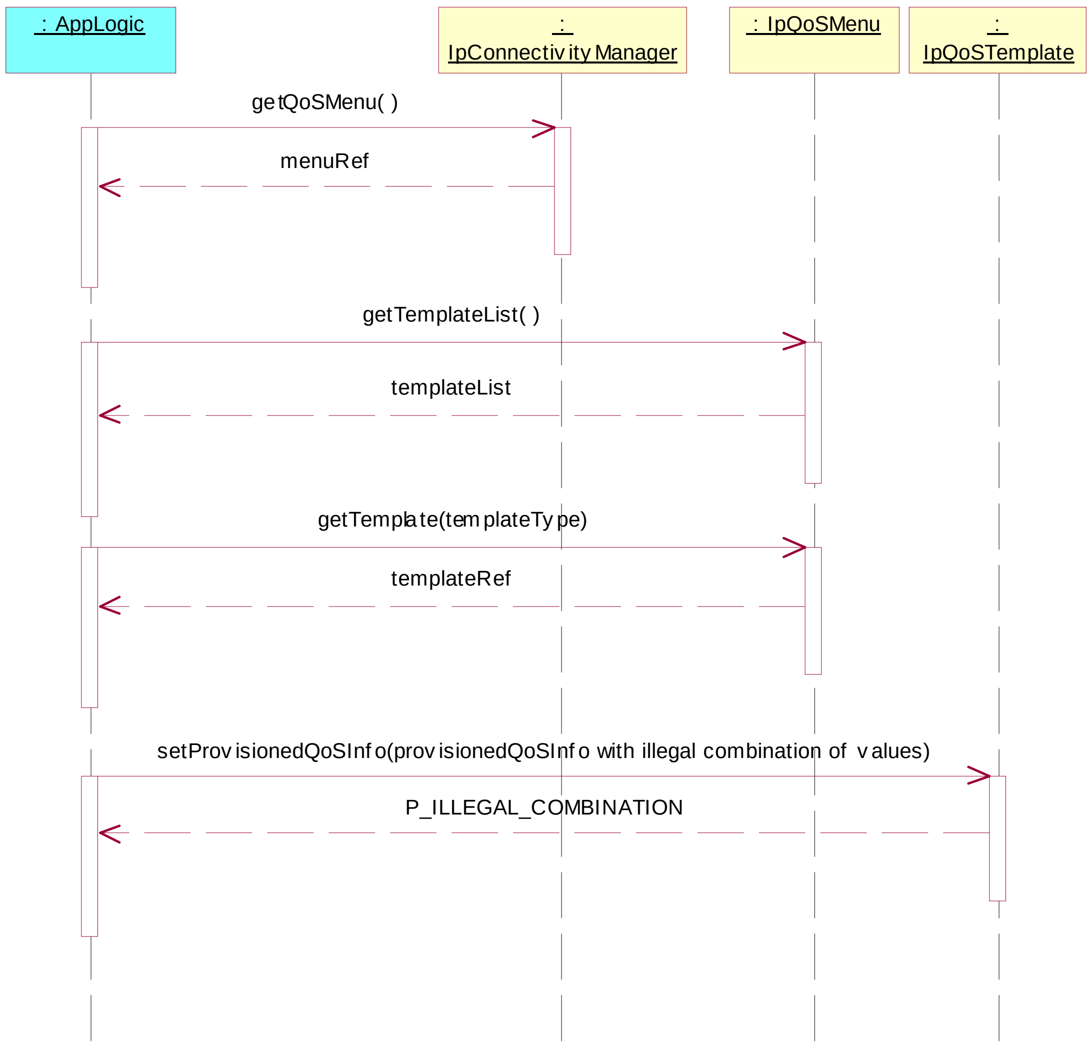
Test Sequence:

1. Method call **getQoSMenu()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getTemplateList()** on IpQoSMenu
   Parameters: None
   Check: valid value of TpStringList is returned

3. Method call **getTemplate()** on IpQoSMenu
   Parameters: Valid value of templateType, which is an item of the TpStringList, returned in 2.
   Check: valid value of IpInterfaceRef is returned

4. Method call **setProvisionedQoSInfo()** on IpQoSTemplate
   Parameters: provisionedQoSInfo with an invalid combination of values.
   Check: P_ILLEGAL_COMBINATION exception is returned



Test CM_20

Summary: getDsCodepoint, successful

Reference: ES 201 915-10 [1], clauses 8.1, 8.4 and 8.5

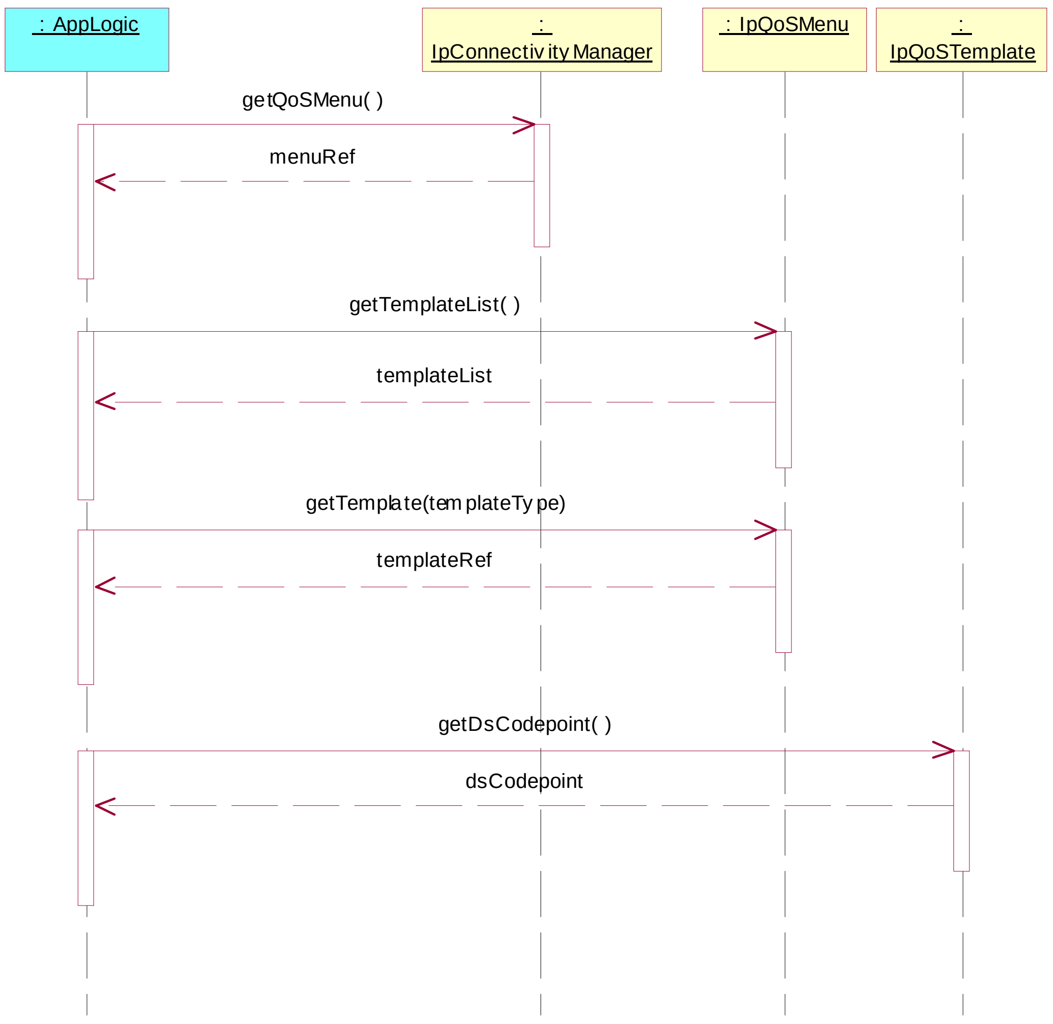
Test Sequence:

1. Method call **getQoSMenu()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getTemplateList()** on IpQoSMenu
   Parameters: None
   Check: valid value of TpStringList is returned

3. Method call **getTemplate()** on IpQoSMenu
   Parameters: Valid value of templateType, which is an item of the TpStringList, returned in 2.
   Check: valid value of IpInterfaceRef is returned

4. Method call **getDsCodepoint()** on IpQoSTemplate
   Parameters: None.
   Check: valid value of TpDsCodePoint is returned



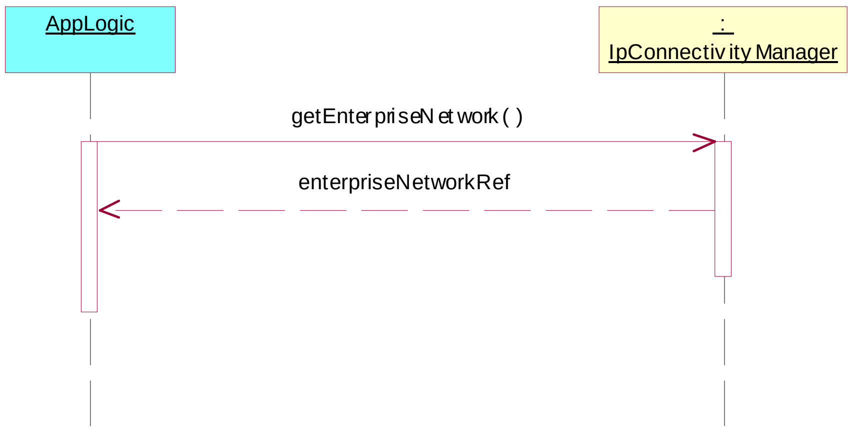
Test CM_21

Summary: getEnterpriseNetwork, successful

Reference: ES 201 915-10 [1], clause 8.1

Test Sequence:

1. Method call **getEnterpriseNetwork()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned



Test CM_22

Summary: getSiteList, getSite, successful

Reference: ES 201 915-10 [1], clause 8.1 and 8.2

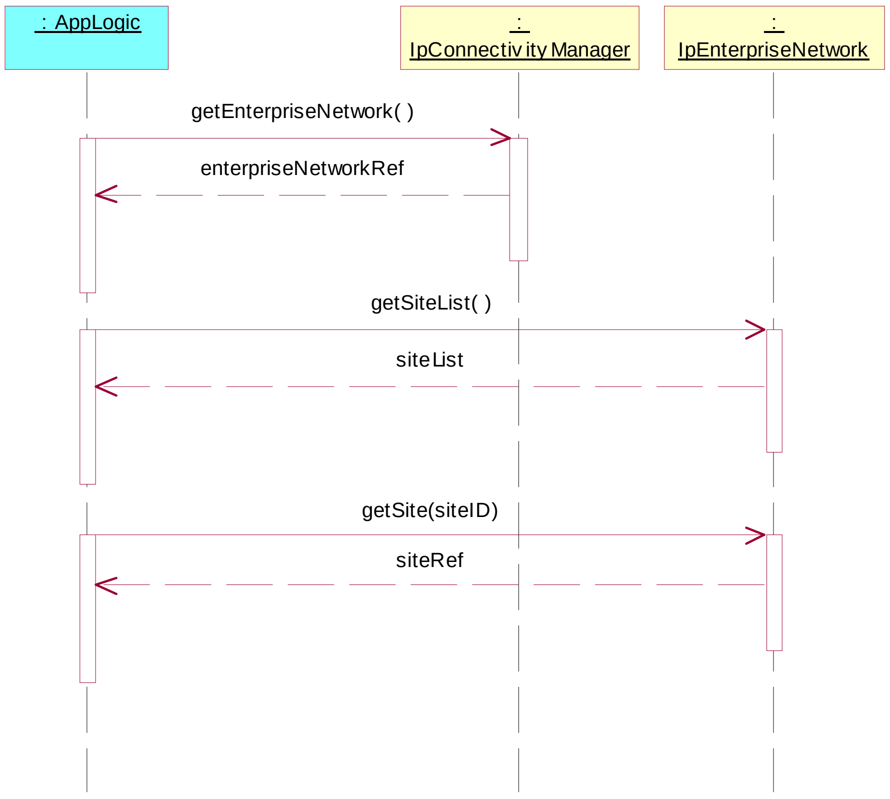
Preamble: The enterprise network must have at least one site ID that is serviced by the provider network.

Test Sequence:

1. Method call **getEnterpriseNetwork()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getSiteList()** on IpEnterpriseNetwork
   Parameters: None
   Check: valid value of TpStringList is returned

3. Method call **getSite()** on IpEnterpriseNetwork
   Parameters: Valid value of siteID, which is an item of the TpStringList, returned in 2.
   Check: valid value of IpInterfaceRef is returned



Test CM_23

Summary: getSite, P_UNKNOWN_SITE_ID

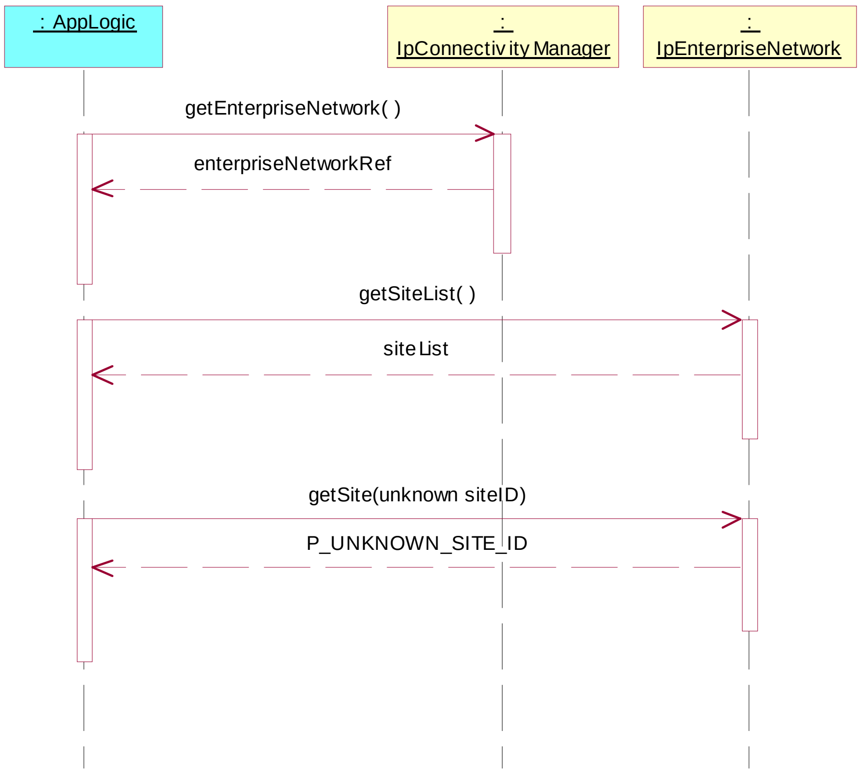
Reference: ES 201 915-10 [1], clauses 8.1 and 8.2

