ETSI ES 202 388-13 V1.1.1 (2005-03)

ETSI Standard

**Open Service Access (OSA);
Application Programming Interface (API);
Test Suite Structure and Test Purposes (TSS&TP);
Part 13: Policy Management SCF
(Parlay 4)**

Reference

DES/TISPAN-06004-13-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).

The present document is part 13 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 Policy Management SCF of the Application Programming Interface (API) for Open Service Access (OSA) defined in ES 202 915-13 [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 202 915-13: "Open Service Access (OSA); Application Programming Interface (API); Part 13: Policy Management SCF (Parlay 4)".

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

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

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

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

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

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in ES 202 915-13 [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
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
PM	Policy Management
TP	Test Purpose
TSS	Test Suite Structure

4 Test Suite Structure (TSS)

- Policy Management SCF

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:	"PM" for **P**olicy **M**anagement 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 202 915-13 [1].

5.1.3 Test strategy

As the base standard ES 202 915-13 [1] contains no explicit requirements for testing, the TPs were generated as a result of an analysis of the base standard and the PICS 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 Policy Management SCF

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

All parameters specified in method calls are valid unless specified.

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

5.2.1 Policy Management, SCF side

5.2.1.1 IpPolicyManager

Test PM_PM_01

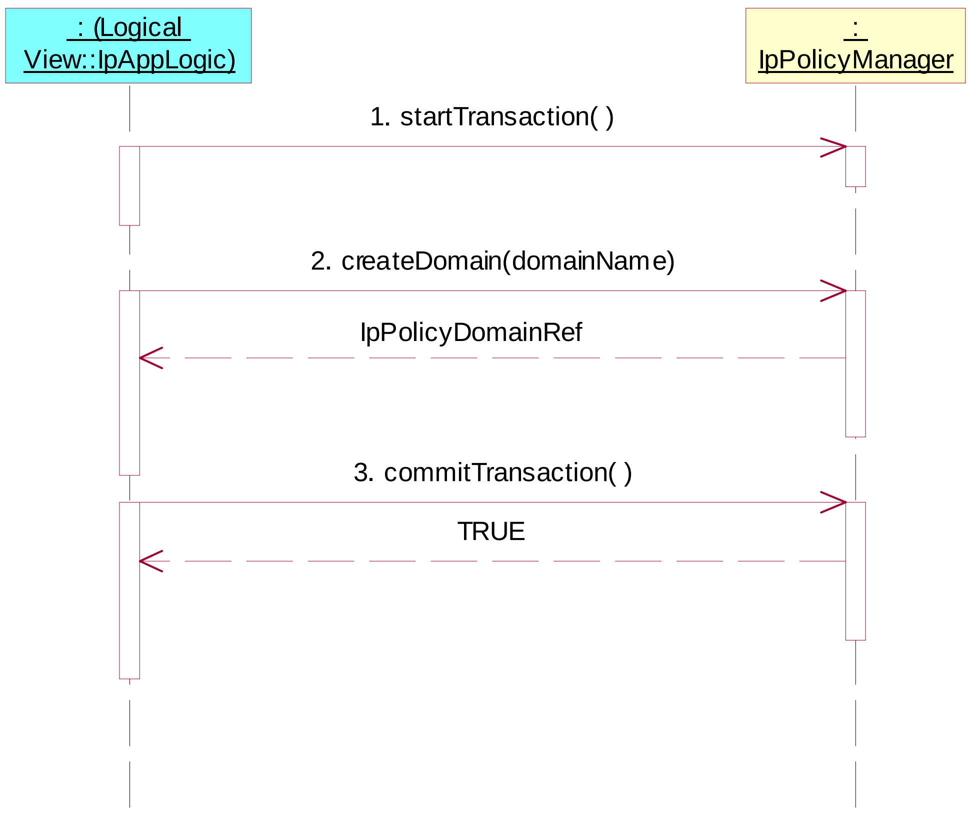
Summary:	create domain.
Reference:	ES 202 915-13 [1], clause 8.1.
Precondition:	**createDomain()** implemented.
Preamble:	Registration of the IUT (Policy Management Control SCF) and the tester (application) to the framework. The tester must have obtained a reference to an instance of the IpPolicyManager interface through selecting that service and signing the required service agreement.

Test Sequence:

1. Method call **startTransaction()**
   Parameters: none
   Check: no exception is returned

2. Method call **createDomain()**
   Parameters: domainName
   Check: valid value of IpPolicyDomainRef is returned

3. Method call **commitTransaction()**
   Parameters: none
   Check: value TRUE is returned



Test PM_PM_02

Summary: get domain.

Reference: ES 202 915-13 [1], clauses 8.1.

Precondition: **getDomain()** implemented.

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

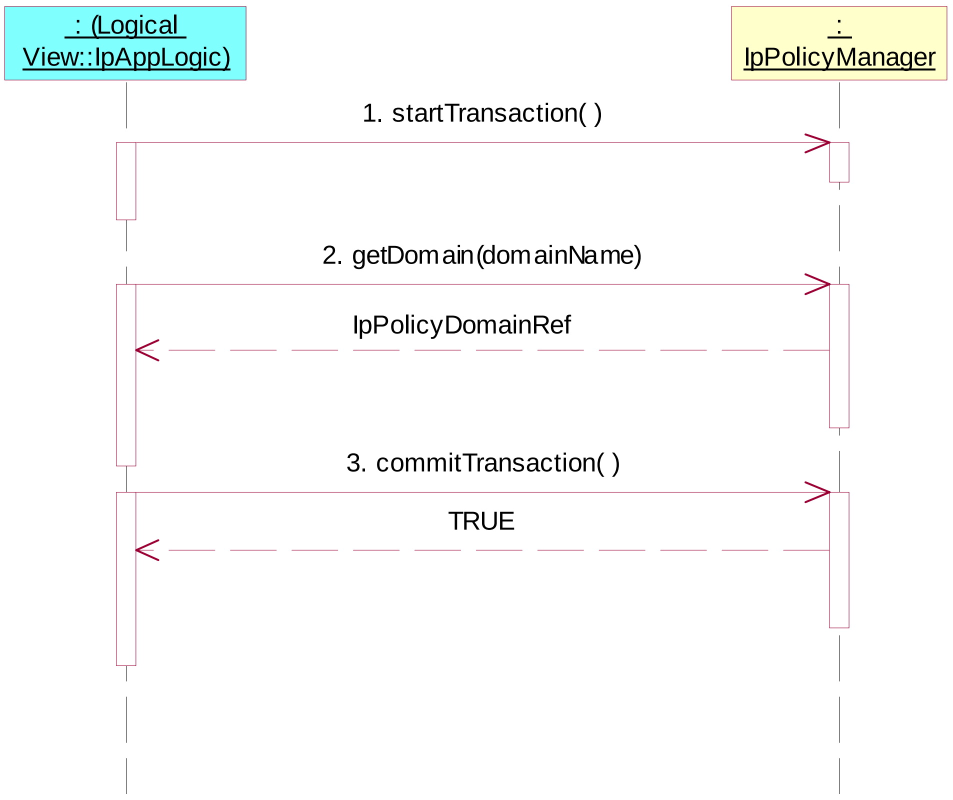
Policy domains have to be present and the tester (application) must be authorized to invoke methods related to them.

Test Sequence:

1. Method call **startTransaction()**
   Parameters: none
   Check: no exception is returned

2. Method call **getDomain()**
   Parameters: domainName
   Check: valid value of IpPolicyDomainRef is returned

3. Method call **commitTransaction()**
   Parameters: none
   Check: value TRUE is returned



Test PM_PM_03

Summary: remove domain.

Reference: ES 202 915-13 [1], clause 8.1.

Precondition: **removeDomain()** implemented.

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

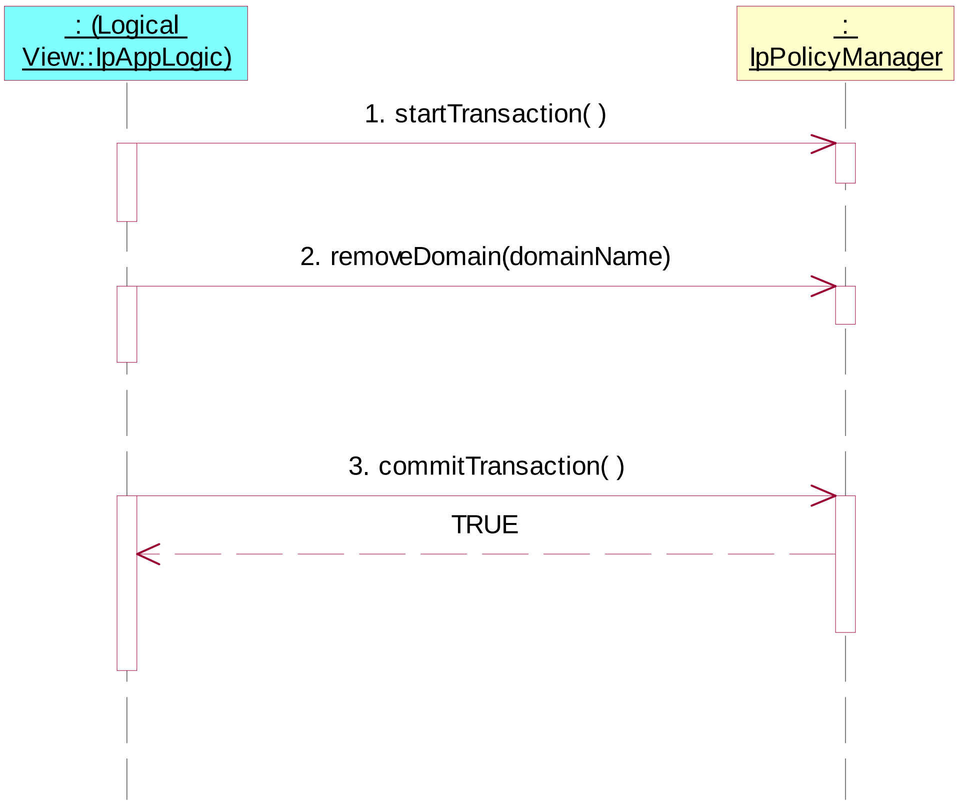
Policy domains have to be present and the tester (application) must be authorized to invoke methods related to them.

Test Sequence:

1. Method call **startTransaction()**
   Parameters: none
   Check: no exception is returned

2. Method call **removeDomain()**
   Parameters: domainName
   Check: no exception is returned

3. Method call **commitTransaction()**
   Parameters: none
   Check: value TRUE is returned



Test PM_PM_04

Summary:	get number of policy domains.
Reference:	ES 202 915-13 [1], clause 8.1.
Precondition:	**getDomainCount()** implemented.
Preamble:	Registration of the IUT (Policy Management Control SCF) and the tester (application) to the framework. The tester must have obtained a reference to an instance of the IpPolicyManager interface through selecting that service and signing the required service agreement.

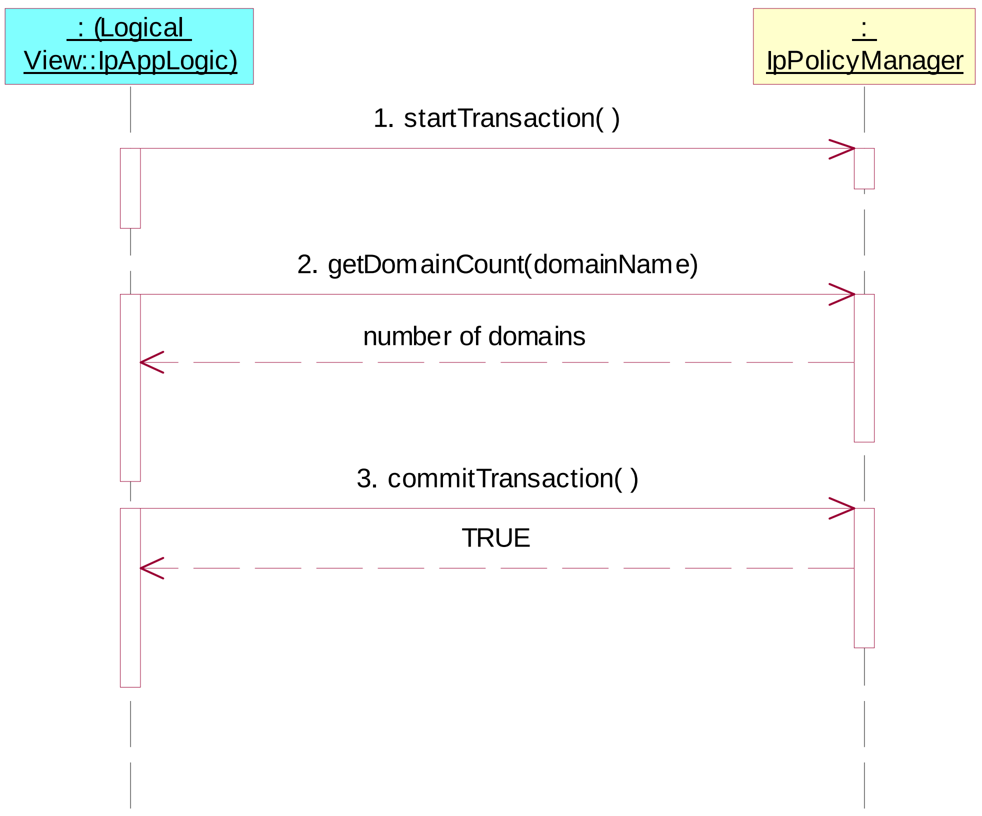
Policy domains have to be present and the tester (application) must be authorized to invoke methods related to them.

Test Sequence:

1. Method call **startTransaction()**
   Parameters: none
   Check: no exception is returned

2. Method call **getDomainCount()**
   Parameters: none
   Check: valid number of domains is returned

3. Method call **commitTransaction()**
   Parameters: none
   Check: value TRUE is returned



Test PM_PM_05

Summary: get reference to policy domain iterator.

Reference: ES 202 915-13 [1], clause 8.1.

Precondition: **getDomainIterator()** implemented.

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

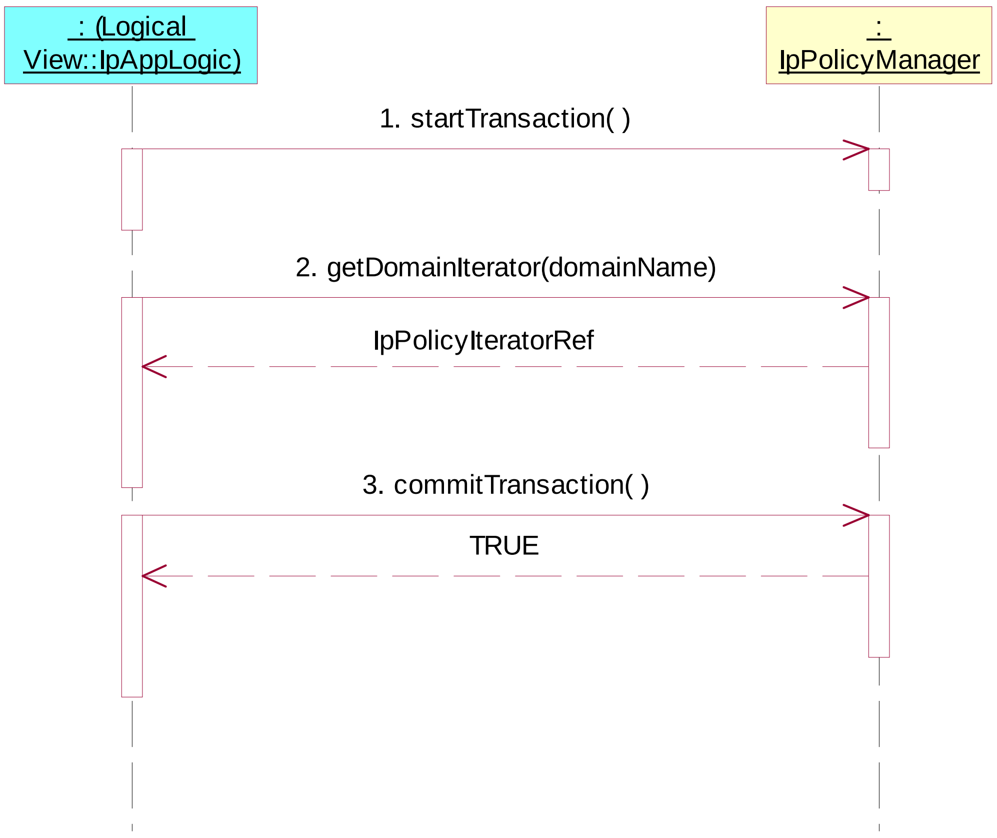
Policy domains have to be present and the tester (application) must be authorized to invoke methods related to them.

Test Sequence:

1. Method call **startTransaction()**
   Parameters: none
   Check: no exception is returned

2. Method call **getDomainIterator()**
   Parameters: none
   Check: valid value of IpPolicyIteratorRef is returned

3. Method call **commitTransaction()**
   Parameters: none
   Check: value TRUE is returned



Test PM_PM_06

Summary: find matching policy domains.

Reference: ES 202 915-13 [1], clause 8.1.

Precondition: **findMatchingDomain()** implemented.

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

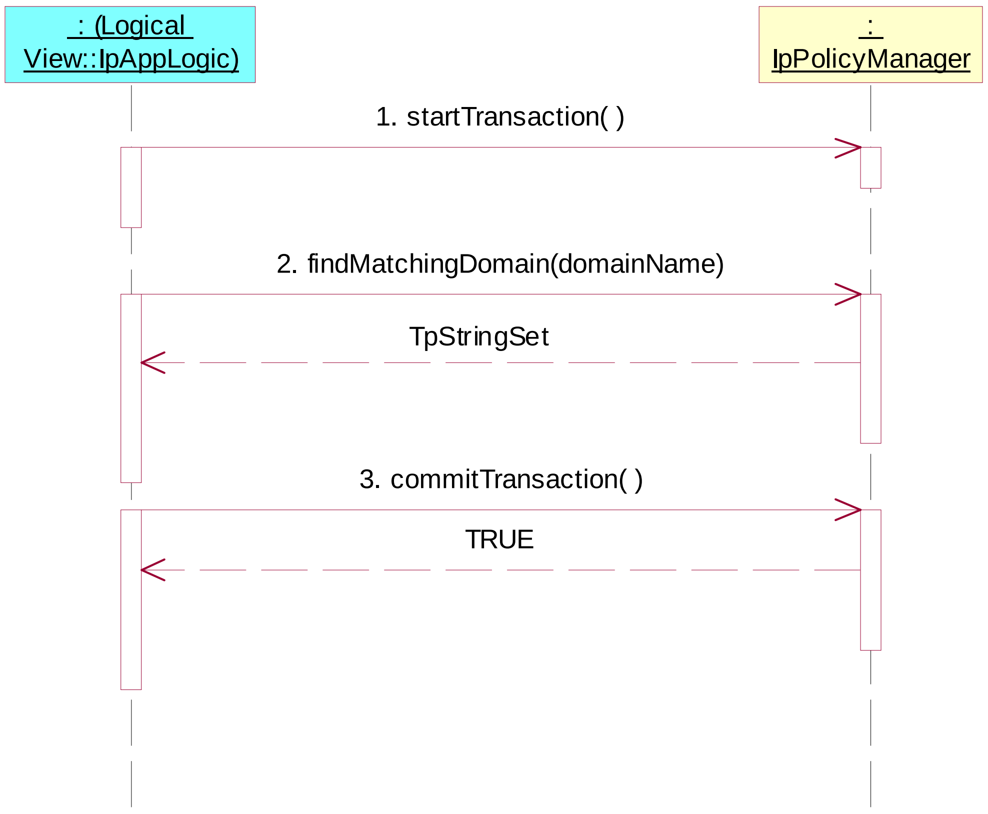
Policy domains have to be present and the tester (application) must be authorized to invoke methods related to them.

Test Sequence:

1. Method call **startTransaction()**
   Parameters: none
   Check: no exception is returned

2. Method call **findMatchingDomain()**
   Parameters: matchingAttributes
   Check: valid TpStringSet is returned

3. Method call **commitTransaction()**
   Parameters: none
   Check: value TRUE is returned



Test PM_PM_07

Summary: create policy repository.

Reference: ES 202 915-13 [1], clause 8.1.

Precondition: **createRepository()** implemented.

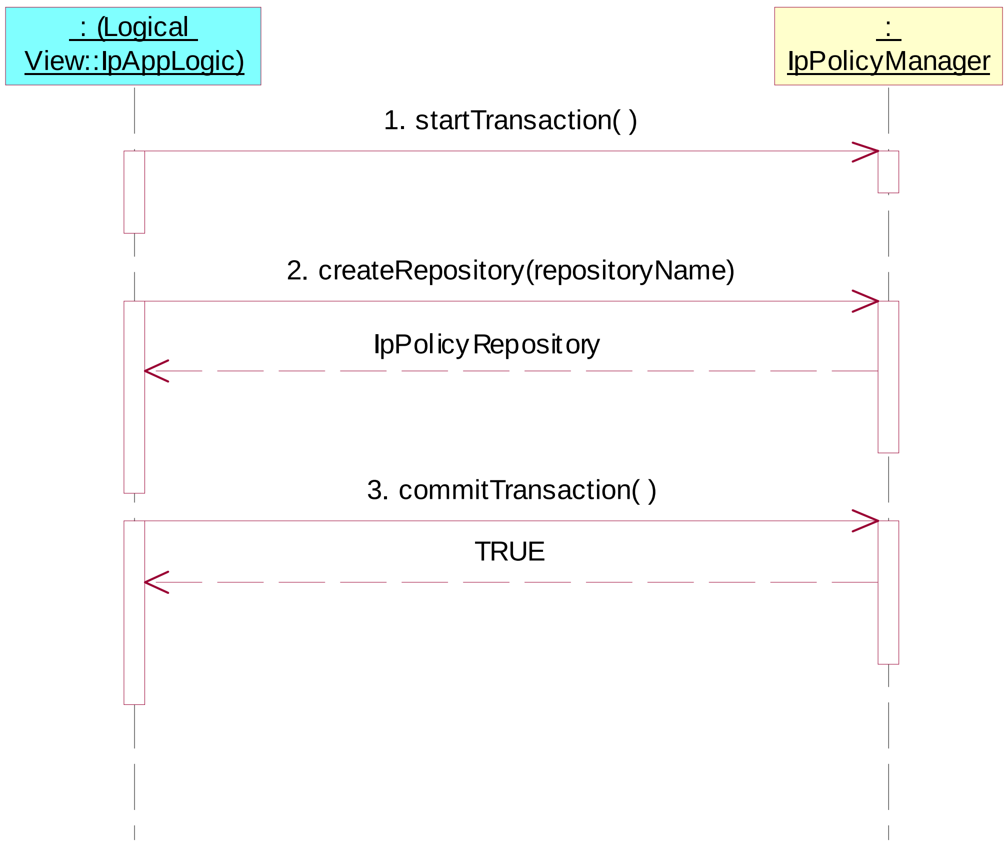
Preamble: Registration of the IUT (Policy Management Control SCF) and the tester (application) to the framework. The tester must have obtained a reference to an instance of the IpPolicyManager interface through selecting that service and signing the required service agreement.

Test Sequence:

1. Method call **startTransaction()**
   Parameters: none
   Check: no exception is returned

2. Method call **createRepository()**
   Parameters: repositoryName
   Check: valid value of IpPolicyRepository is returned

3. Method call **commitTransaction()**
   Parameters: none
   Check: value TRUE is returned



Test PM_PM_08

Summary: get policy repository.

Reference: ES 202 915-13 [1], clause 8.1.

Precondition: **getRepository()** implemented.

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

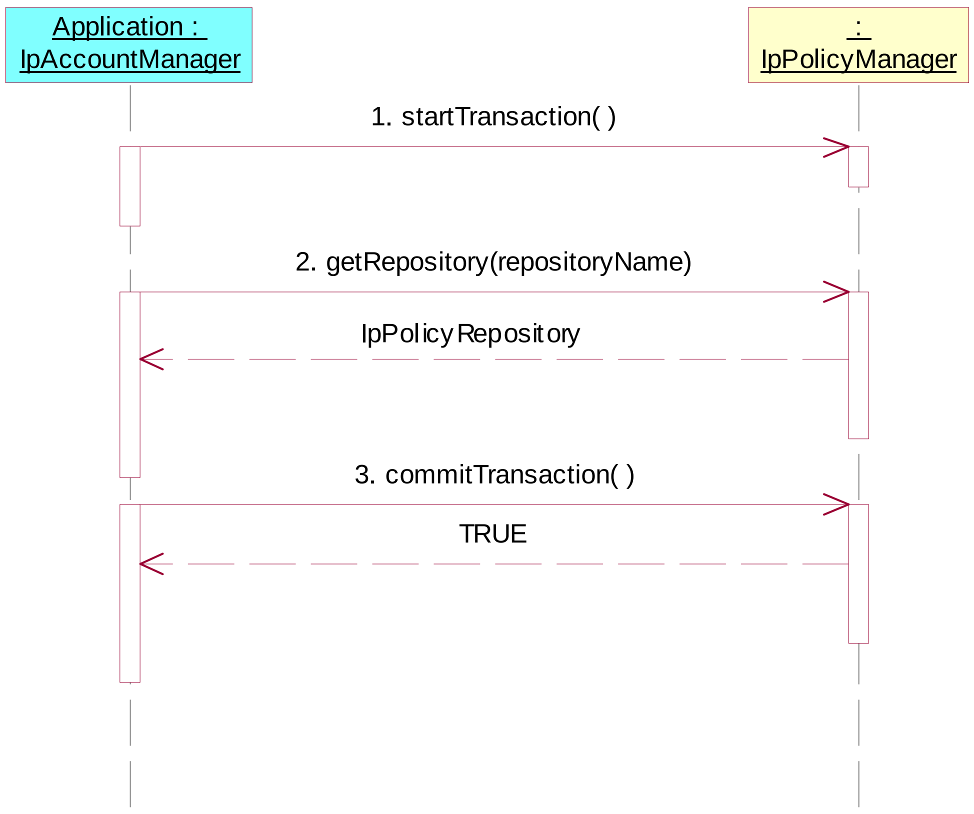
Policy repositories have to be present and the tester (application) must be authorized to invoke methods related to them.