Test Sequence:

1. Method call **getEnterpriseNetwork()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getSiteList()** on IpEnterpriseNetwork
   Parameters: None
   Check: valid value of TpStringList is returned

3. Method call **getSite()** on IpEnterpriseNetwork
   Parameters: Valid value of siteID, which is not an item of the TpStringList, returned in 2.
   Check: P_UNKNOWN_SITE_ID is returned



Test CM_24

Summary: getSite, P_ILLEGAL_SITE_ID

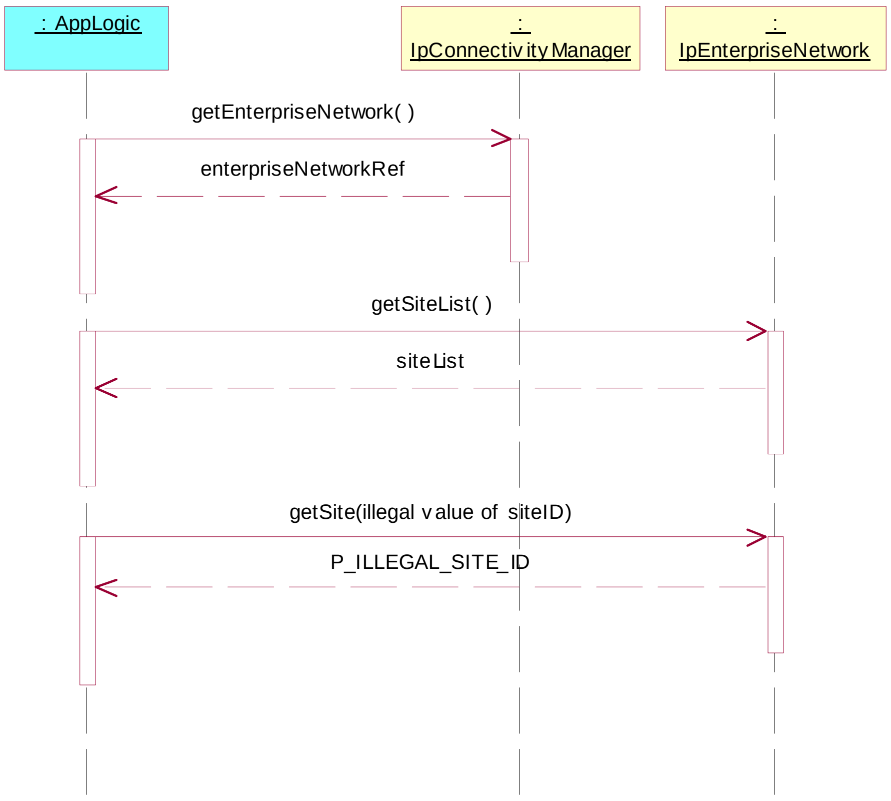
Reference: ES 201 915-10 [1], clauses 8.1 and 8.2

Test Sequence:

1. Method call **getEnterpriseNetwork()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getSiteList()** on IpEnterpriseNetwork
   Parameters: None
   Check: valid value of TpStringList is returned

3. Method call **getSite()** on IpEnterpriseNetwork
   Parameters: Invalid value of siteID
   Check: P_ILLEGAL_SITE_ID is returned



Test CM_25

Summary: getSAPList, successful

Reference: ES 201 915-10 [1], clauses 8.1, 8.2 and 8.3

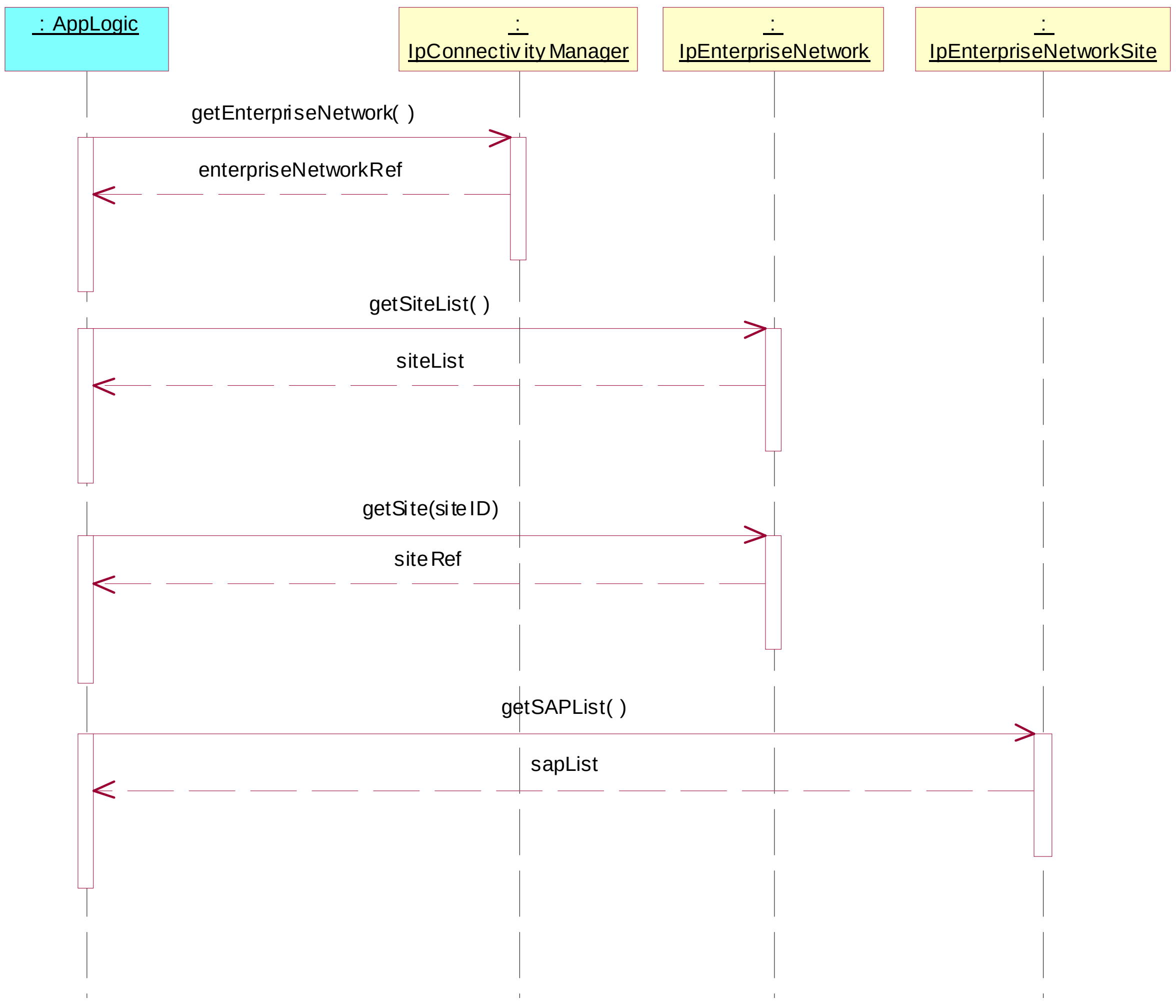
Test Sequence:

1. Method call **getEnterpriseNetwork()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getSiteList()** on IpEnterpriseNetwork
   Parameters: None
   Check: valid value of TpStringList is returned

3. Method call **getSite()** on IpEnterpriseNetwork
   Parameters: Valid value of siteID, which is an item of the TpStringList, returned in 2.
   Check: valid value of IpInterfaceRef is returned

4. Method call **getSAPList()** on IpEnterpriseNetworkSite
   Parameters: None
   Check: valid value of TpStringList is returned



Test CM_26

Summary: getSAPIPSubnet, successful

Reference: ES 201 915-10 [1], clauses 8.1, 8.2 and 8.3

Test Sequence:

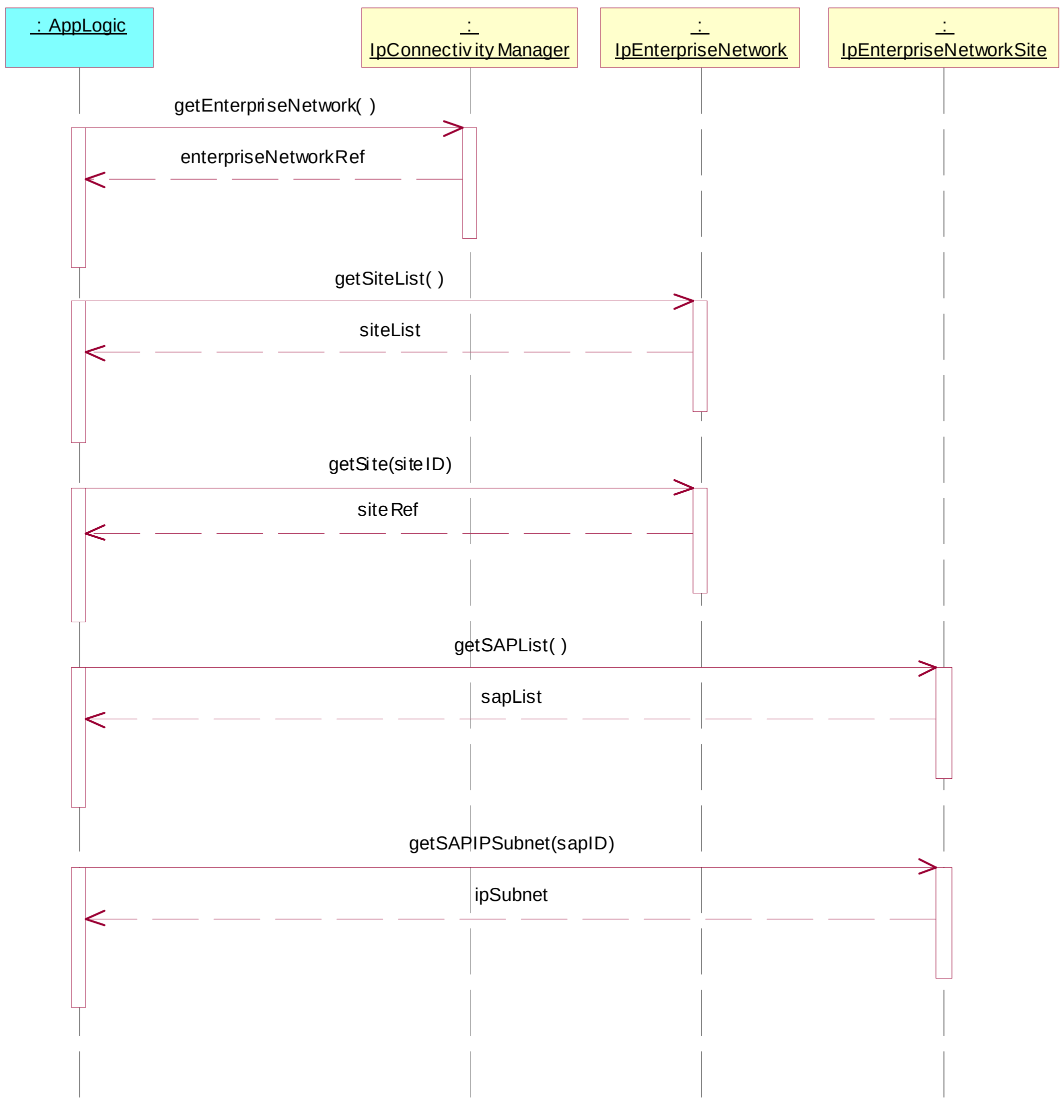
1. Method call **getEnterpriseNetwork()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getSiteList()** on IpEnterpriseNetwork
   Parameters: None
   Check: valid value of TpStringList is returned

3. Method call **getSite()** on IpEnterpriseNetwork
   Parameters: Valid value of siteID, which is an item of the TpStringList, returned in 2.
   Check: valid value of IpInterfaceRef is returned

4. Method call **getSAPList()** on IpEnterpriseNetworkSite
   Parameters: None
   Check: valid value of TpStringList is returned

5. Method call **getSAPIPSubnet()** on IpEnterpriseNetworkSite
   Parameters: Valid value of sapID, which is an item of the TpStringList, returned in 4.
   Check: valid value of TpIpSubnet is returned



Test CM_27

Summary: getIPSubnet, P_UNKNOWN_SAP

Reference: ES 201 915-10 [1], clauses 8.1, 8.2 and 8.3

Test Sequence:

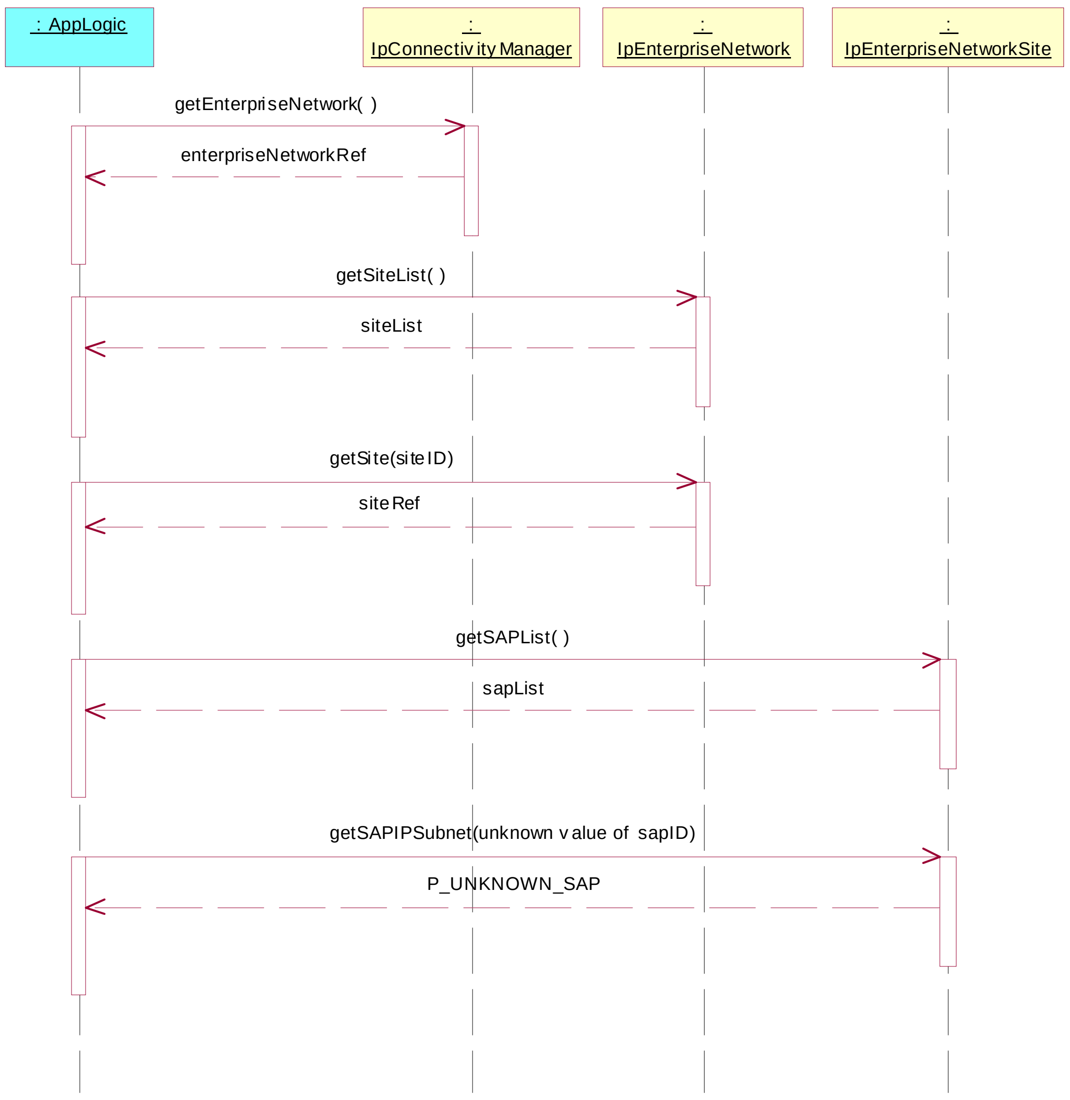
1. Method call **getEnterpriseNetwork()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getSiteList()** on IpEnterpriseNetwork
   Parameters: None
   Check: valid value of TpStringList is returned

3. Method call **getSite()** on IpEnterpriseNetwork
   Parameters: Valid value of siteID, which is an item of the TpStringList, returned in 2.
   Check: valid value of IpInterfaceRef is returned

4. Method call **getSAPList()** on IpEnterpriseNetworkSite
   Parameters: None
   Check: valid value of TpStringList is returned

5. Method call **getSAPIPSubnet()** on IpEnterpriseNetworkSite
   Parameters: Valid value of sapID, which is not an item of the TpStringList, returned in 4.
   Check: P_UNKNOWN_SAP is returned



Test CM_28

Summary: getIPSubnet, P_ILLEGAL_SITE_ID

Reference: ES 201 915-10 [1], clauses 8.1, 8.2 and 8.3

Test Sequence:

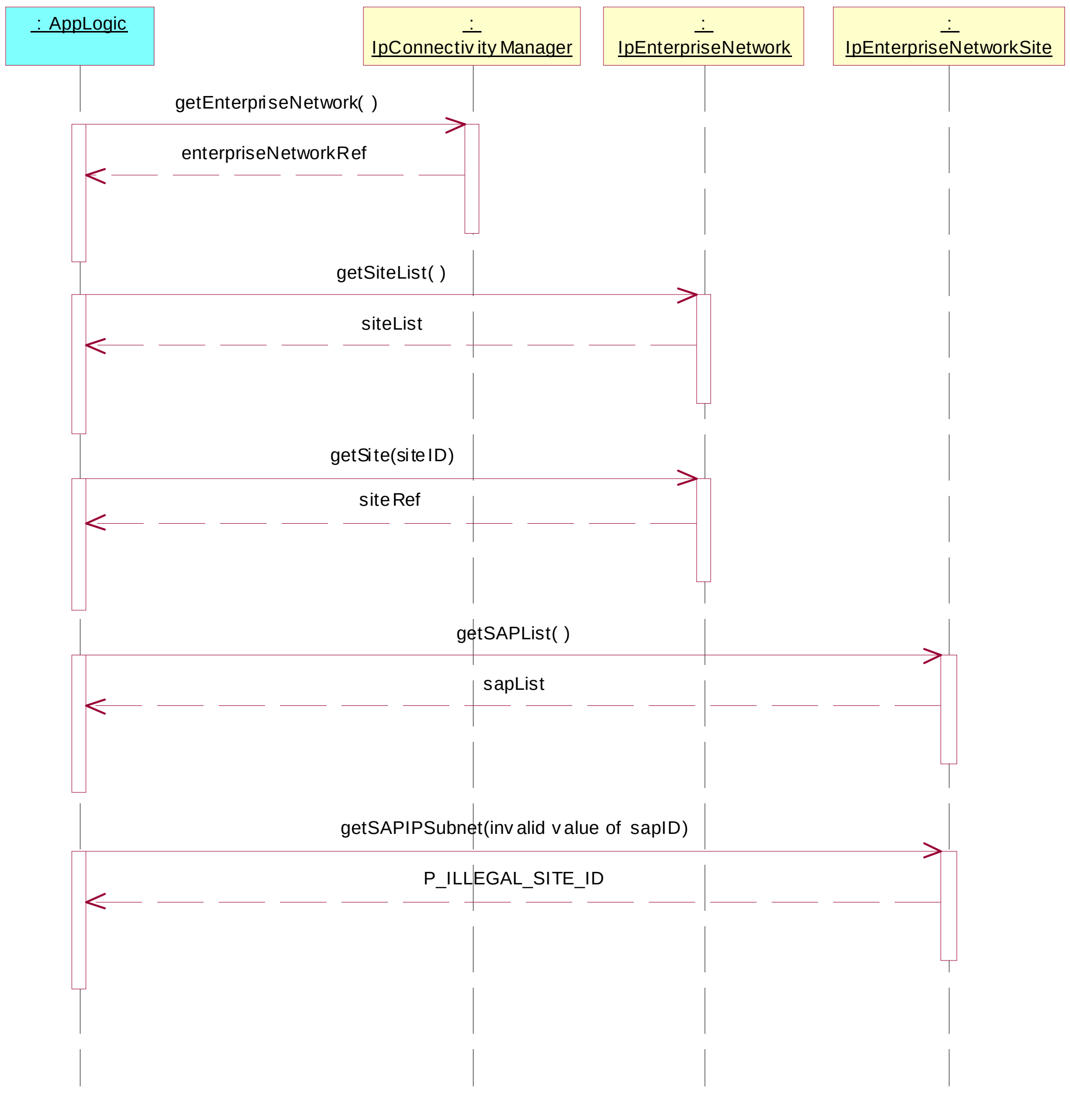
1. Method call **getEnterpriseNetwork()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getSiteList()** on IpEnterpriseNetwork
   Parameters: None
   Check: valid value of TpStringList is returned

3. Method call **getSite()** on IpEnterpriseNetwork
   Parameters: Valid value of siteID, which is an item of the TpStringList, returned in 2.
   Check: valid value of IpInterfaceRef is returned

4. Method call **getSAPList()** on IpEnterpriseNetworkSite
   Parameters: None
   Check: valid value of TpStringList is returned

5. Method call **getSAPIPSubnet ()** on IpEnterpriseNetworkSite
   Parameters: Invalid value of sapID.
   Check: P_ILLEGAL_SITE_ID is returned



Test CM_29

Summary: getSiteID, successful

Reference: ES 201 915-10 [1], clauses 8.1, 8.2 and 8.3

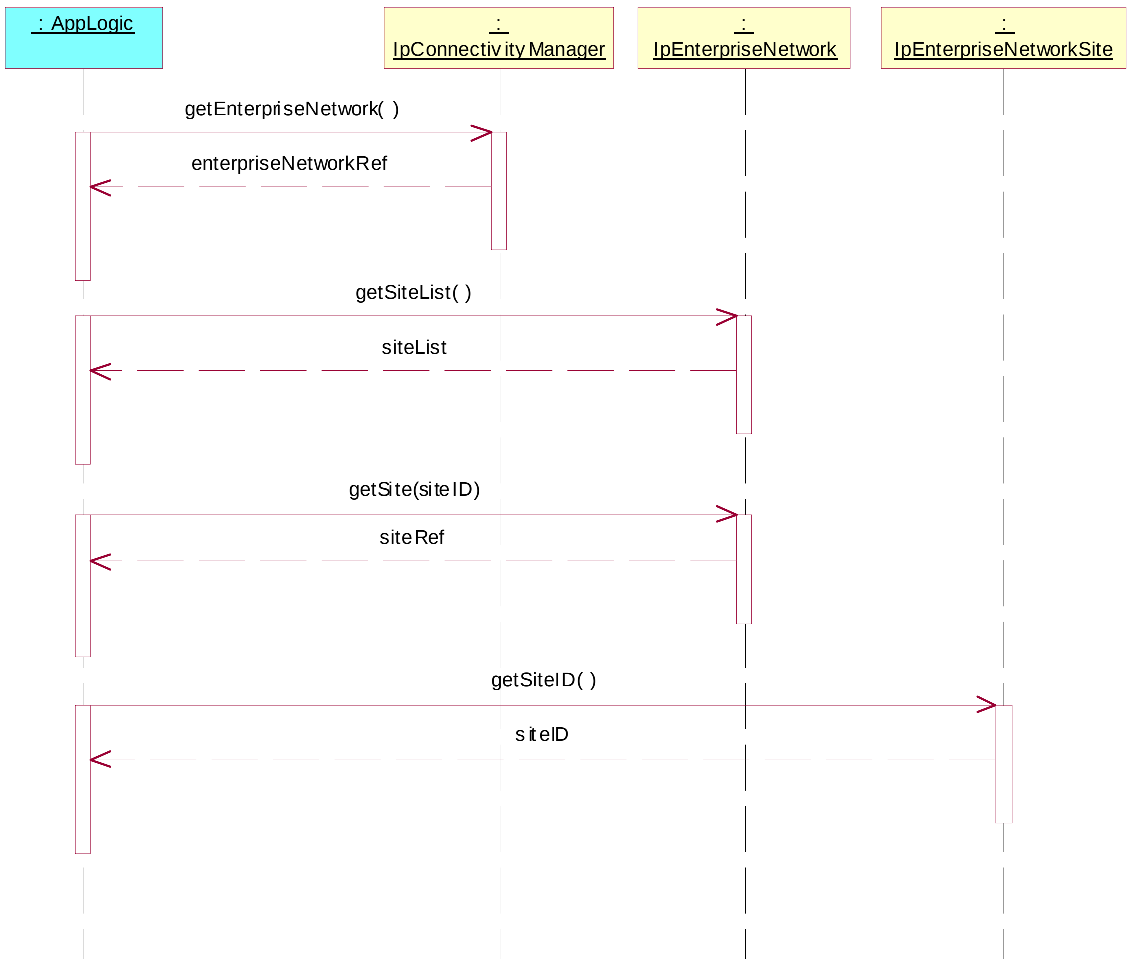
Test Sequence:

1. Method call **getEnterpriseNetwork()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getSiteList()** on IpEnterpriseNetwork
   Parameters: None
   Check: valid value of TpStringList is returned

3. Method call **getSite()** on IpEnterpriseNetwork
   Parameters: Invalid value of siteID.
   Check: valid value of IpInterfaceRef is returned

4. Method call **getSiteID()** on IpEnterpriseNetworkSite
   Parameters: None
   Check: valid value of TpString is returned



Test CM_30

Summary: getSiteLocation, successful

Reference: ES 201 915-10 [1], clauses 8.1, 8.2 and 8.3

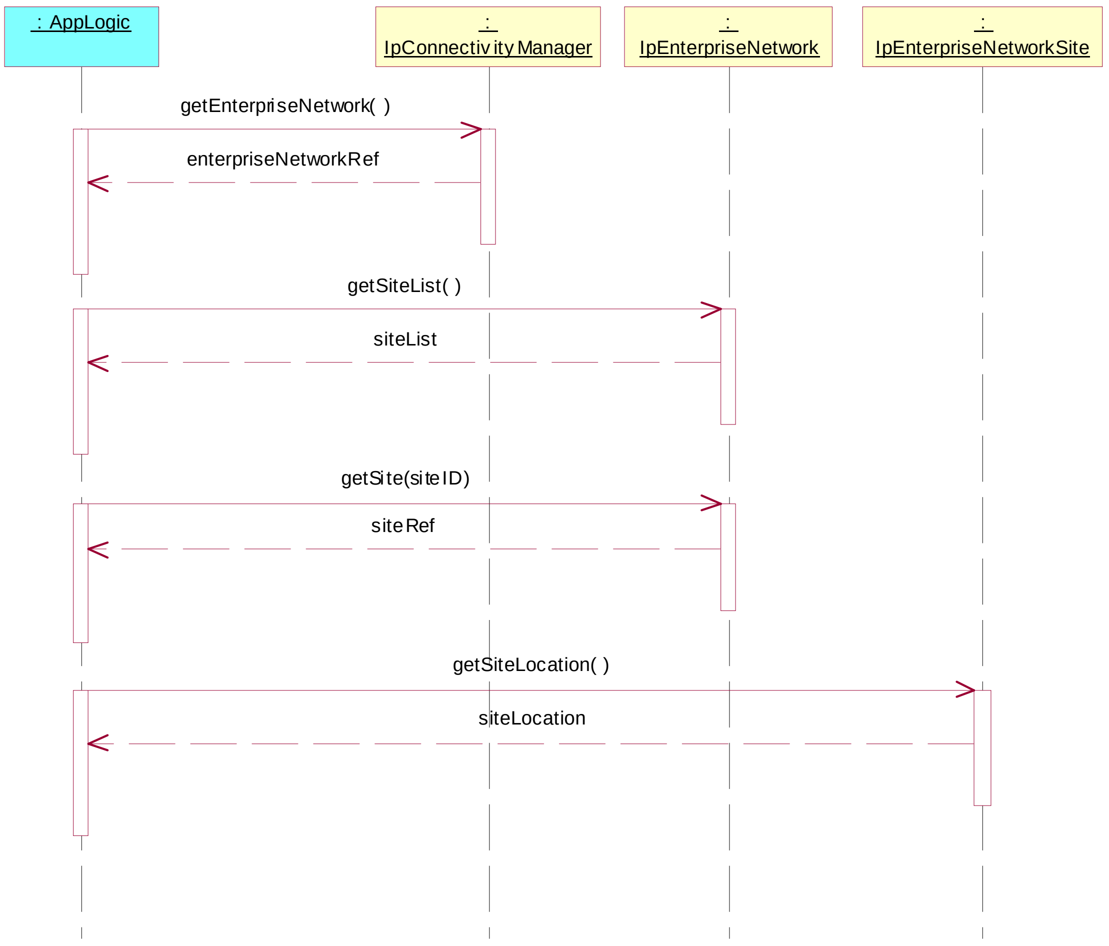
Test Sequence:

1. Method call **getEnterpriseNetwork()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getSiteList()** on IpEnterpriseNetwork
   Parameters: None
   Check: valid value of TpStringList is returned

3. Method call **getSite()** on IpEnterpriseNetwork
   Parameters: Valid value of siteID, which is an item of the TpStringList, returned in 2.
   Check: valid value of IpInterfaceRef is returned

4. Method call **getSiteLocation()** on IpEnterpriseNetworkSite
   Parameters: None
   Check: valid value of TpString is returned



Test CM_31

Summary: getSiteDescription, successful

Reference: ES 201 915-10 [1], clauses 8.1, 8.2 and 8.3

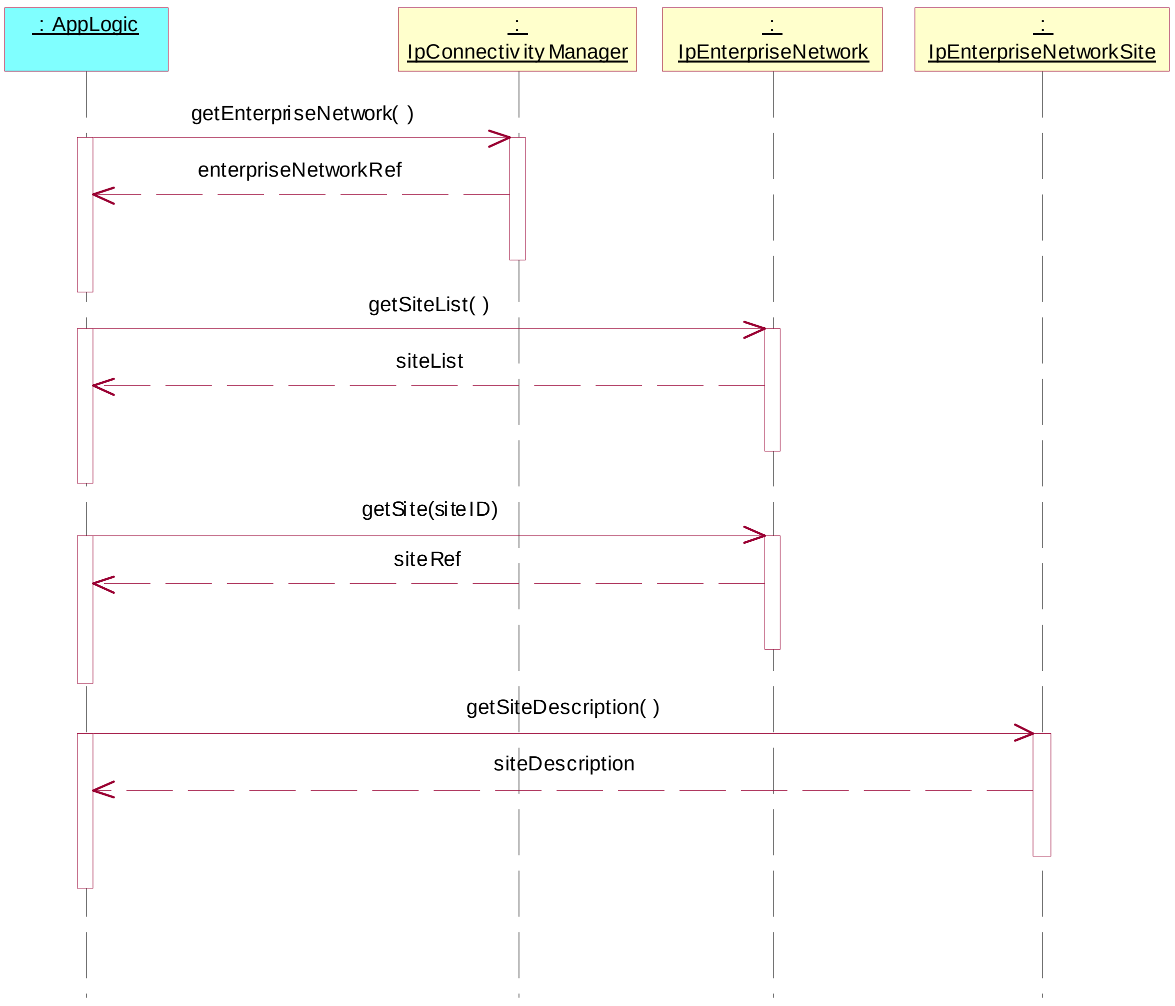
Test Sequence:

1. Method call **getEnterpriseNetwork()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getSiteList()** on IpEnterpriseNetwork
   Parameters: None
   Check: valid value of TpStringList is returned

3. Method call **getSite()** on IpEnterpriseNetwork
   Parameters: Valid value of siteID, which is an item of the TpStringList, returned in 2.
   Check: valid value of IpInterfaceRef is returned

4. Method call **getSiteDescription()** on IpEnterpriseNetworkSite
   Parameters: None
   Check: valid value of TpString is returned



Test CM_32

Summary: getIpSubnet, successful

Reference: ES 201 915-10 [1], clauses 8.1, 8.2 and 8.3

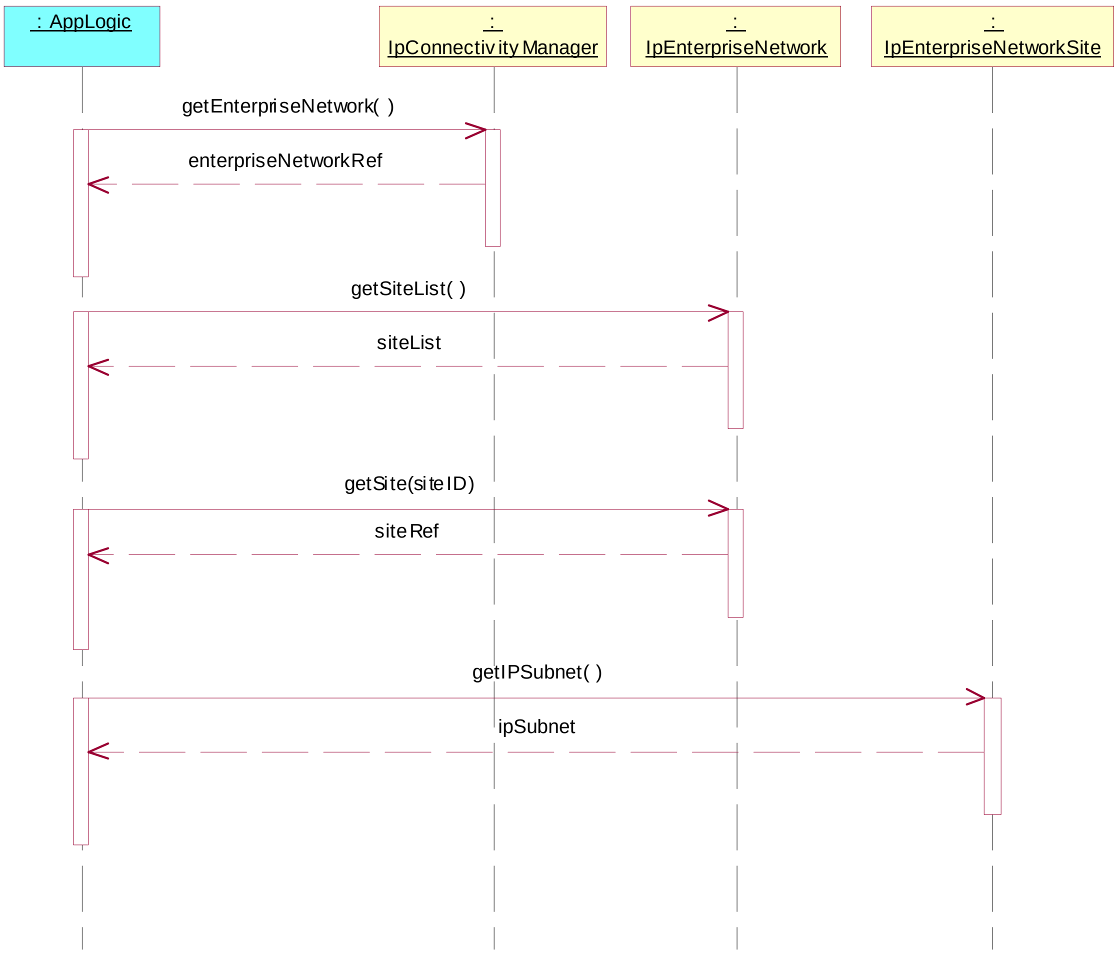
Test Sequence:

1. Method call **getEnterpriseNetwork()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getSiteList()** on IpEnterpriseNetwork
   Parameters: None
   Check: valid value of TpStringList is returned

3. Method call **getSite()** on IpEnterpriseNetwork
   Parameters: Valid value of siteID, which is an item of the TpStringList, returned in 2.
   Check: valid value of IpInterfaceRef is returned

4. Method call **getIpSubnet()** on IpEnterpriseNetworkSite
   Parameters: None
   Check: valid value of TpIpSubnet is returned



Test CM_33

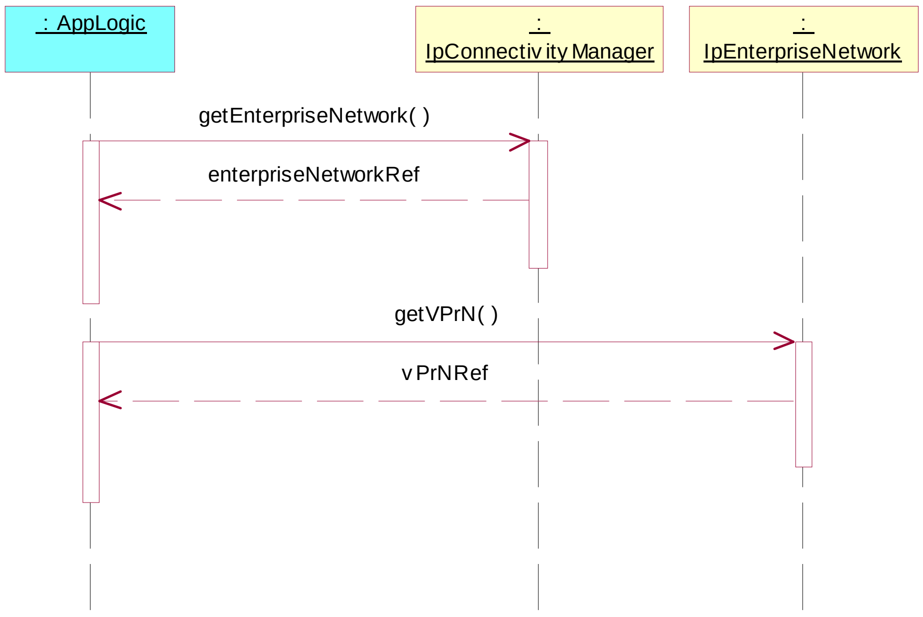
Summary: getVPrN, successful

Reference: ES 201 915-10 [1], clauses 8.1 and 8.2

Test Sequence:

1. Method call **getEnterpriseNetwork()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getVPrN()** on IpEnterpriseNetwork
   Parameters: None
   Check: valid value of IpInterfaceRef is returned



Test CM_34

Summary: getVPrPList, getVPrP, successful

Reference: ES 201 915-10 [1], clauses 8.1, 8.2 and 8.6

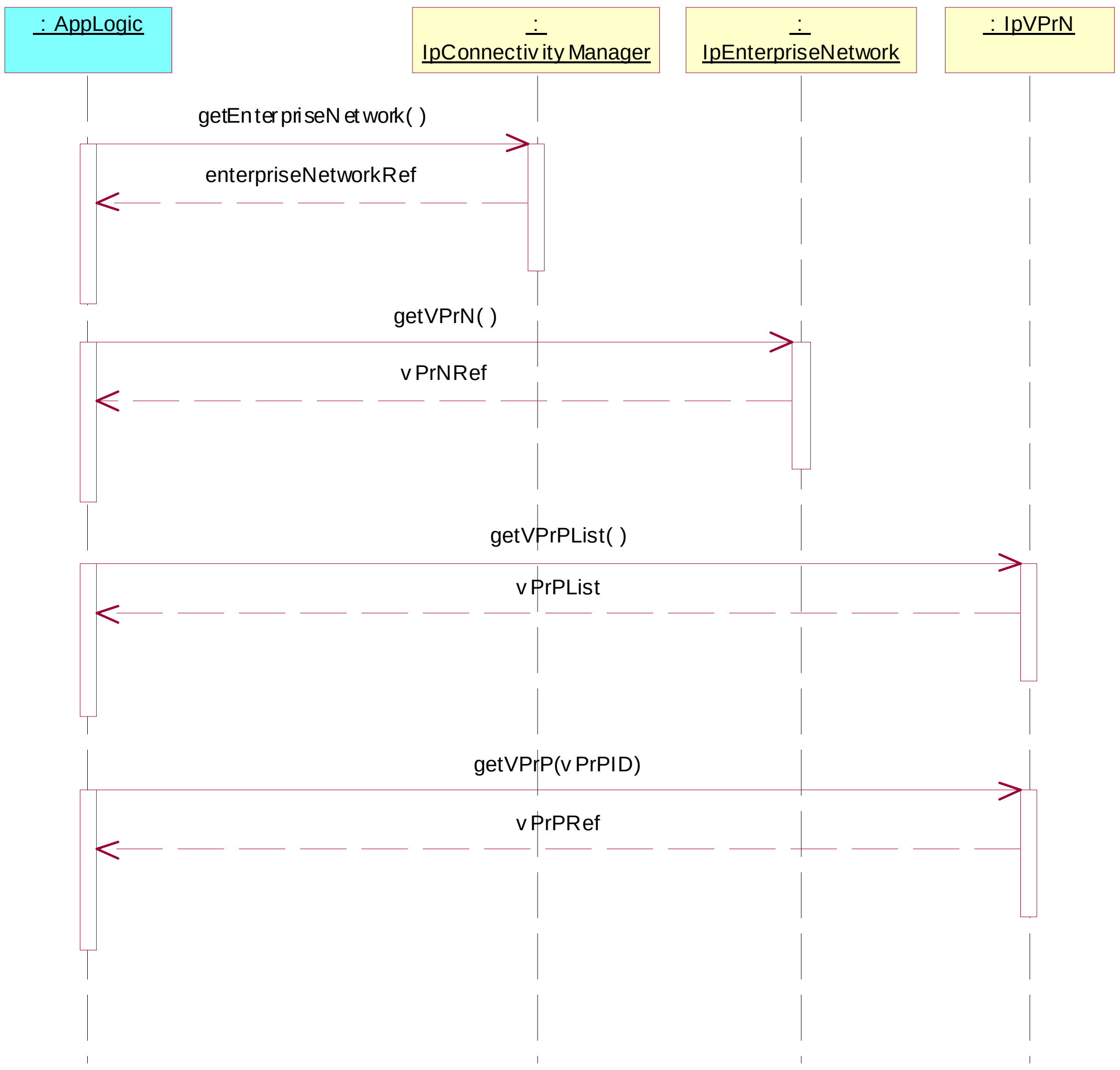
Test Sequence:

1. Method call **getEnterpriseNetwork()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getVPrN()** on IpEnterpriseNetwork
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

3. Method call **getVPrPList()** on IpVPrN
   Parameters: None
   Check: valid value of TpStringList is returned

4. Method call **getVPrP()** on IpVPrN
   Parameters: Valid value of vPrPID, which is an item of the TpStringList, returned in 3.
   Check: valid value of IpInterfaceRef is returned



Test CM_35

Summary: getVPrP, P_UNKNOWN_VPRP_ID

Reference: ES 201 915-10 [1], clauses 8.1, 8.2 and 8.6

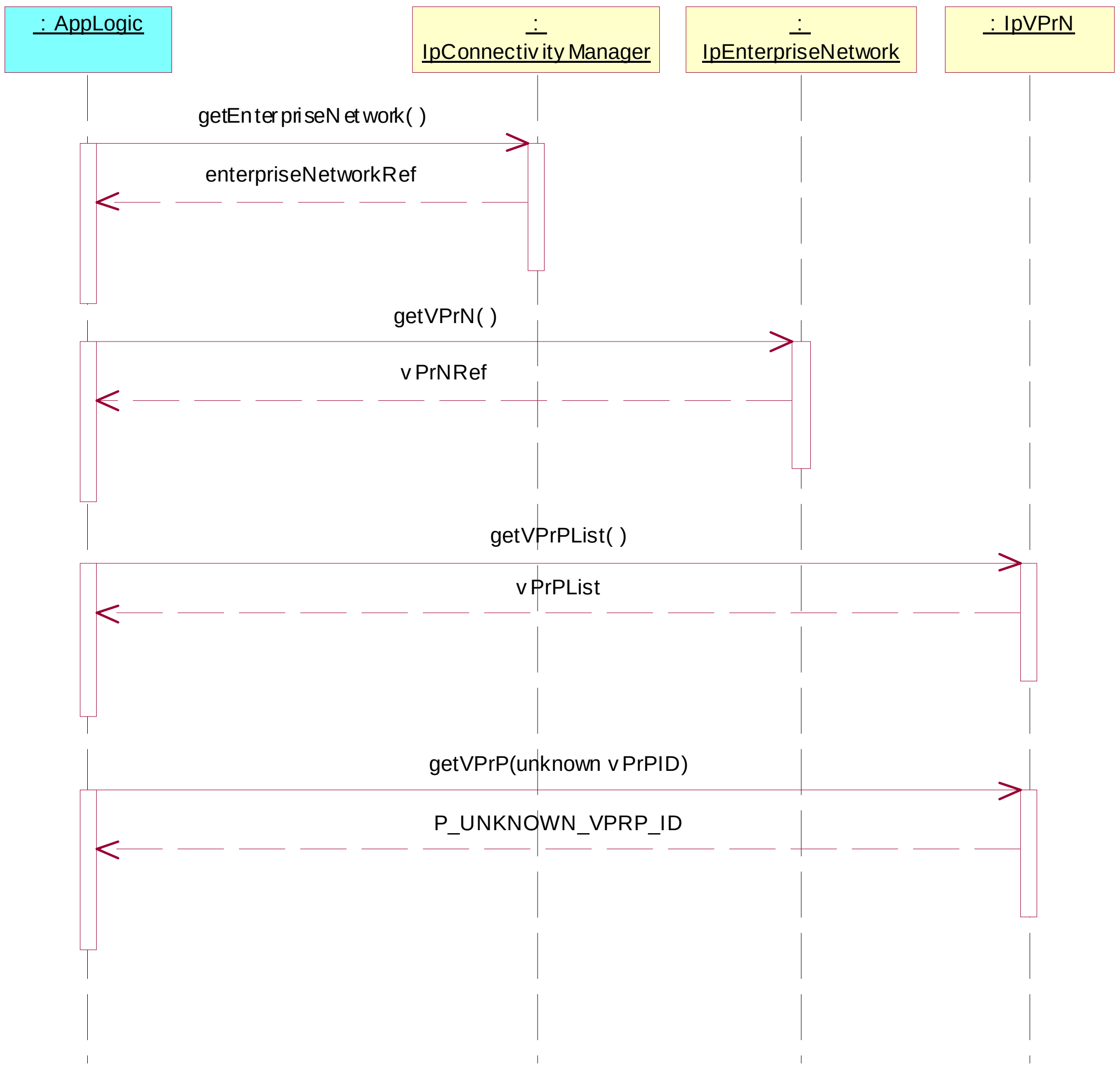
Test Sequence:

1. Method call **getEnterpriseNetwork()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getVPrN()** on IpEnterpriseNetwork
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

3. Method call **getVPrPList()** on IpVPrN
   Parameters: None
   Check: valid value of TpStringList is returned

4. Method call **getVPrP()** on IpVPrN
   Parameters: Valid value of vPrPID, which is not an item of the TpStringList, returned in 3.
   Check: P_UNKNOWN_VPRP_ID is returned



Test CM_36

Summary: getVPrP, P_ILLEGAL_VPRPID

Reference: ES 201 915-10 [1], clauses 8.1, 8.2 and 8.6

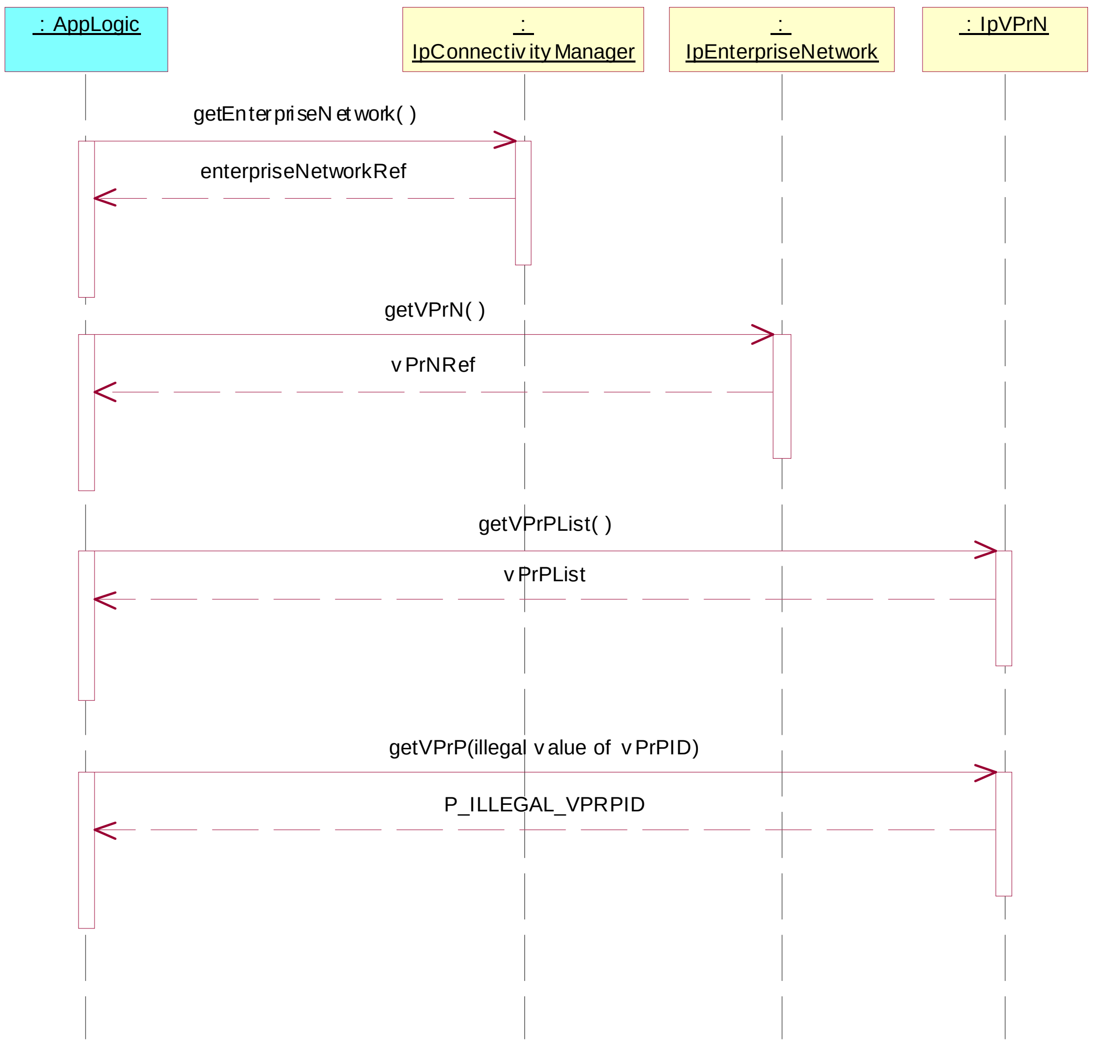
Test Sequence:

1. Method call **getEnterpriseNetwork()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getVPrN()** on IpEnterpriseNetwork
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

3. Method call **getVPrPList()** on IpVPrN
   Parameters: None
   Check: valid value of TpStringList is returned

4. Method call **getVPrP()** on IpVPrN
   Parameters: Invalid value of vPrPID
   Check: P_ILLEGAL_VPRPID is returned



Test CM_37

Summary: getVPrPID, successful

Reference: ES 201 915-10 [1], clauses 8.1, 8.2, 8.6 and 8.7

Test Sequence:

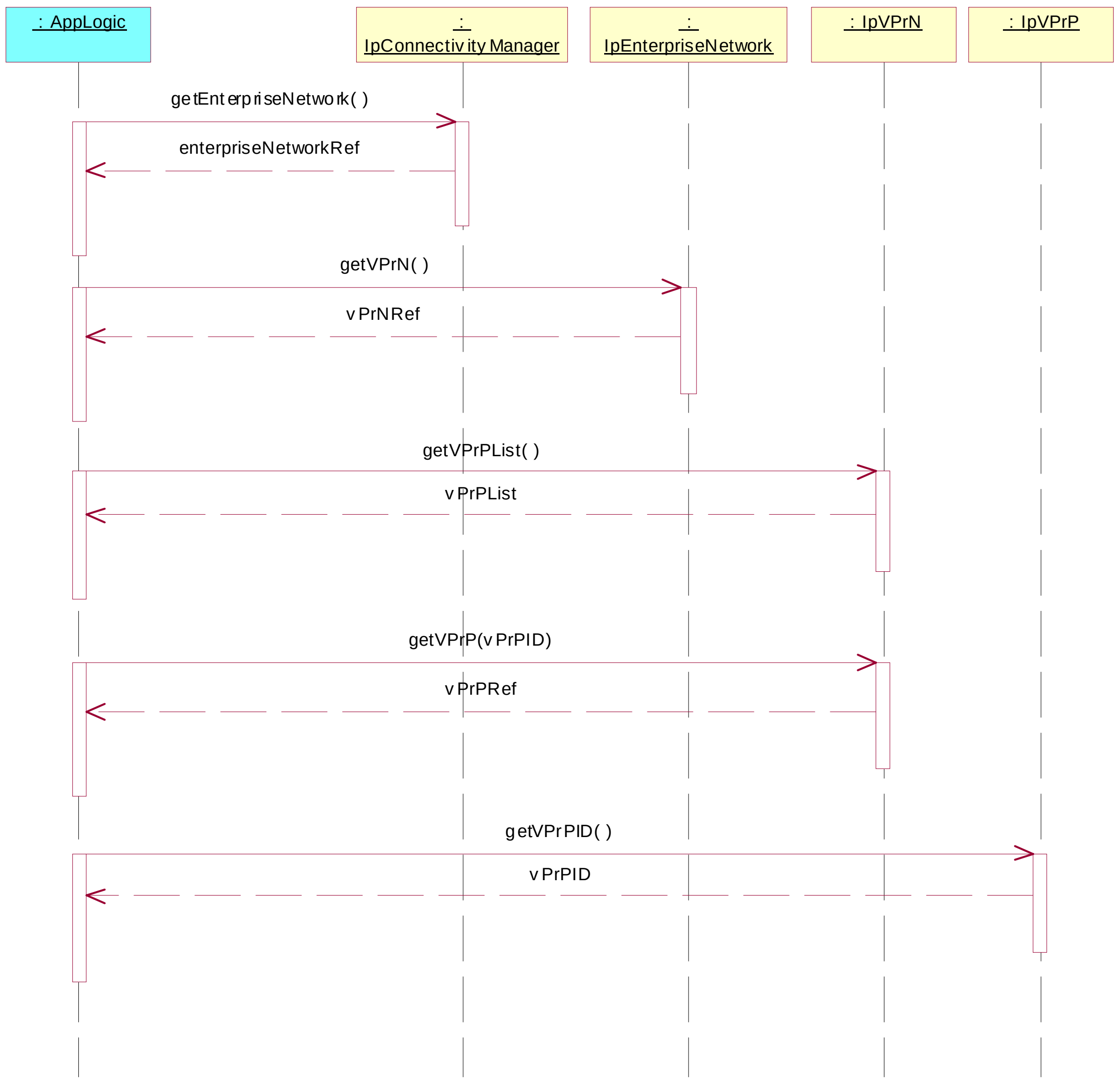
1. Method call **getEnterpriseNetwork()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getVPrN()** on IpEnterpriseNetwork
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