Test Sequence:

1. Method call **startTransaction()**
   Parameters: none
   Check: no exception is returned

2. Method call **getRepository()**
   Parameters: repositoryName
   Check: valid value of IpPolicyRepository is returned

3. Method call **commitTransaction()**
   Parameters: none
   Check: value TRUE is returned



Test PM_PM_09

Summary: remove policy repository.

Reference: ES 202 915-13 [1], clause 8.1.

Precondition: **removeRepository()** implemented.

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

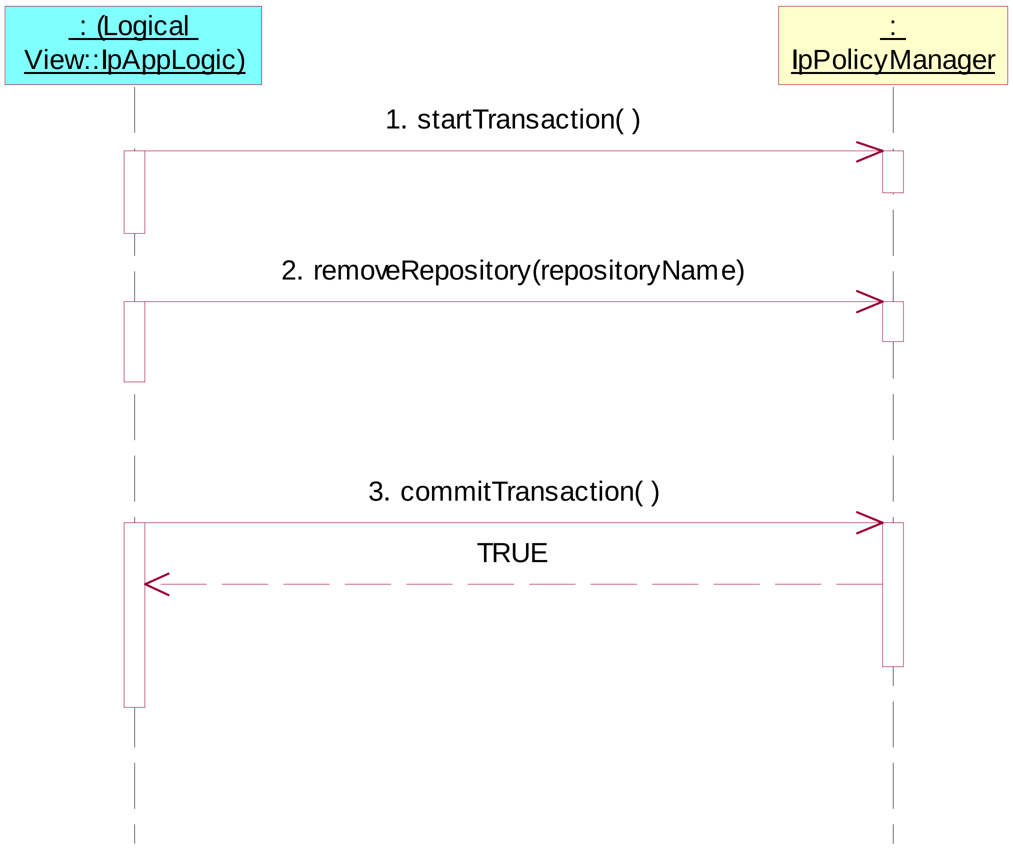
Policy repositories have to be present and the tester (application) must be authorized to invoke methods related to them.

Test Sequence:

1. Method call **startTransaction()**
   Parameters: none
   Check: no exception is returned

2. Method call **removeRepository()**
   Parameters: repositoryName
   Check: no exception is returned

3. Method call **commitTransaction()**
   Parameters: none
   Check: value TRUE is returned



Test PM_PM_10

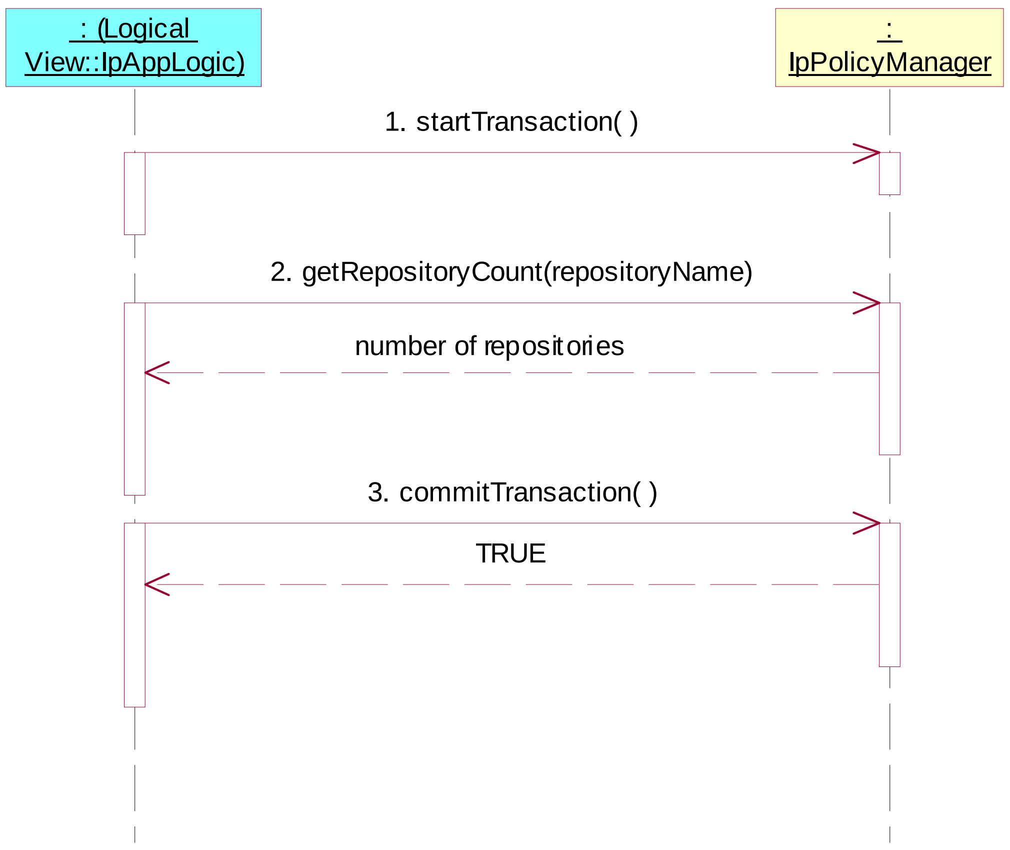
Summary:	get number of policy repositories.
Reference:	ES 202 915-13 [1], clause 8.1.
Precondition:	**getRepositoryCount()** implemented.
Preamble:	Registration of the IUT (Policy Management Control SCF) and the tester (application) to the framework. The tester must have obtained a reference to an instance of the IpPolicyManager interface through selecting that service and signing the required service agreement.
	Policy repositories have to be present and the tester (application) must be authorized to invoke methods related to them.

Test Sequence:

1. Method call **startTransaction()**
   Parameters: none
   Check: no exception is returned

2. Method call **getRepositoryCount()**
   Parameters: none
   Check: valid number of policy repositories is returned

3. Method call **commitTransaction()**
   Parameters: none
   Check: value TRUE is returned



Test PM_PM_11

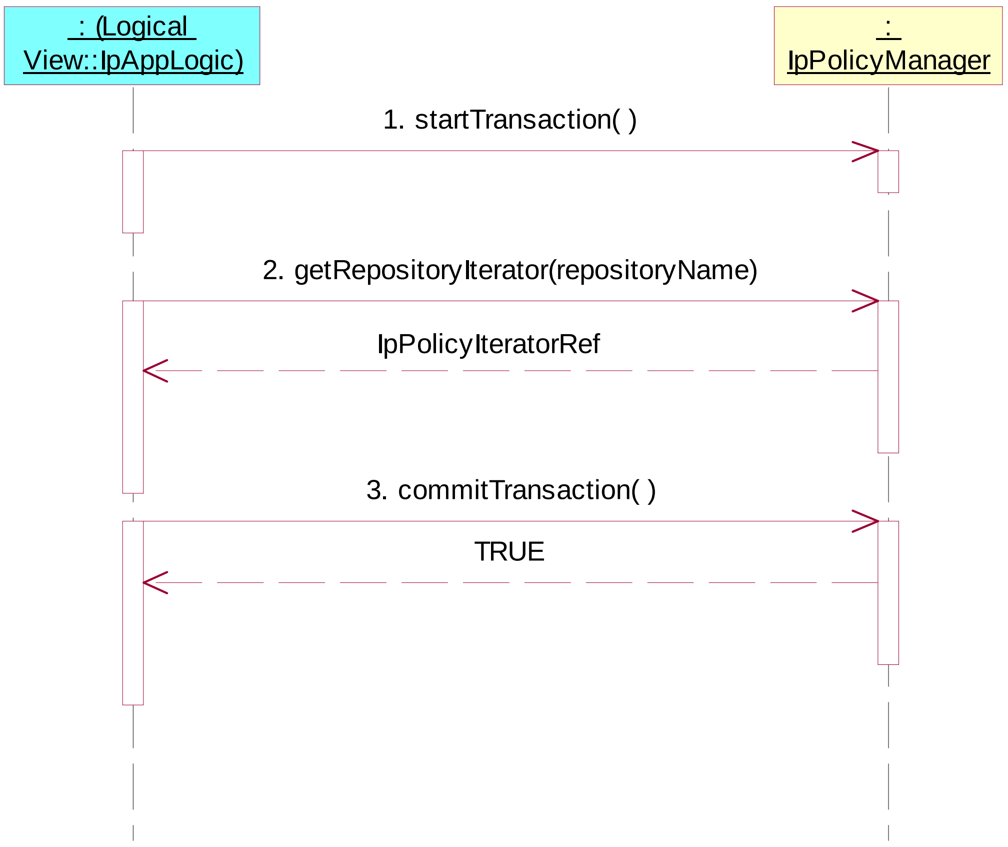
Summary:	get reference to policy repository iterator.
Reference:	ES 202 915-13 [1], clause 8.1.
Precondition:	**getRepositoryIterator()** implemented.
Preamble:	Registration of the IUT (Policy Management Control SCF) and the tester (application) to the framework. The tester must have obtained a reference to an instance of the IpPolicyManager interface through selecting that service and signing the required service agreement.
	Policy repositories have to be present and the tester (application) must be authorized to invoke methods related to them.

Test Sequence:

1. Method call **startTransaction()**
   Parameters: none
   Check: no exception is returned

2. Method call **getRepositoryIterator()**
   Parameters: none
   Check: valid value of IpPolicyIteratorRef is returned

3. Method call **commitTransaction()**
   Parameters: none
   Check: value TRUE is returned



Test PM_PM_12

Summary: start and abort transaction.

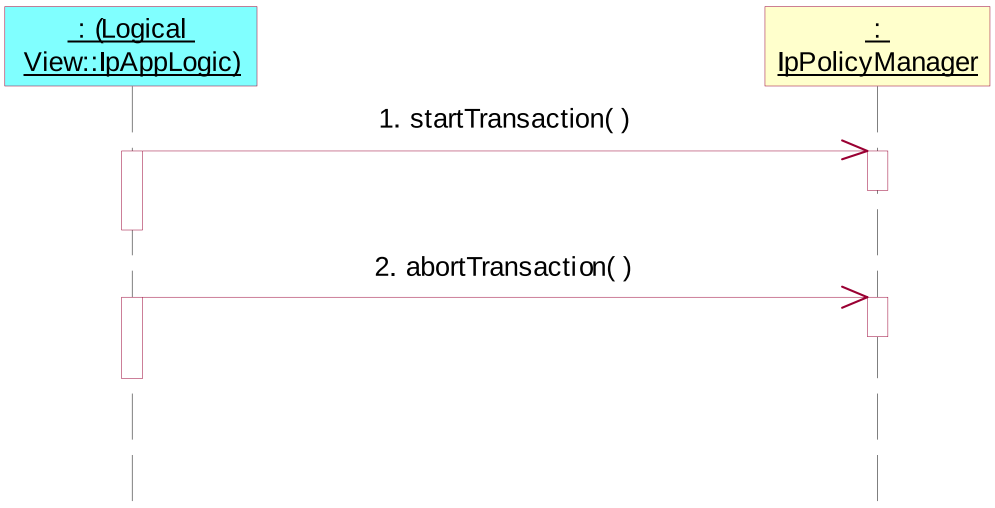
Reference: ES 202 915-13 [1], clause 8.1.

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

Test Sequence:

1. Method call **startTransaction()**
   Parameters: none
   Check: no exception is returned

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



Test PM_PM_13

Summary: start transaction twice.