3. Method call **getVPrPList()** on IpVPrN
   Parameters: None
   Check: valid value of TpStringList is returned

4. Method call **getVPrP()** on IpVPrN
   Parameters: Valid value of vPrPID, which is an item of the TpStringList, returned in 3.
   Check: valid value of IpInterfaceRef is returned

5 Method call **getVPrPID()** on IpVPrP
  Parameters: None
  Check: valid value of TpString is returned



Test CM_38

Summary: getSlaID, successful

Reference: ES 201 915-10 [1], clauses 8.1, 8.2, 8.6 and 8.7

Test Sequence:

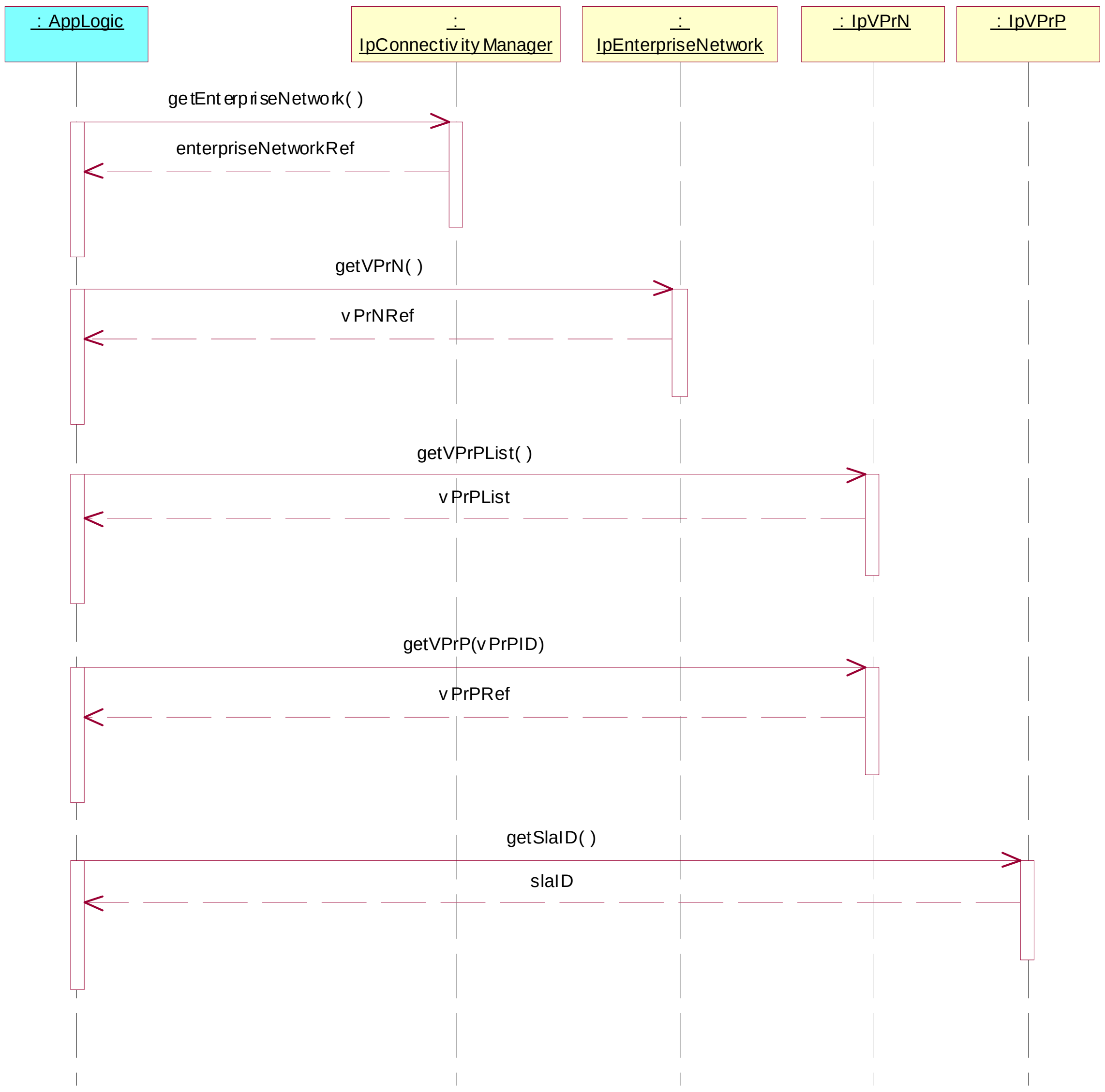
1. Method call **getEnterpriseNetwork()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getVPrN()** on IpEnterpriseNetwork
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

3. Method call **getVPrPList()** on IpVPrN
   Parameters: None
   Check: valid value of TpStringList is returned

4. Method call **getVPrP()** on IpVPrN
   Parameters: Valid value of vPrPID, which is an item of the TpStringList, returned in 3.
   Check: valid value of IpInterfaceRef is returned

5 Method call **getSlaID()** on IpVPrP
  Parameters: None
  Check: valid value of TpString is returned


: AppLogic
: IpConnectivityManager
: IpEnterpriseNetwork
: IpVPrN
: IpVPrP
getEnterpriseNetwork( )
enterpriseNetworkRef
getVPrN( )
vPrNRef
getVPrPList( )
vPrPList
getVPrP(vPrPID)
vPrPRef
getSlaID( )
slaID

Test CM_39

Summary: getStatus, successful

Reference: ES 201 915-10 [1], clauses 8.1, 8.2, 8.6 and 8.7

Test Sequence:

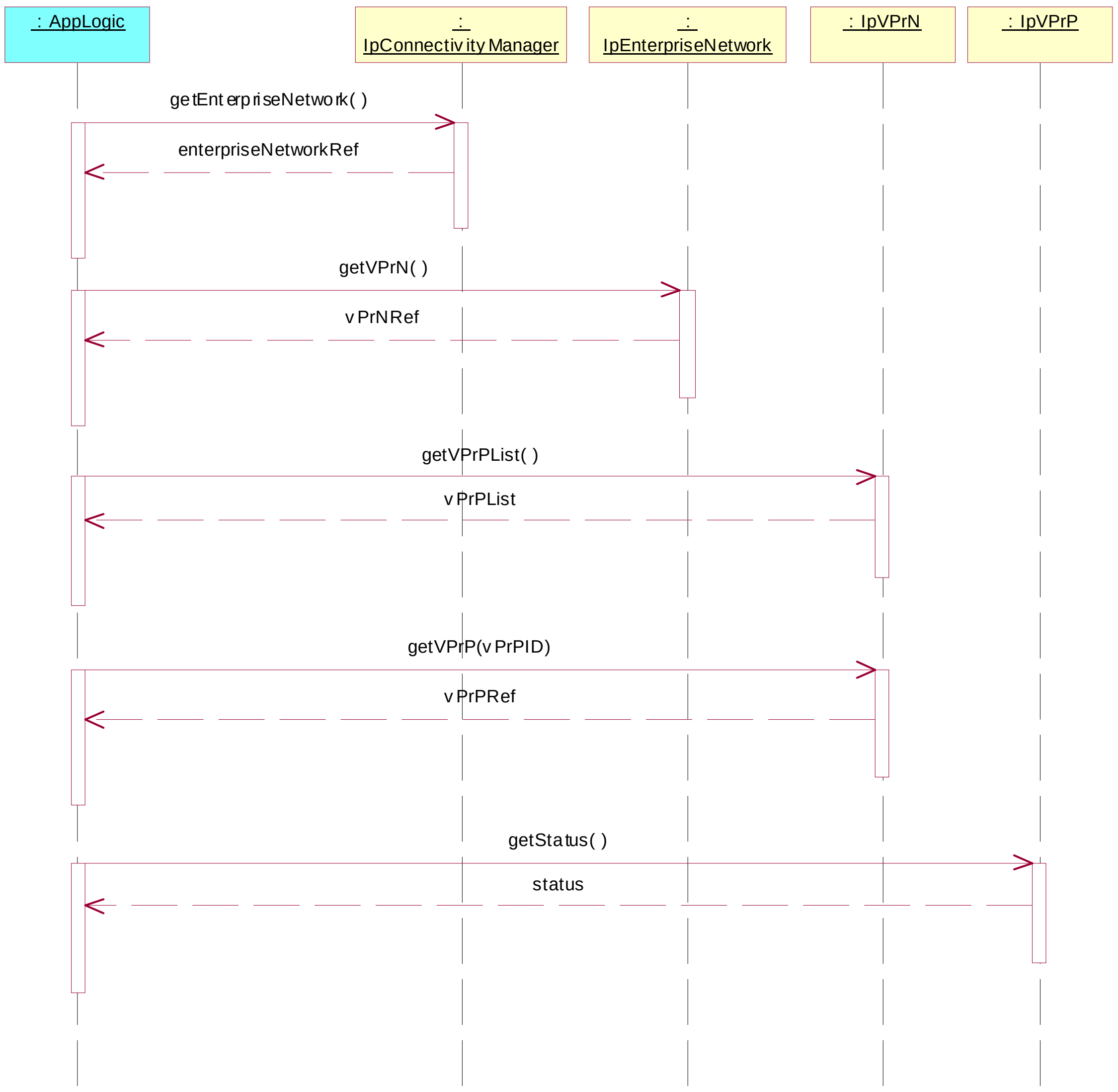
1. Method call **getEnterpriseNetwork()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getVPrN()** on IpEnterpriseNetwork
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

3. Method call **getVPrPList()** on IpVPrN
   Parameters: None
   Check: valid value of TpStringList is returned

4. Method call **getVPrP()** on IpVPrN
   Parameters: Valid value of vPrPID, which is an item of the TpStringList, returned in 3.
   Check: valid value of IpInterfaceRef is returned

5 Method call **getStatus()** on IpVPrP
   Parameters: None
   Check: valid value of TpVprpStatus is returned



Test CM_40

Summary: getProvisionedQoSInfo, successful

Reference: ES 201 915-10 [1], clauses 8.1, 8.2, 8.6 and 8.7

Test Sequence:

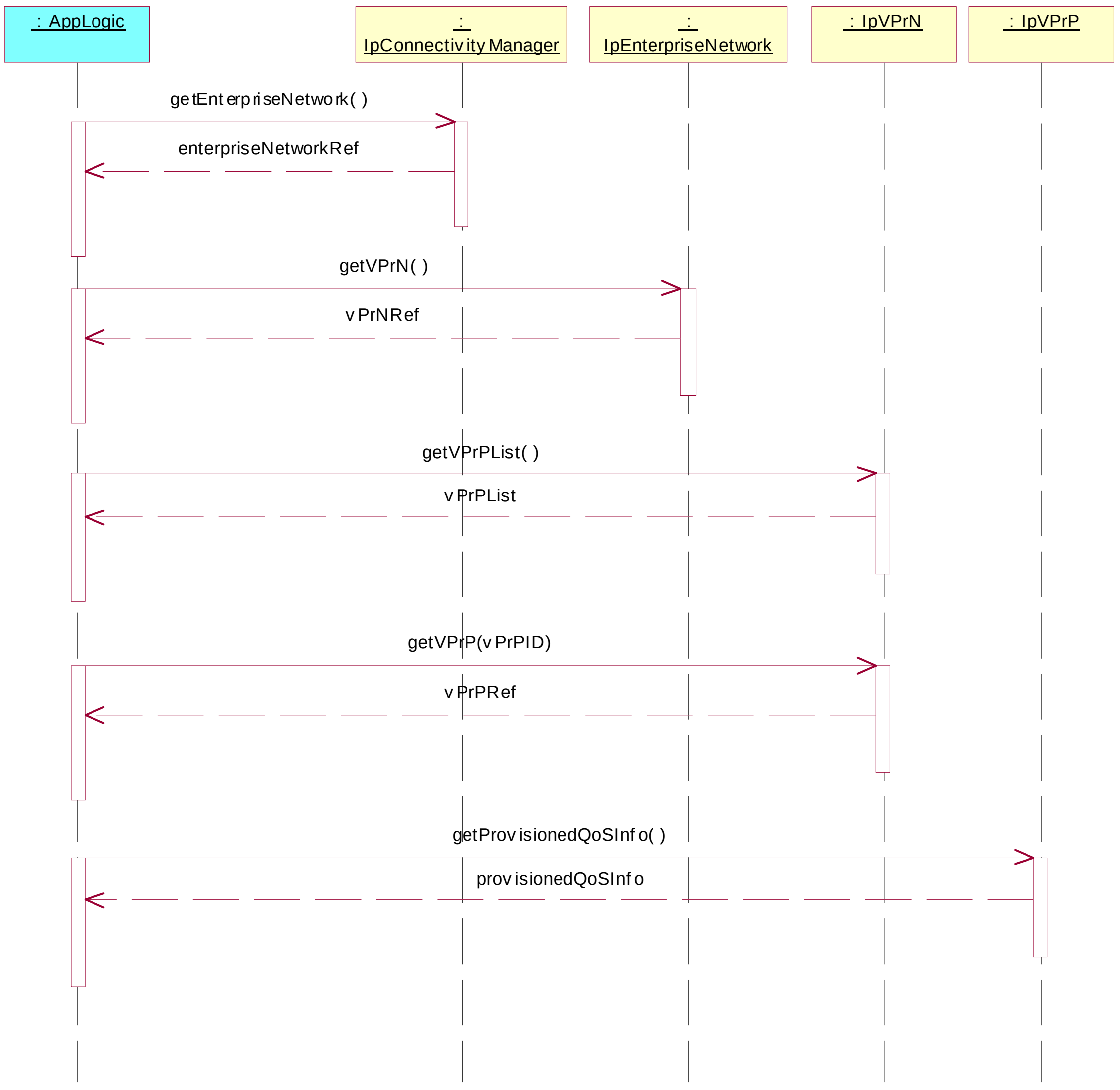
1. Method call **getEnterpriseNetwork()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getVPrN()** on IpEnterpriseNetwork
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

3. Method call **getVPrPList()** on IpVPrN
   Parameters: None
   Check: valid value of TpStringList is returned

4. Method call **getVPrP()** on IpVPrN
   Parameters: Valid value of vPrPID, which is an item of the TpStringList, returned in 3.
   Check: valid value of IpInterfaceRef is returned

5 Method call **getProvisionedQoSInfo()** on IpVPrP
   Parameters: None
   Check: valid value of TpProvisionedQoSInfo is returned



Test CM_41

Summary: getValidityInfo, successful

Reference: ES 201 915-10 [1], clauses 8.1, 8.2, 8.6 and 8.7

Test Sequence:

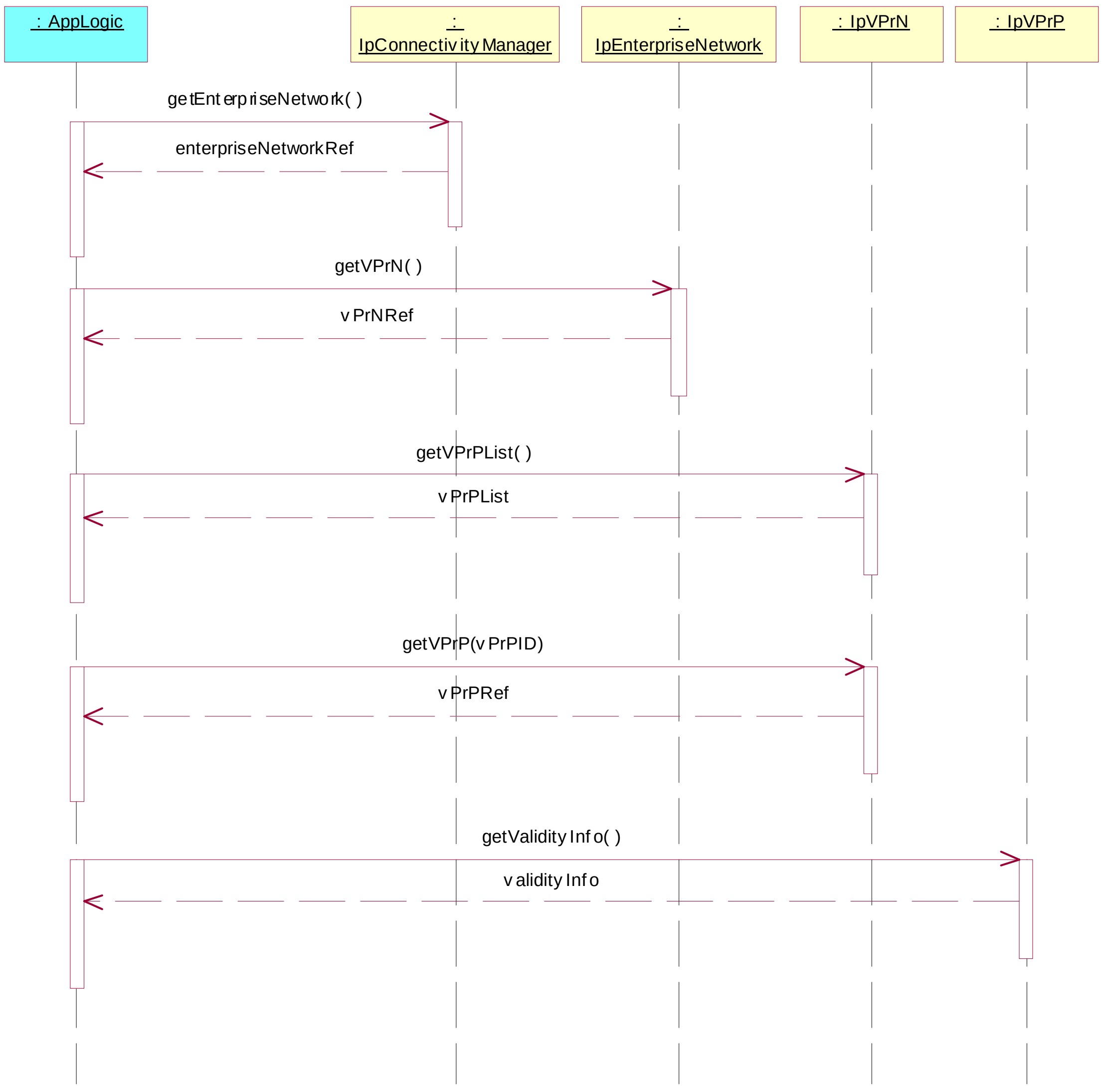
1. Method call **getEnterpriseNetwork()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getVPrN()** on IpEnterpriseNetwork
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

3. Method call **getVPrPList()** on IpVPrN
   Parameters: None
   Check: valid value of TpStringList is returned

4. Method call **getVPrP()** on IpVPrN
   Parameters: Valid value of vPrPID, which is an item of the TpStringList, returned in 3.
   Check: valid value of IpInterfaceRef is returned

5 Method call **getValidityInfo()** on IpVPrP
   Parameters: None
   Check: valid value of TpValidityInfo is returned



Test CM_42

Summary: getPipeQoSInfo, successful

Reference: ES 201 915-10 [1], clauses 8.1, 8.2, 8.6 and 8.7

Test Sequence:

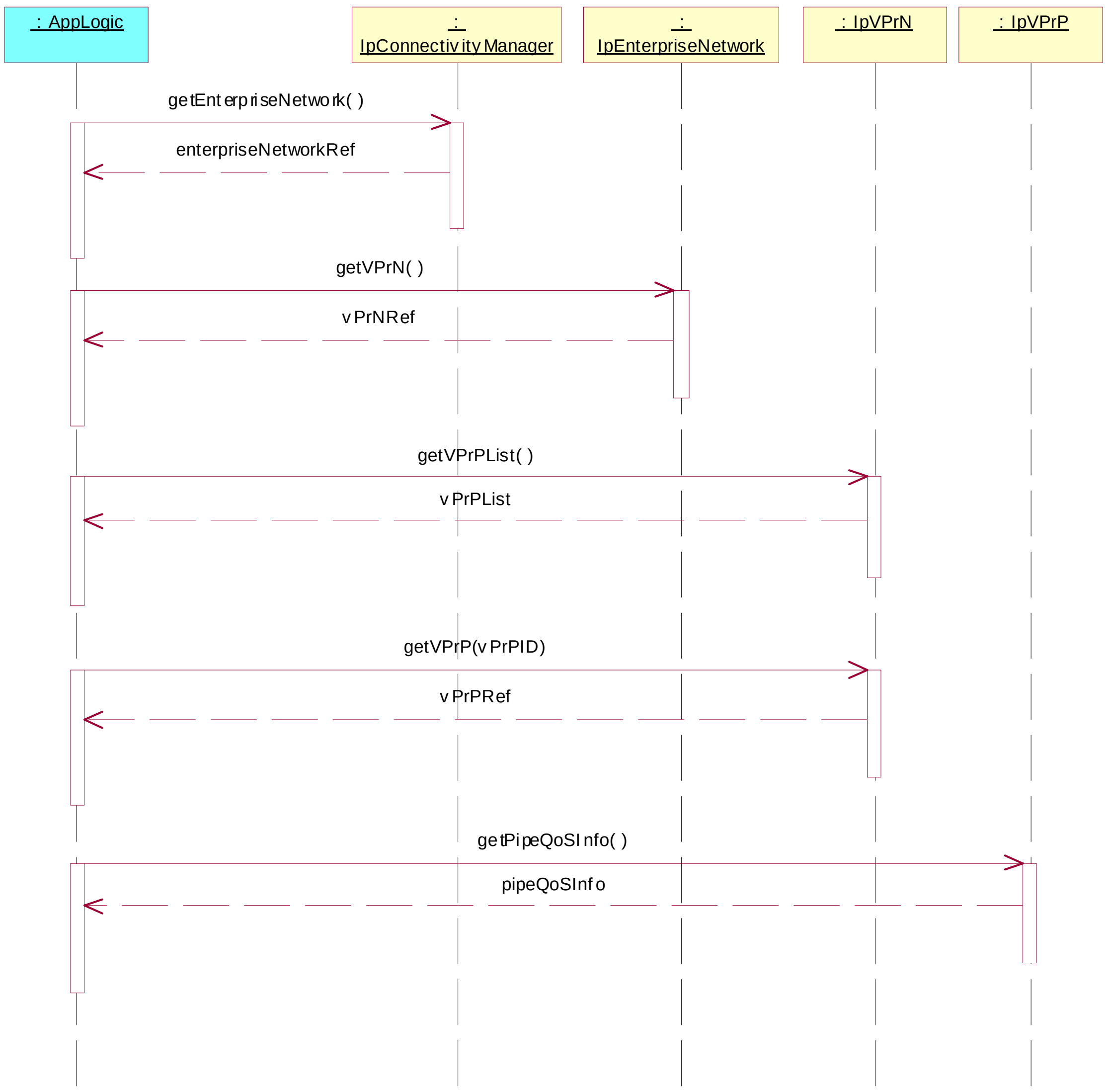
1. Method call **getEnterpriseNetwork()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getVPrN()** on IpEnterpriseNetwork
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

3. Method call **getVPrPList()** on IpVPrN
   Parameters: None
   Check: valid value of TpStringList is returned

4. Method call **getVPrP()** on IpVPrN
   Parameters: Valid value of vPrPID, which is an item of the TpStringList, returned in 3.
   Check: valid value of IpInterfaceRef is returned

5 Method call **getPipeQoSInfo()** on IpVPrP
   Parameters: None
   Check: valid value of TpPipeQoSInfo is returned



Test CM_43

Summary: getDsCodePoint, successful

Reference: ES 201 915-10 [1], clauses 8.1, 8.2, 8.6 and 8.7

Test Sequence:

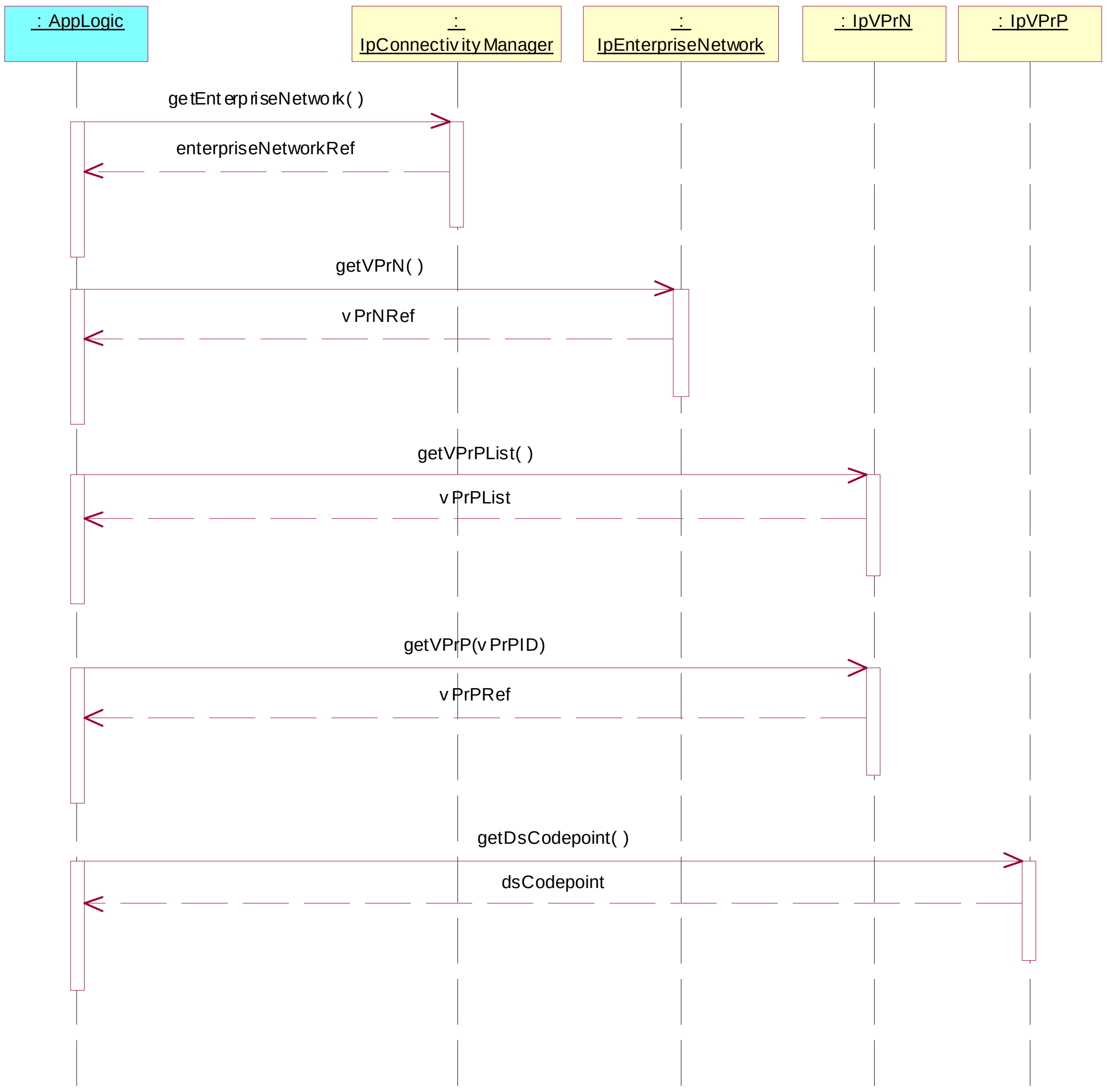
1. Method call **getEnterpriseNetwork()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getVPrN()** on IpEnterpriseNetwork
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

3. Method call **getVPrPList()** on IpVPrN
   Parameters: None
   Check: valid value of TpStringList is returned

4. Method call **getVPrP()** on IpVPrN
   Parameters: Valid value of vPrPID, which is an item of the TpStringList, returned in 3.
   Check: valid value of IpInterfaceRef is returned

5 Method call **getDsCodePoint()** on IpVPrP
   Parameters: None
   Check: valid value of TpDsCodePoint is returned



Test CM_44

Summary: createVPrP, deleteVPrP, successful

Reference: ES 201 915-10 [1], clauses 8.1, 8.2, 8.4, 8.6 and 8.7

Test Sequence:

1. Method call **getEnterpriseNetwork()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getVPrN()** on IpEnterpriseNetwork
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