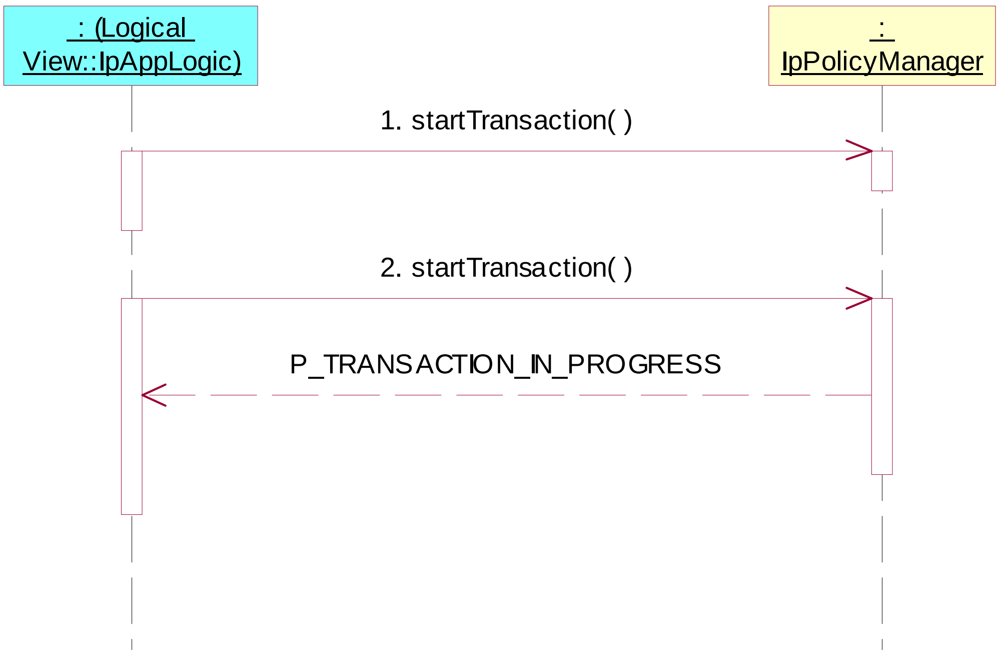
Reference: ES 202 915-13 [1], clause 8.1.

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

Test Sequence:

1. Method call **startTransaction()**
   Parameters: none
   Check: no exception is returned

2. Method call **startTransaction()**
   Parameters: none
   Check: P_TRANSACTION_IN_PROGRESS is returned



Test PM_PM_14

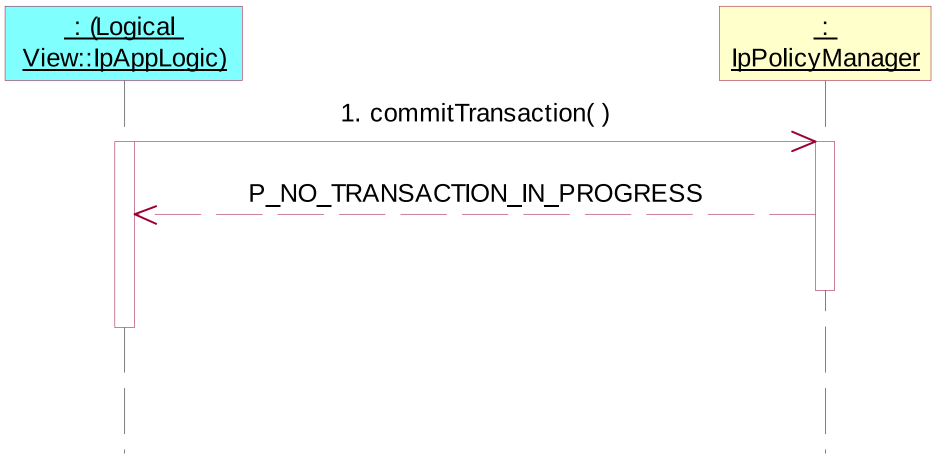
Summary: commit non-started transaction.

Reference: ES 202 915-13 [1], clause 8.1.

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

Test Sequence:

1. Method call **commitTransaction()**
   Parameters: none
   Check: P_NO_TRANSACTION_IN_PROGRESS is returned.



Test PM_PM_15

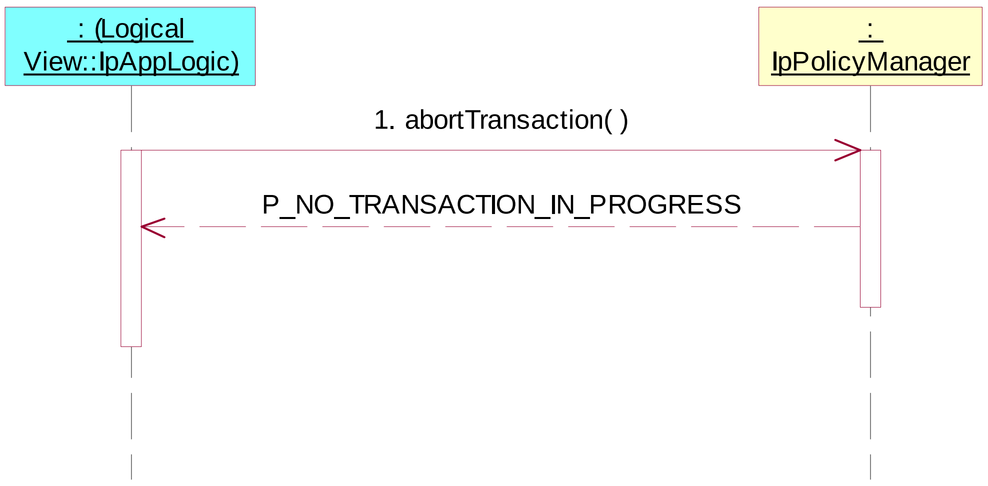
Summary: abort non-started transaction.

Reference: ES 202 915-13 [1], clause 8.1.

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

Test Sequence:

1. Method call **abortTransaction()**
   Parameters: none
   Check: P_NO_TRANSACTION_IN_PROGRESS is returned.



5.2.1.2 IpPolicyDomain

Test PM_PD_01

Summary: create, get and remove subdomain.

Reference: ES 202 915-13 [1], clauses 8.1 and 8.3.

Precondition: **createDomain(), getDomain()** and **removeDomain()** are implemented.

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

Test Sequence:

1. Method call **startTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: no exception is returned

2. Method call **createDomain()** on the IpPolicyManager interface
   Parameters: domainName1
   Check: valid value of IpPolicyDomainRef is returned

3. Method call **createDomain()** on the IpPolicyDomain interface
   Parameters: domainName2
   Check: valid value of IpPolicyDomainRef is returned

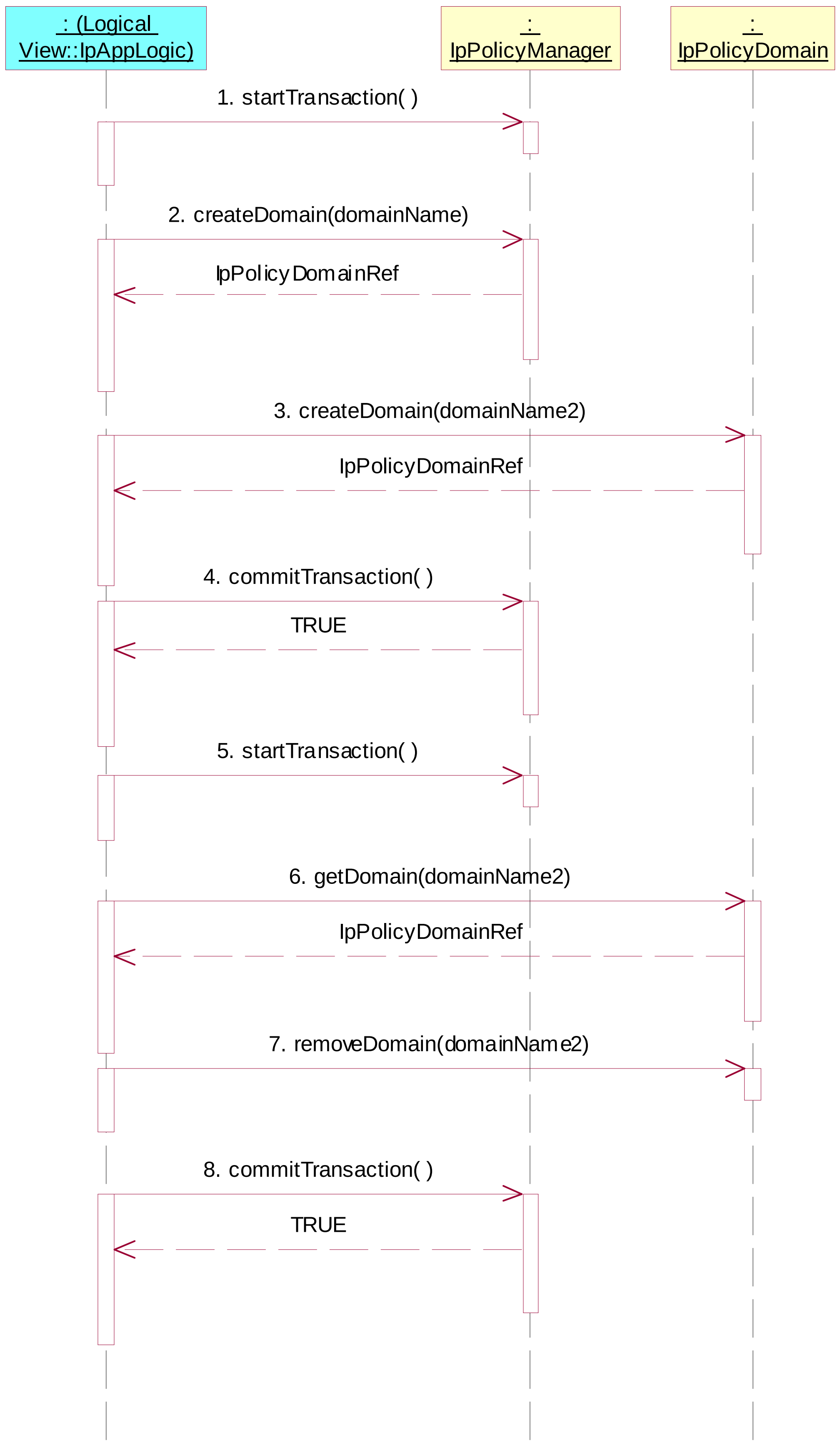
4. Method call **commitTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: value TRUE is returned

5. Method call **startTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: no exception is returned.

6. Method call **getDomain()** on the IpPolicyDomain interface
   Parameters: domainName2
   Check: valid value of IpPolicyDomainRef is returned

7. Method call **removeDomain()** on the IpPolicyDomain interface
   Parameters: domainName2
   Check: no exception is returned

8. Method call **commitTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: value TRUE is returned


: (Logical View::IpAppLogic)
: IpPolicyManager
: IpPolicyDomain
1. startTransaction( )
2. createDomain(domainName)
IpPolicyDomainRef
3. createDomain(domainName2)
IpPolicyDomainRef
4. commitTransaction( )
TRUE
5. startTransaction( )
6. getDomain(domainName2)
IpPolicyDomainRef
7. removeDomain(domainName2)
8. commitTransaction( )
TRUE

Test PM_PD_02

Summary: create domain and set attribute.

Reference: ES 202 915-13 [1], clauses 8.1, 8.2 and 8.3.

Precondition: **createDomain()** and **setAttribute()** are implemented.

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

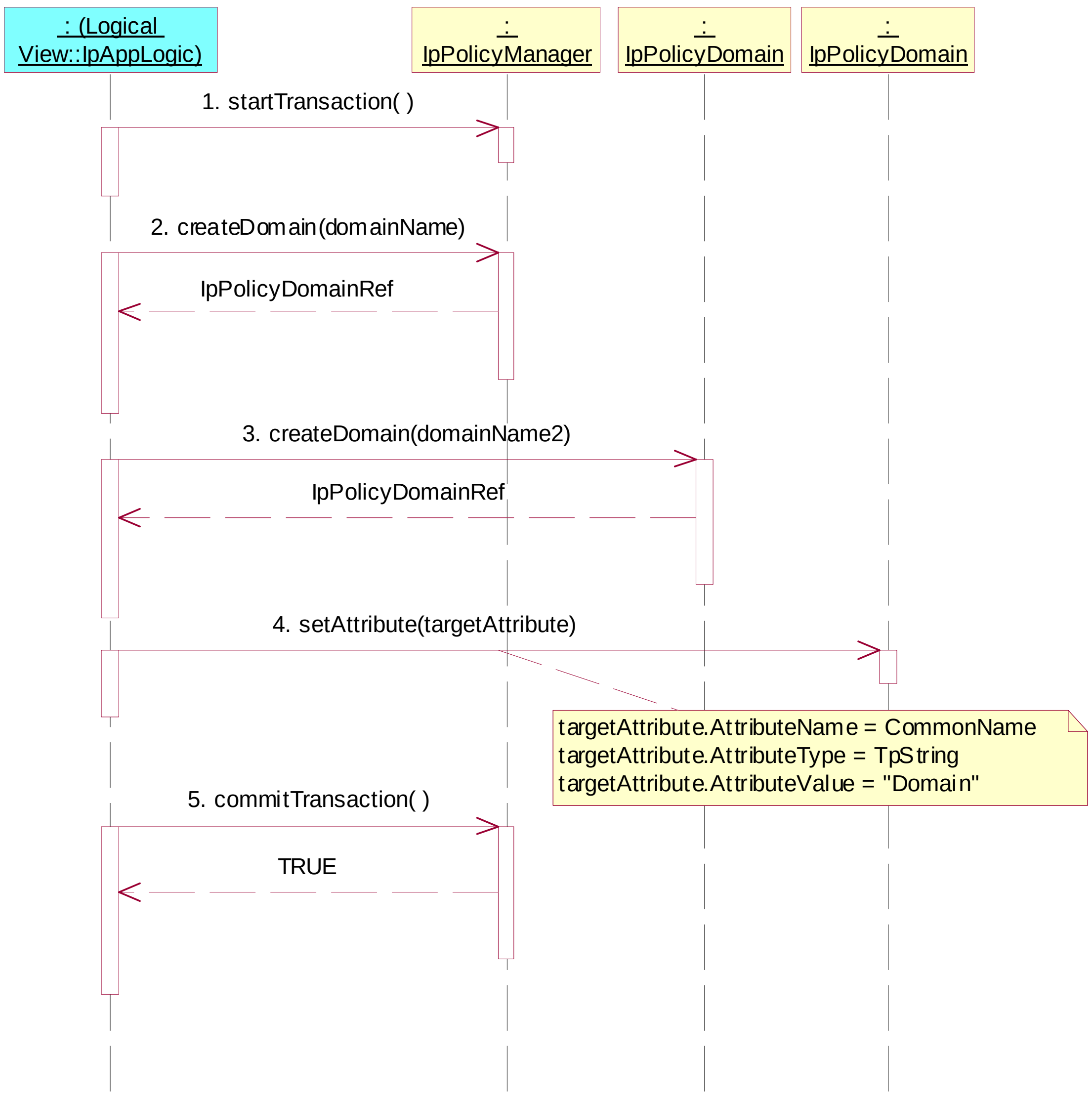
Test Sequence:

1. Method call **startTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: no exception is returned

2. Method call **createDomain()** on the IpPolicyManager interface
   Parameters: domainName1
   Check: valid value of IpPolicyDomainRef is returned

3. Method call **setAttribute()** on the IpPolicyDomain interface
   Parameters: targetAttribute.AttributeName = CommonName
   targetAttribute.AttributeType = TpString
   targetAttribute.AttributeValue = 'Domain'
   Check: no exception is returned

4. Method call **commitTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: value TRUE is returned


: (Logical View::IpAppLogic)
: IpPolicyManager
: IpPolicyDomain
: IpPolicyDomain
1. startTransaction( )
2. createDomain(domainName)
IpPolicyDomainRef
3. createDomain(domainName2)
IpPolicyDomainRef
4. setAttribute(targetAttribute)
targetAttribute.AttributeName = CommonName
targetAttribute.AttributeType = TpString
targetAttribute.AttributeValue = "Domain"
5. commitTransaction( )
TRUE

Test PM_PD_03

Summary: create, get and remove group.

Reference: ES 202 915-13 [1], clauses 8.1 and 8.3.

Precondition: **createDomain(), createGroup(), getGroup()** and **removeGroup()** are implemented.

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

Test Sequence:

1. Method call **startTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: no exception is returned

2. Method call **createDomain()** on the IpPolicyManager interface
   Parameters: DomainName
   Check: valid value of IpPolicyDomainRef is returned

3. Method call **createGroup()** on the IpPolicyDomain interface
   Parameters: groupName
   Check: valid value of IpPolicyGroupRef is returned

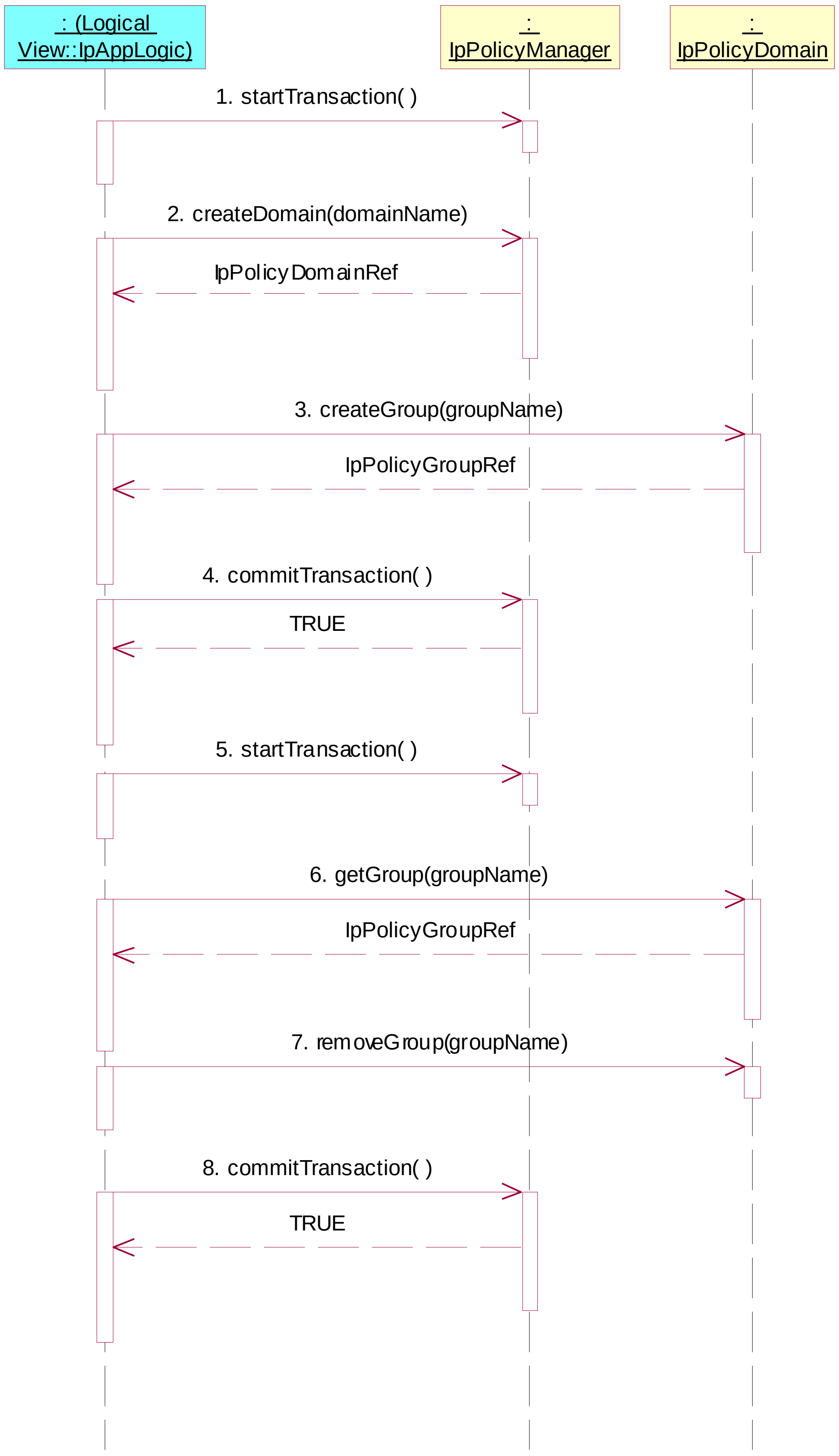
4. Method call **commitTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: value TRUE is returned

5. Method call **startTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: no exception is returned

6. Method call **getGroup()** on the IpPolicyDomain interface
   Parameters: groupName
   Check: valid value of IpPolicyGroupRef is returned

7. Method call **removeGroup()** on the IpPolicyDomain interface
   Parameters: groupName
   Check: no exception is returned

8. Method call **commitTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: value TRUE is returned


: (Logical View::IpAppLogic)
: IpPolicyManager
: IpPolicyDomain
1. startTransaction( )
2. createDomain(domainName)
IpPolicyDomainRef
3. createGroup(groupName)
IpPolicyGroupRef
4. commitTransaction( )
TRUE
5. startTransaction( )
6. getGroup(groupName)
IpPolicyGroupRef
7. removeGroup(groupName)
8. commitTransaction( )
TRUE

Test PM_PD_04

Summary: create group and set attribute.

Reference: ES 202 915-13 [1], clauses 8.1, 8.2 and 8.3.

Precondition: **createDomain(), createGroup()** and **setAttribute()** are implemented.

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

Test Sequence:

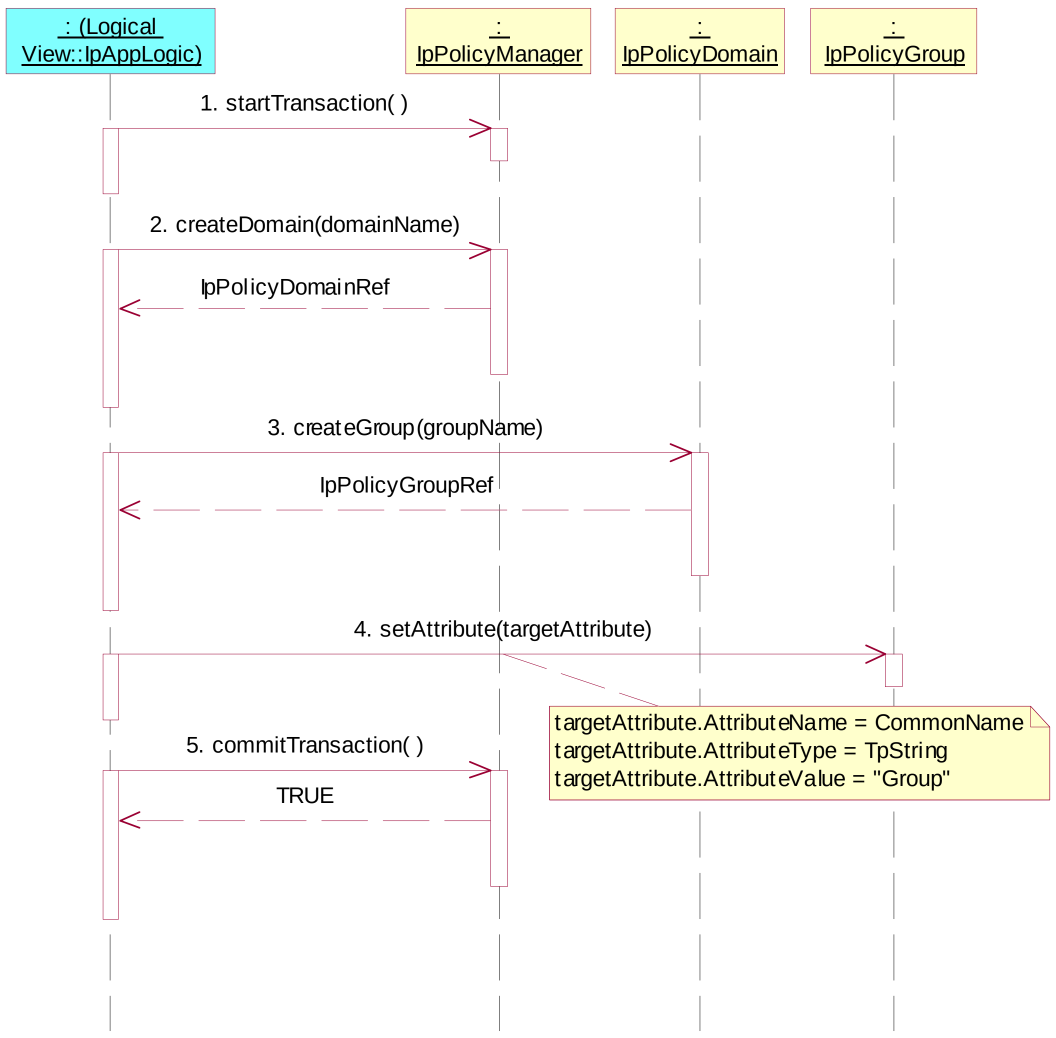
1. Method call **startTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: no exception is returned

2. Method call **createDomain()** on the IpPolicyManager interface
   Parameters: DomainName
   Check: valid value of IpPolicyDomainRef is returned

3. Method call **createGroup()** on the IpPolicyDomain interface
   Parameters: groupName
   Check: valid value of IpPolicyGroupRef is returned

4. Method call **setAttribute()** on the IpPolicyGroup interface
   Parameters: targetAttribute.AttributeName = CommonName
   targetAttribute.AttributeType = TpString
   targetAttribute.AttributeValue = 'Group'
   Check: no exception is returned

5. Method call **commitTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: value TRUE is returned


: (Logical View::IpAppLogic)
: IpPolicyManager
: IpPolicyDomain
: IpPolicyGroup
1. startTransaction( )
2. createDomain(domainName)
IpPolicyDomainRef
3. createGroup(groupName)
IpPolicyGroupRef
4. setAttribute(targetAttribute)
targetAttribute.AttributeName = CommonName
targetAttribute.AttributeType = TpString
targetAttribute.AttributeValue = "Group"
5. commitTransaction( )
TRUE

Test PM_PD_05

Summary:	create, get and remove rule.
Reference:	ES 202 915-13 [1], clauses 8.1 and 8.3.
Precondition:	**createDomain(), createRule(), getRule()** and **removeRule()** are implemented.
Preamble:	Registration of the IUT (Policy Management Control SCF) and the tester (application) to the framework. The tester must have obtained a reference to an instance of the IpPolicyManager interface through selecting that service and signing the required service agreement.