3. Method call **getQoSMenu()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

4. Method call **getTemplateList()** on IpQoSMenu
   Parameters: None
   Check: valid value of TpStringList is returned

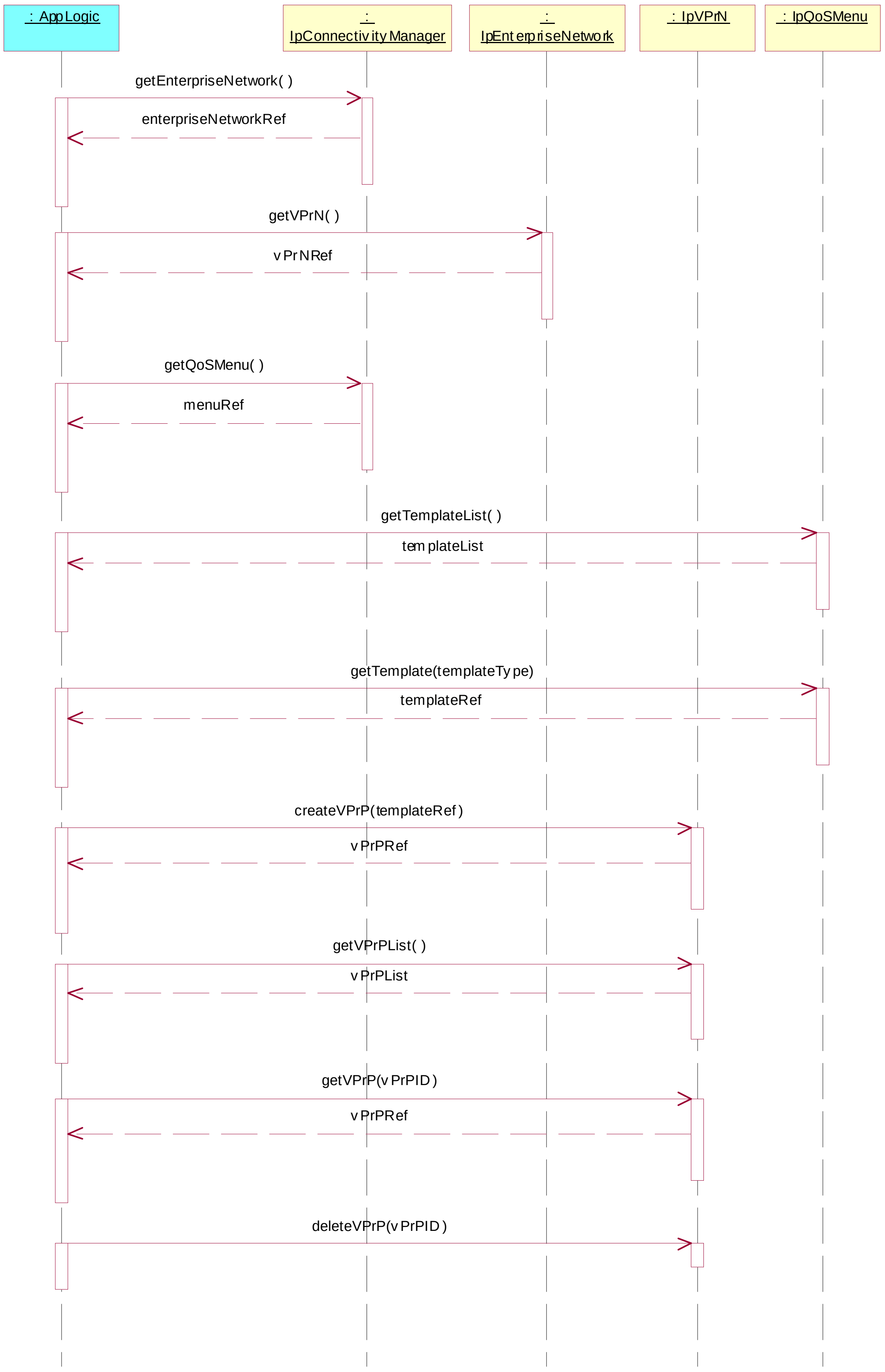
5. Method call **getTemplate()** on IpQoSMenu
   Parameters: Valid value of templateType, which is an item of the TpStringList, returned in 2.
   Check: valid value of IpInterfaceRef is returned

6. Method call **createVPrP()** on IpVPrN
   Parameters: IpInterfaceRef returned in step 5.
   Check: valid value of IpInterfaceRef is returned

7. Method call **getVPrPList()** on IpVPrN
   Parameters: None.
   Check: valid value of TpStringList is returned, in the list the new VprP interface must be found

8. Method call **getVPrP()** on IpVPrN
   Parameters: Valid value of vPrPID, which is the item of the TpStringList, returned in 7 corresponding with VpRP created in step 6.
   Check: valid value of IpInterfaceRef is returned with values matches IpInterfaceRef returned in step 5.

9. Method call **deleteVPrP()** on IpVPrN
   Parameters: Valid value of vPrPID, which is the item of the TpStringList, returned in 7 corresponding with VpRP created in step 6.
   Check: No exception is returned.


: AppLogic
: IpConnectivityManager
: IpEnterpriseNetwork
: IpVPrN
: IpQoSMenu
getEnterpriseNetwork( )
enterpriseNetworkRef
getVPrN( )
vPrNRef
getQoSMenu( )
menuRef
getTemplateList( )
templateList
getTemplate(templateType)
templateRef
createVPrP(templateRef)
vPrPRef
getVPrPList( )
vPrPList
getVPrP(vPrPID)
vPrPRef
deleteVPrP(vPrPID)

Test CM_45

Summary: createVPrP(), P_UNKNOWN_INTERFACE

Reference: ES 201 915-10 [1], clauses 8.1, 8.2, 8.4, 8.6 and 8.7

Test Sequence:

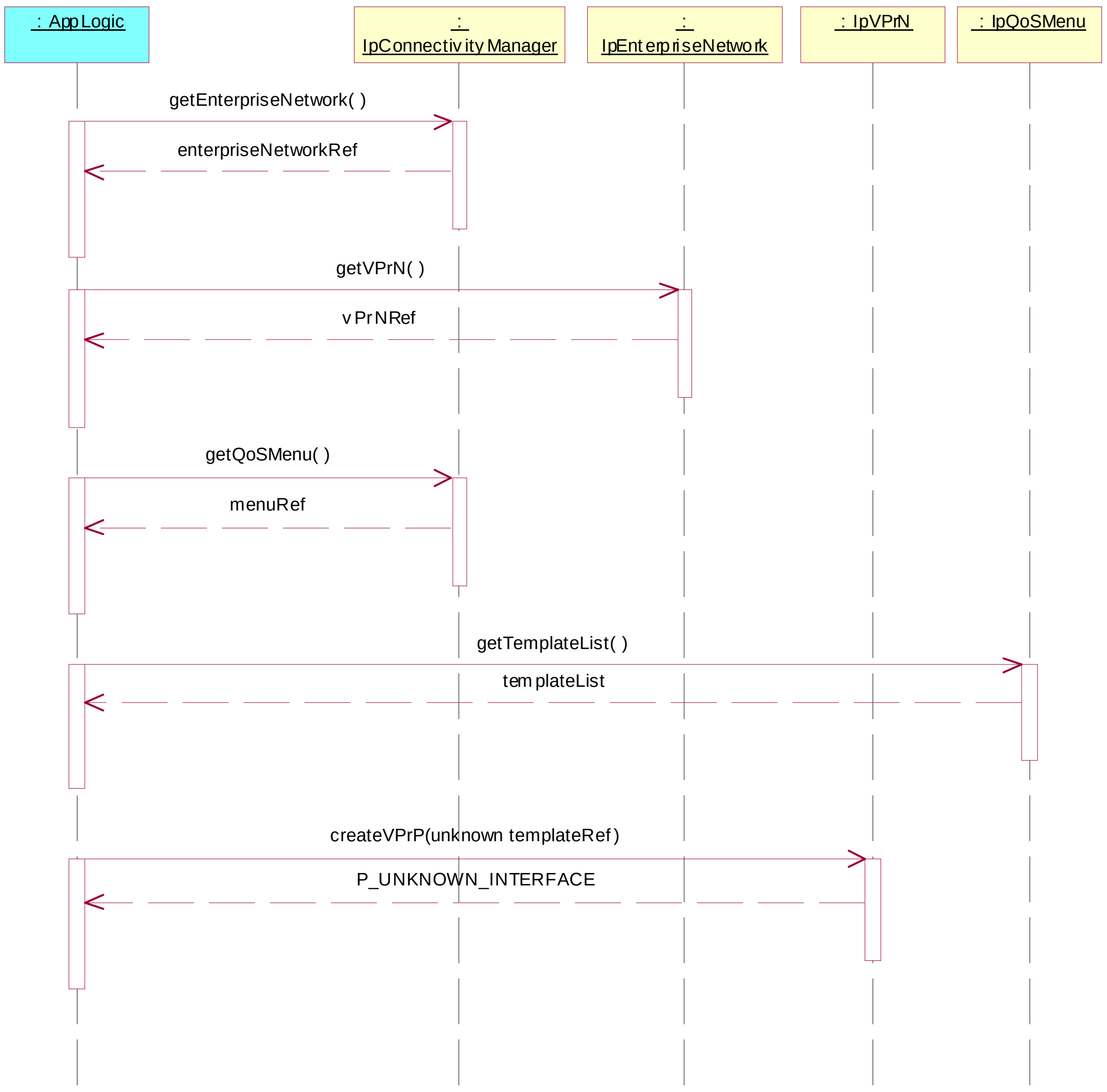
1. Method call **getEnterpriseNetwork()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getVPrN()** on IpEnterpriseNetwork
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

3. Method call **getQoSMenu()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

4. Method call **getTemplateList()** on IpQoSMenu
   Parameters: None
   Check: valid value of TpStringList is returned

6. Method call **createVPrP()** on IpVPrN
   Parameters: IpInterfaceRef, which is not an item of the TpStringList, returned in 4
   Check: P_UNKNOWN_INTERFACE is returned



Test CM_46

Summary: deleteVPrP, P_UNKNOWN_VPRP_ID

Reference: ES 201 915-10 [1], clauses 8.1, 8.2, 8.4, 8.6 and 8.7

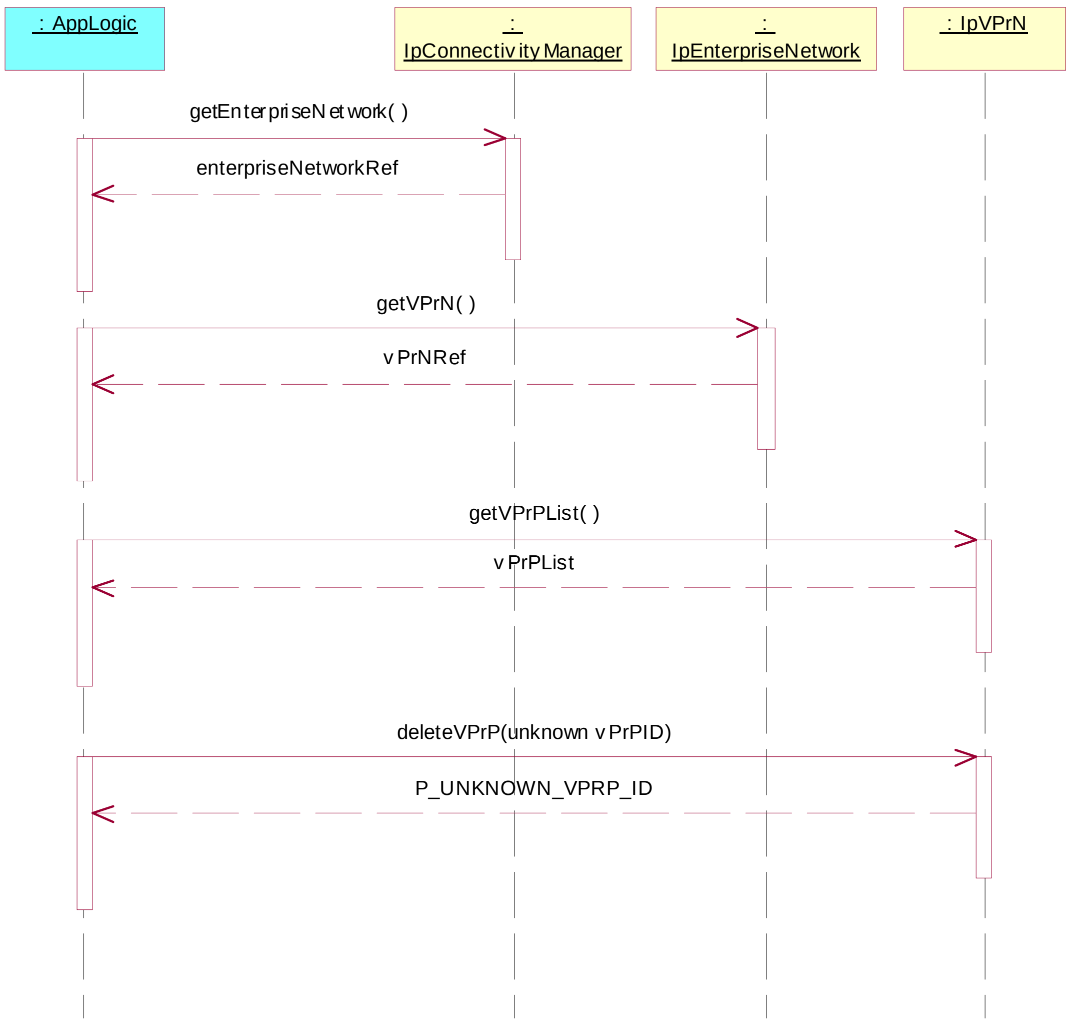
Test Sequence:

1. Method call **getEnterpriseNetwork()** on IpConnectivityManager
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

2. Method call **getVPrN()** on IpEnterpriseNetwork
   Parameters: None
   Check: valid value of IpInterfaceRef is returned

4. Method call **getVPrPList()** on IpVPrN
   Parameters: None.
   Check: valid value of TpStringList is returned, in the list the new VprP interface must be found

5. Method call **deleteVPrP()** on IpVPrN
   Parameters: Valid value of vPrPID, which is not an item of the TpStringList, returned in 3
   Check: P_UNKNOWN_VPRP_ID is returned.



History

Document history		
V1.1.1	August 2003	Publication
V1.2.1	December 2004	Membership Approval Procedure MV 20050211: 2004-12-14 to 2005-02-11