Test Sequence:

1. Method call **startTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: no exception is returned

2. Method call **createDomain()** on the IpPolicyManager interface
   Parameters: DomainName
   Check: valid value of IpPolicyDomainRef is returned

3. Method call **createRule()** on the IpPolicyDomain interface
   Parameters: ruleName
   Check: valid value of IpPolicyRuleRef is returned

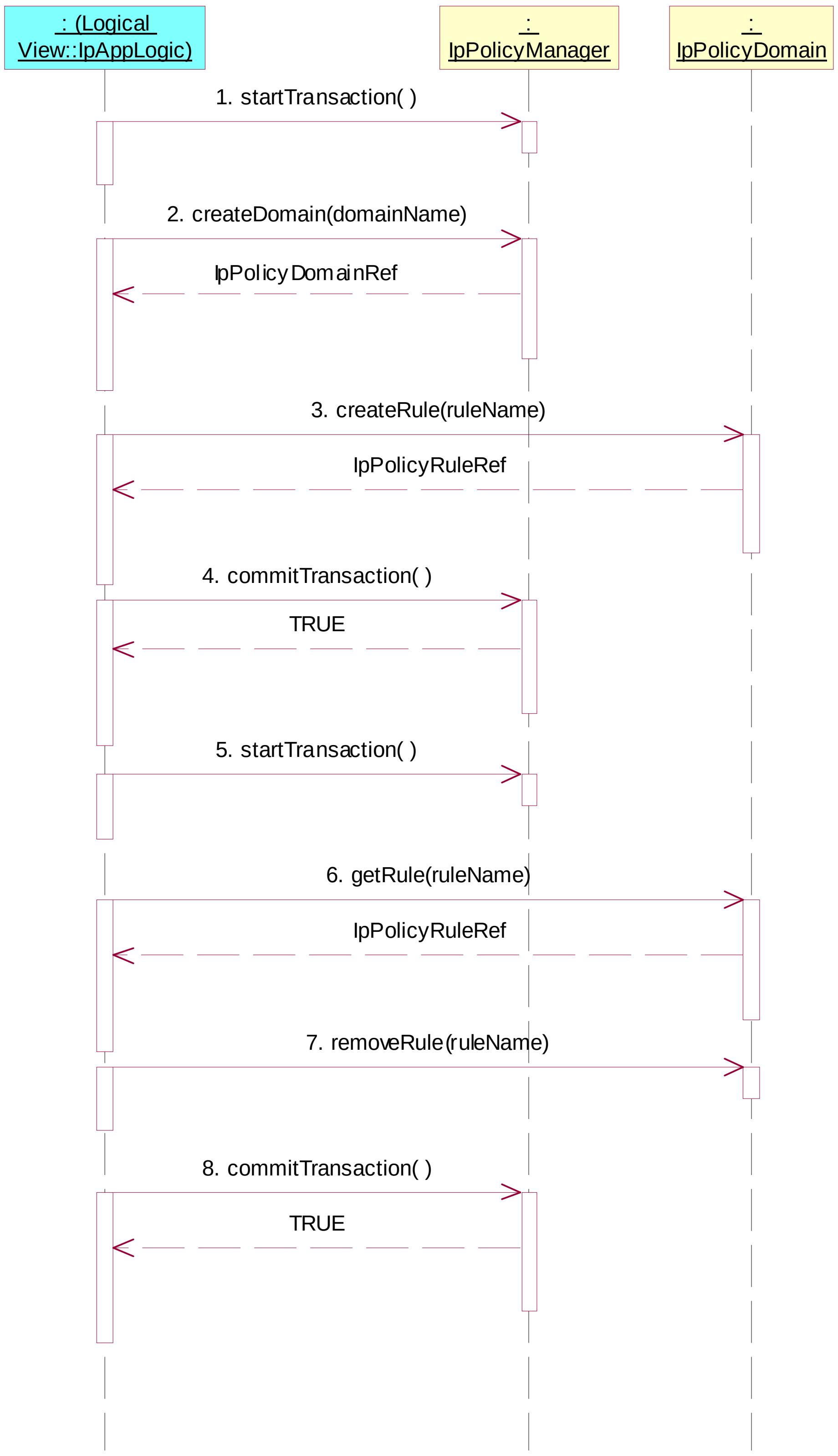
4. Method call **commitTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: value TRUE is returned

5. Method call **startTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: no exception is returned

6. Method call **getRule()** on the IpPolicyDomain interface
   Parameters: ruleName
   Check: valid value of IpPolicyRuleRef is returned

7. Method call **removeRule()** on the IpPolicyDomain interface
   Parameters: ruleName
   Check: no exception is returned

8. Method call **commitTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: value TRUE is returned


: (Logical View::IpAppLogic)
: IpPolicyManager
: IpPolicyDomain
1. startTransaction( )
2. createDomain(domainName)
IpPolicyDomainRef
3. createRule(ruleName)
IpPolicyRuleRef
4. commitTransaction( )
TRUE
5. startTransaction( )
6. getRule(ruleName)
IpPolicyRuleRef
7. removeRule(ruleName)
8. commitTransaction( )
TRUE

Test PM_PD_06

Summary: create and destroy notification.

Reference: ES 202 915-13 [1], clauses 8.1 and 8.3.

Precondition: **createDomain(), createNotification()** and **destroyNotification()** are implemented.

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

Test Sequence:

1. Method call **startTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: no exception is returned

2. Method call **createDomain()** on the IpPolicyManager interface
   Parameters: DomainName
   Check: valid value of IpPolicyDomainRef is returned

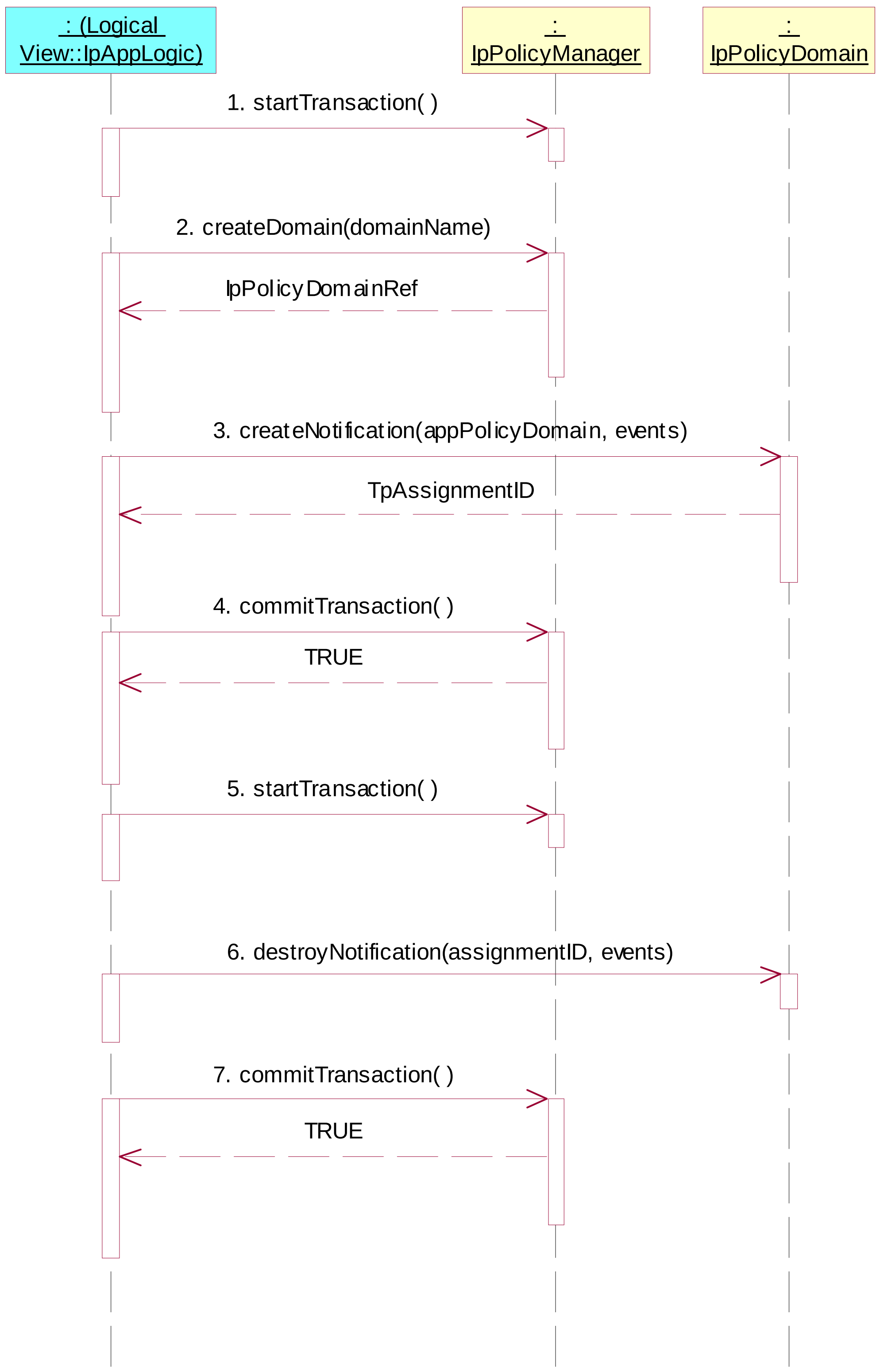
3. Method call **createNotification()** on the IpPolicyDomain interface
   Parameters: appPolicyDomain, events
   Check: valid value of TpAssignmentID is returned

4. Method call **commitTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: value TRUE is returned

5. Method call **startTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: no exception is returned

6. Method call **destroyNotification()** on the IpPolicyDomain interface
   Parameters: assignmentID, events
   Check: no exception is returned

7. Method call **commitTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: value TRUE is returned


: (Logical View::IpAppLogic)
: IpPolicyManager
: IpPolicyDomain
1. startTransaction( )
2. createDomain(domainName)
IpPolicyDomainRef
3. createNotification(appPolicyDomain, events)
TpAssignmentID
4. commitTransaction( )
TRUE
5. startTransaction( )
6. destroyNotification(assignmentID, events)
7. commitTransaction( )
TRUE

5.2.1.3 IpPolicyRule

Test PM_PRU_01

Summary: create, get and remove condition.

Reference: ES 202 915-13 [1], clauses 8.1, 8.3 and 8.6.

Precondition: **createDomain(), createRule(), createCondition(), getCondition()** and **removeCondition()** are implemented.

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

Test Sequence:

1. Method call **startTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: no exception is returned

2. Method call **createDomain()** on the IpPolicyManager interface
   Parameters: domainName
   Check: valid value of IpPolicyDomainRef is returned

3. Method call **createRule()** on the IpPolicyDomain interface
   Parameters: ruleName
   Check: valid value of IpPolicyRuleRef is returned

4. Method call **createCondition()** on the IpPolicyRule interface
   Parameters: conditionName, conditionType, conditionAttributes
   Check: valid value of IpPolicyConditionRef is returned

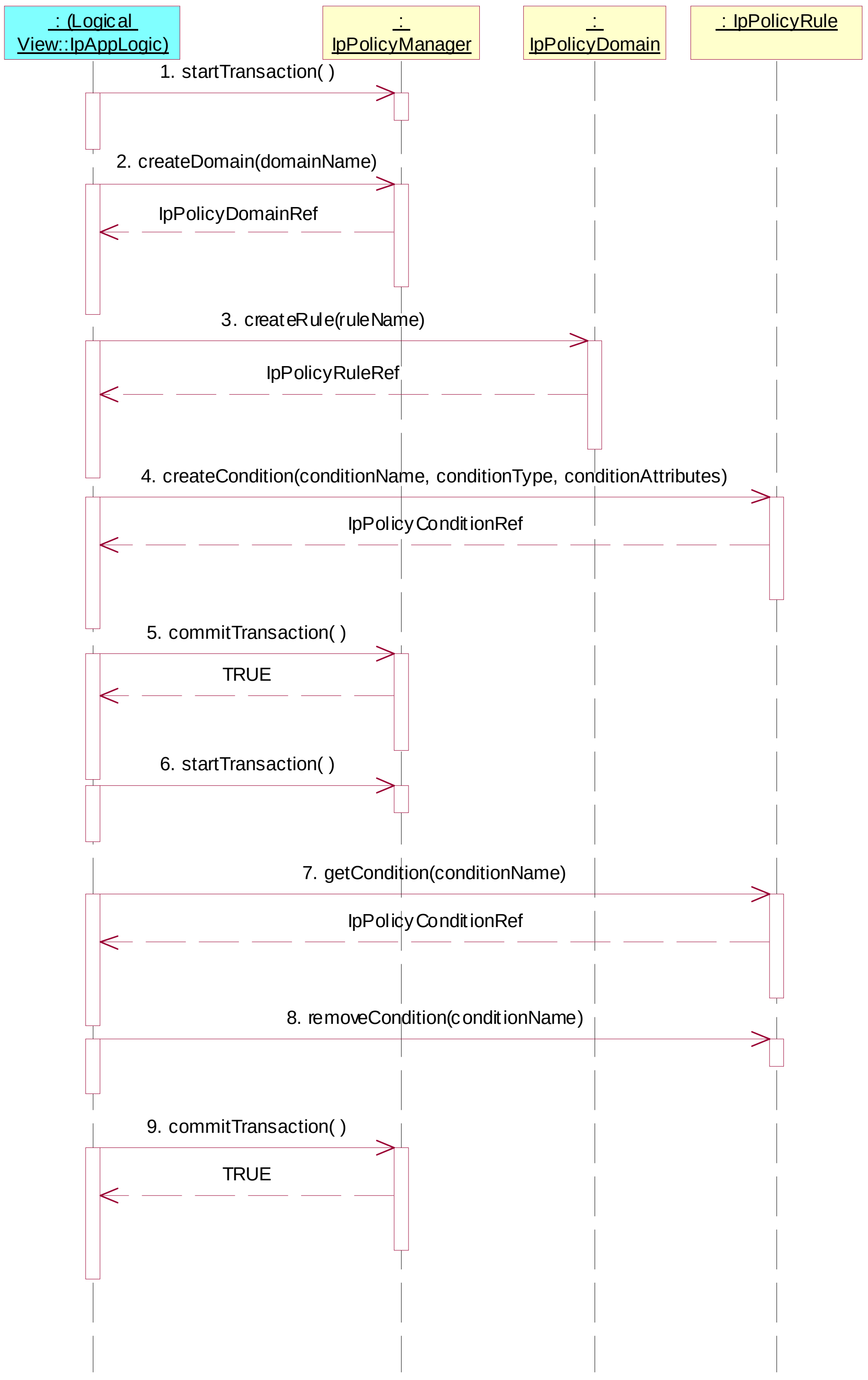
5. Method call **commitTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: value TRUE is returned

6. Method call **startTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: no exception is returned

7. Method call **getCondition()** on the IpPolicyRule interface
   Parameters: conditionName
   Check: valid value of IpPolicyConditionRef is returned

8. Method call **removeCondition()** on the IpPolicyRule interface
   Parameters: conditionName
   Check: no exception is returned

9. Method call **commitTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: value TRUE is returned


: (Logical View::IpAppLogic)
: IpPolicyManager
: IpPolicyDomain
: IpPolicyRule
1. startTransaction( )
2. createDomain(domainName)
IpPolicyDomainRef
3. createRule(ruleName)
IpPolicyRuleRef
4. createCondition(conditionName, conditionType, conditionAttributes)
IpPolicyConditionRef
5. commitTransaction( )
TRUE
6. startTransaction( )
7. getCondition(conditionName)
IpPolicyConditionRef
8. removeCondition(conditionName)
9. commitTransaction( )
TRUE

Test PM_PRU_02

Summary: create, get and remove action.

Reference: ES 202 915-13 [1], clauses 8.1, 8.3 and 8.6.

Precondition: **createDomain(), createRule(), createAction(), getAction()** and **removeAction()** are implemented.

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

Test Sequence:

1. Method call **startTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: no exception is returned

2. Method call **createDomain()** on the IpPolicyManager interface
   Parameters: domainName
   Check: valid value of IpPolicyDomainRef is returned

3. Method call **createRule()** on the IpPolicyDomain interface
   Parameters: ruleName
   Check: valid value of IpPolicyRuleRef is returned

4. Method call **createAction()** on the IpPolicyRule interface
   Parameters: actionName, actionType, actionAttributes
   Check: valid value of IpPolicyActionRef is returned

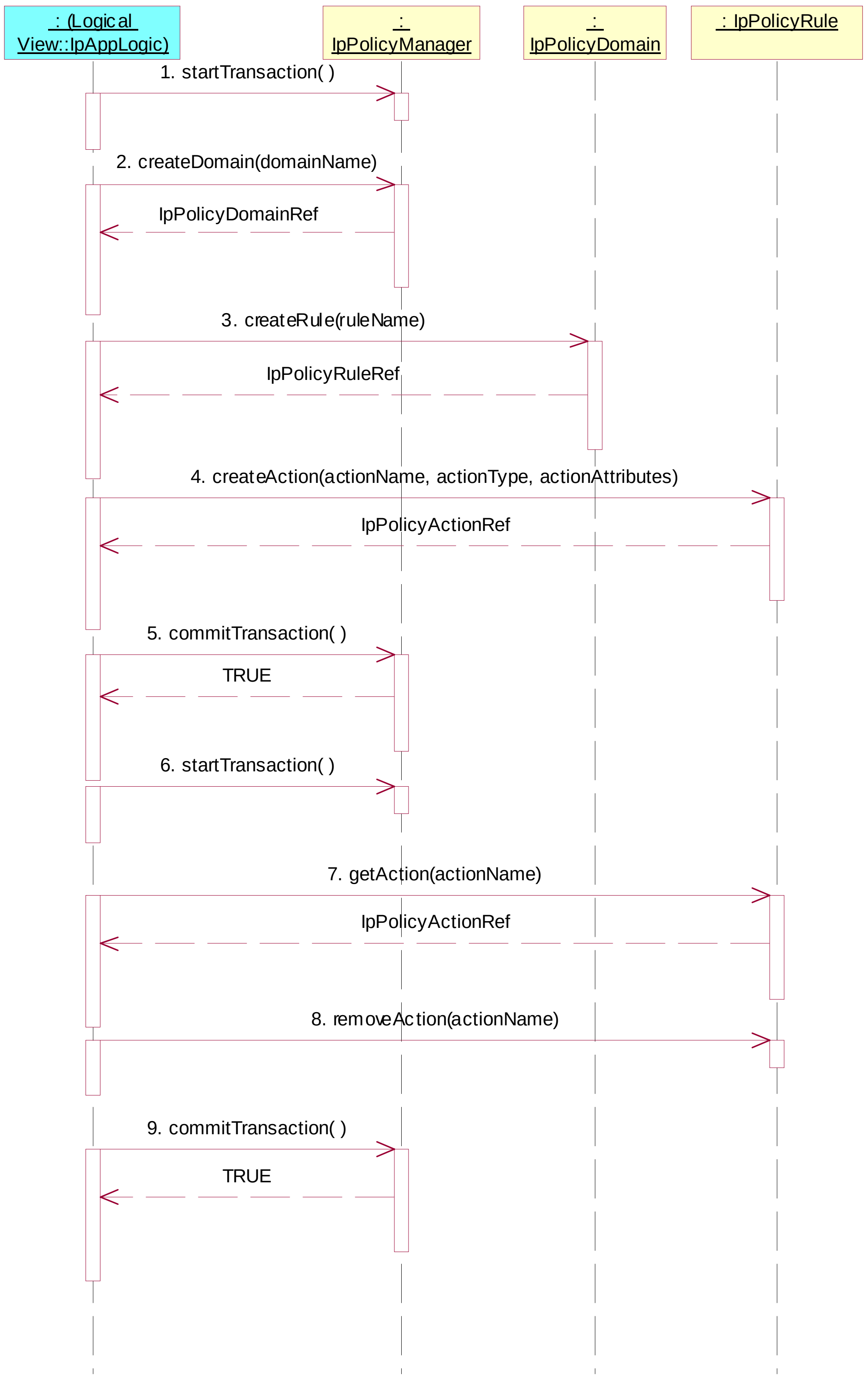
5. Method call **commitTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: value TRUE is returned

6. Method call **startTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: no exception is returned

7. Method call **getAction()** on the IpPolicyRule interface
   Parameters: actionName
   Check: valid value of IpPolicyActionRef is returned

8. Method call **removeAction()** on the IpPolicyRule interface
   Parameters: actionName
   Check: no exception is returned

9. Method call **commitTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: value TRUE is returned


: (Logical View::IpAppLogic)
: IpPolicyManager
: IpPolicyDomain
: IpPolicyRule
1. startTransaction( )
2. createDomain(domainName)
IpPolicyDomainRef
3. createRule(ruleName)
IpPolicyRuleRef
4. createAction(actionName, actionType, actionAttributes)
IpPolicyActionRef
5. commitTransaction( )
TRUE
6. startTransaction( )
7. getAction(actionName)
IpPolicyActionRef
8. removeAction(actionName)
9. commitTransaction( )
TRUE

Test PM_PRU_03

Summary: set, get and unset validity period.

Reference: ES 202 915-13 [1], clauses 8.1, 8.3 and 8.6.

Precondition: **createDomain(), createRule(), setValidityPeriodCondition(), getValidityPeriodCondition()** and **unsetValidityPeriodCondition()** are implemented.

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

Test Sequence:

1. Method call **startTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: no exception is returned

2. Method call **createDomain()** on the IpPolicyManager interface
   Parameters: domainName
   Check: valid value of IpPolicyDomainRef is returned

3. Method call **createRule()** on the IpPolicyDomain interface
   Parameters: ruleName
   Check: valid value of IpPolicyRuleRef is returned

4. Method call **setValidityPeriodCondition()** on the IpPolicyRule interface
   Parameters: conditionReference, actionType, actionAttributes
   Check: no exception is returned

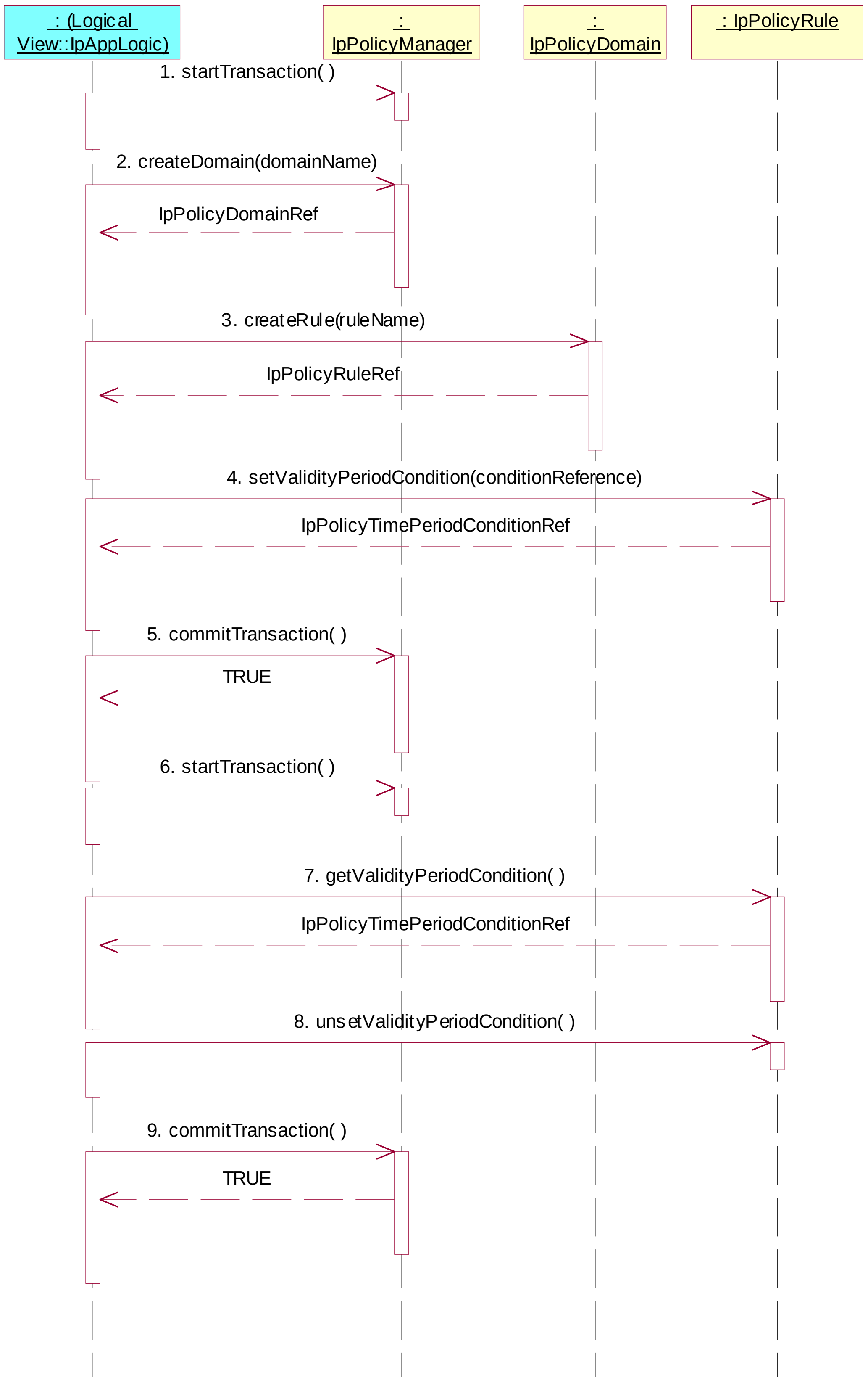
5. Method call **commitTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: value TRUE is returned

6. Method call **startTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: no exception is returned

7. Method call **getValidityPeriodCondition()** on the IpPolicyRule interface
   Parameters: actionName
   Check: valid value of IpPolicyTimePeriodConditionRef is returned

8. Method call **unsetValidityPeriodCondition()** on the IpPolicyRule interface
   Parameters: none
   Check: no exception is returned

9. Method call **commitTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: value TRUE is returned


: (Logical View::IpAppLogic)
: IpPolicyManager
: IpPolicyDomain
: IpPolicyRule
1. startTransaction( )
2. createDomain(domainName)
IpPolicyDomainRef
3. createRule(ruleName)
IpPolicyRuleRef
4. setValidityPeriodCondition(conditionReference)
IpPolicyTimePeriodConditionRef
5. commitTransaction( )
TRUE
6. startTransaction( )
7. getValidityPeriodCondition( )
IpPolicyTimePeriodConditionRef
8. unsetValidityPeriodCondition( )
9. commitTransaction( )
TRUE

Test PM_PRU_04

Summary: create action and condition, set action and condition list, enable rule.

Reference: ES 202 915-13 [1], clauses 8.1, 8.2, 8.3 and 8.6.

Precondition: **createDomain(), createRule(), createAction(), createCondition(), setActionList(), setConditionList()** and **setAttribute()** are implemented.

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

Test Sequence:

1. Method call **startTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: no exception is returned

2. Method call **createDomain()** on the IpPolicyManager interface
   Parameters: domainName
   Check: valid value of IpPolicyDomainRef is returned

3. Method call **createRule()** on the IpPolicyDomain interface
   Parameters: ruleName
   Check: valid value of IpPolicyRuleRef is returned

4. Method call **createAction()** on the IpPolicyRule interface
   Parameters: actionName, actionType, actionAttributes
   Check: valid value of IpPolicyActionRef is returned

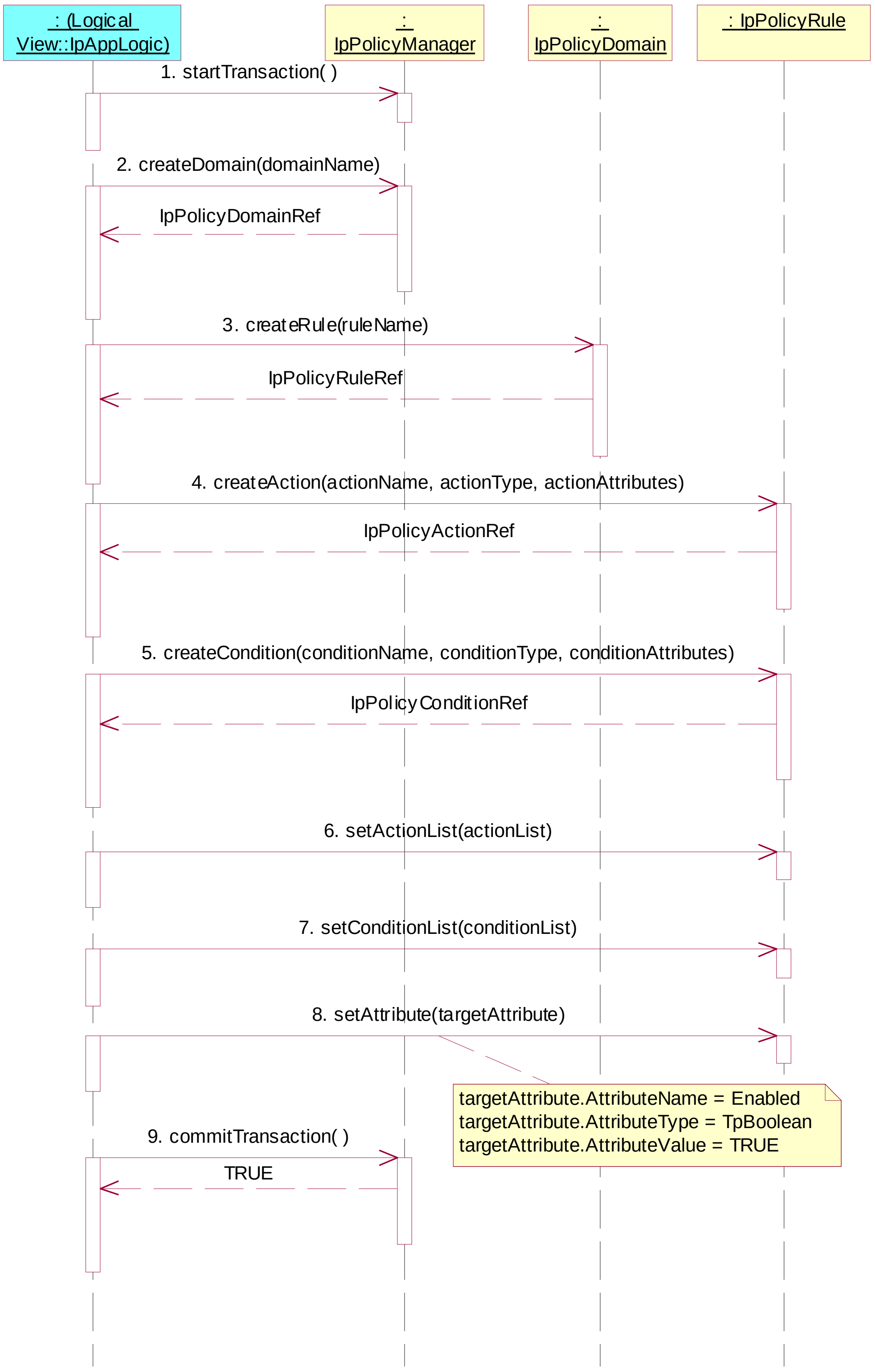
5. Method call **createCondition()** on the IpPolicyRule interface
   Parameters: conditionName, conditionType, conditionAttributes
   Check: valid value of IpPolicyConditionRef is returned

6. Method call **setActionList()** on the IpPolicyRule interface
   Parameters: actionList
   Check: no exception is returned

7. Method call **setConditionList()** on the IpPolicyRule interface
   Parameters: conditionList
   Check: no exception is returned.

8. Method call **setAttribute()** on the IpPolicyRule interface
   Parameters: targetAttribute.AttributeName = Enabled
   targetAttribute.AttributeType = TpBoolean
   targetAttribute.AttributeValue = TRUE
   Check: no exception is returned

9. Method call **commitTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: value TRUE is returned


: (Logical View::IpAppLogic)
: IpPolicyManager
: IpPolicyDomain
: IpPolicyRule
1. startTransaction( )
2. createDomain(domainName)
IpPolicyDomainRef
3. createRule(ruleName)
IpPolicyRuleRef
4. createAction(actionName, actionType, actionAttributes)
IpPolicyActionRef
5. createCondition(conditionName, conditionType, conditionAttributes)
IpPolicyConditionRef
6. setActionList(actionList)
7. setConditionList(conditionList)
8. setAttribute(targetAttribute)
targetAttribute.AttributeName = Enabled
targetAttribute.AttributeType = TpBoolean
targetAttribute.AttributeValue = TRUE
9. commitTransaction( )
TRUE

Test PM_PRU_05

Summary: get action and condition from repository, set action and condition list, enable rule.

Reference: ES 202 915-13 [1], clauses 8.1, 8.2, 8.3 and 8.6.

Precondition: **createDomain(), createRule(), getAction(), getCondition(), setActionList(), setConditionList()** and **setAttribute()** are implemented.

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

A policy repository containing at least one rule and one condition has to be present and the tester (application) must be authorized to invoke methods related to it.

Test Sequence:

1. Method call **startTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: no exception is returned

2. Method call **createDomain()** on the IpPolicyManager interface
   Parameters: domainName
   Check: valid value of IpPolicyDomainRef is returned

3. Method call **createRule()** on the IpPolicyDomain interface
   Parameters: ruleName
   Check: valid value of IpPolicyRuleRef is returned

4. Method call **getRepository()** on the IpPolicyManager interface
   Parameters: repositoryName
   Check: valid value of IpPolicyRepositoryRef is returned

5. Method call **getAction()** on the IpPolicyRepository interface
   Parameters: actionName
   Check: valid value of IpPolicyActionRef is returned

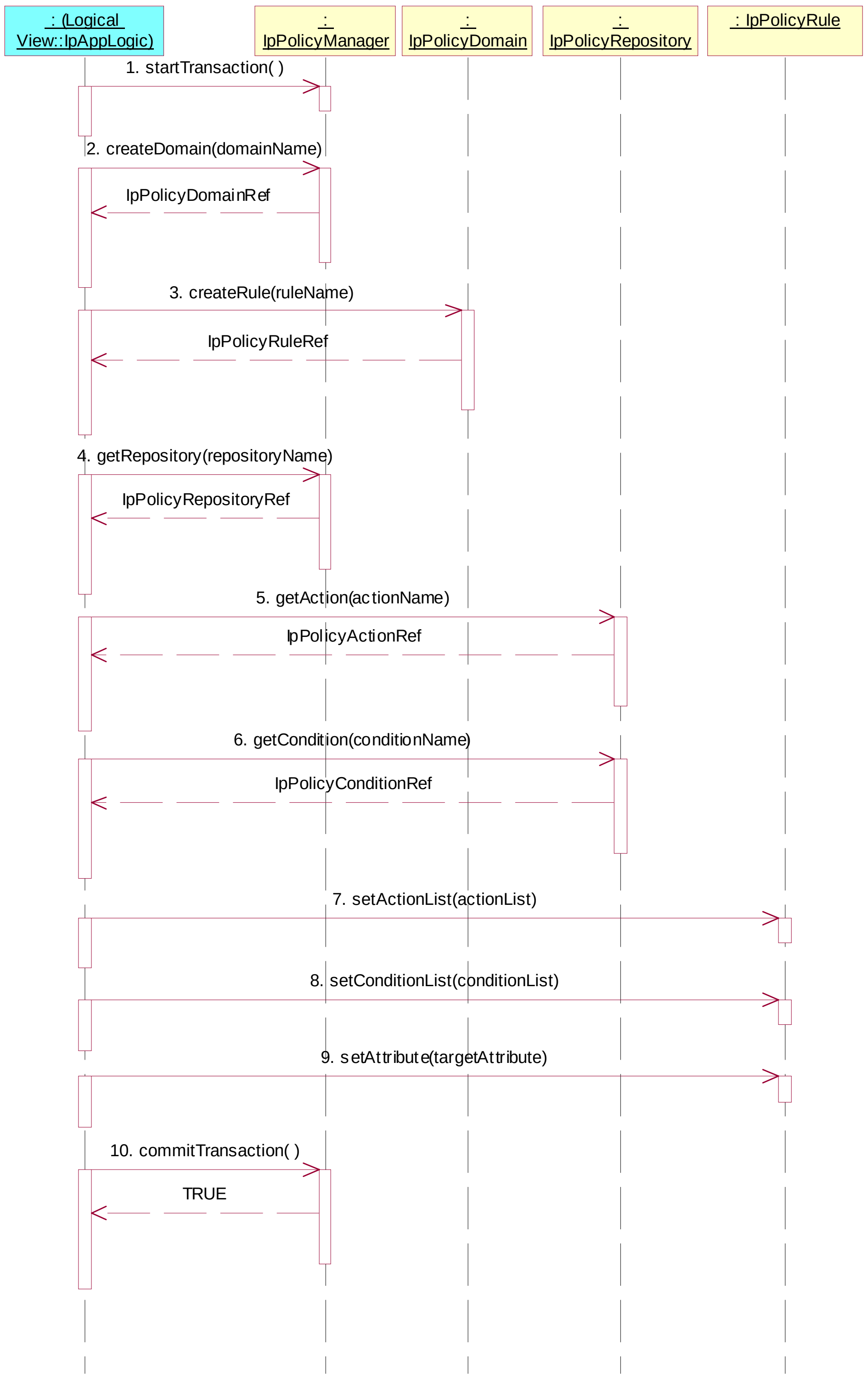
6. Method call **getCondition()** on the IpPolicyRepository interface
   Parameters: conditionName
   Check: valid value of IpPolicyConditionRef is returned

7. Method call **setActionList()** on the IpPolicyRule interface
   Parameters: actionList
   Check: no exception is returned

8. Method call **setConditionList()** on the IpPolicyRule interface
   Parameters: conditionList
   Check: no exception is returned.

9. Method call **setAttribute()** on the IpPolicyRule interface
   Parameters: targetAttribute.AttributeName = Enabled
   targetAttribute.AttributeType = TpBoolean
   targetAttribute.AttributeValue = TRUE
   Check: no exception is returned

10. Method call **commitTransaction()** on the IpPolicyManager interface
    Parameters: none
    Check: value TRUE is returned


: (Logical View::IpAppLogic)
: IpPolicyManager
: IpPolicyDomain
: IpPolicyRepository
: IpPolicyRule
1. startTransaction( )
2. createDomain(domainName)
IpPolicyDomainRef
3. createRule(ruleName)
IpPolicyRuleRef
4. getRepository(repositoryName)
IpPolicyRepositoryRef
5. getAction(actionName)
IpPolicyActionRef
6. getCondition(conditionName)
IpPolicyConditionRef
7. setActionList(actionList)
8. setConditionList(conditionList)
9. setAttribute(targetAttribute)
10. commitTransaction( )
TRUE

5.2.1.4 IpPolicyRepository

Test PM_PR_01

Summary: create, get and remove repository.

Reference: ES 202 915-13 [1], clauses 8.1 and 8.5.

Precondition: **createRepository(), getRepository()** and **removeRepository()** are implemented.

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

Test Sequence:

1. Method call **startTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: no exception is returned

2. Method call **createRepository()** on the IpPolicyManager interface
   Parameters: repositoryName1
   Check: valid value of IpPolicyRepositoryRef is returned

3. Method call **createRepository()** on the IpRepository interface
   Parameters: repositoryName2
   Check: valid value of IpPolicyRepositoryRef is returned

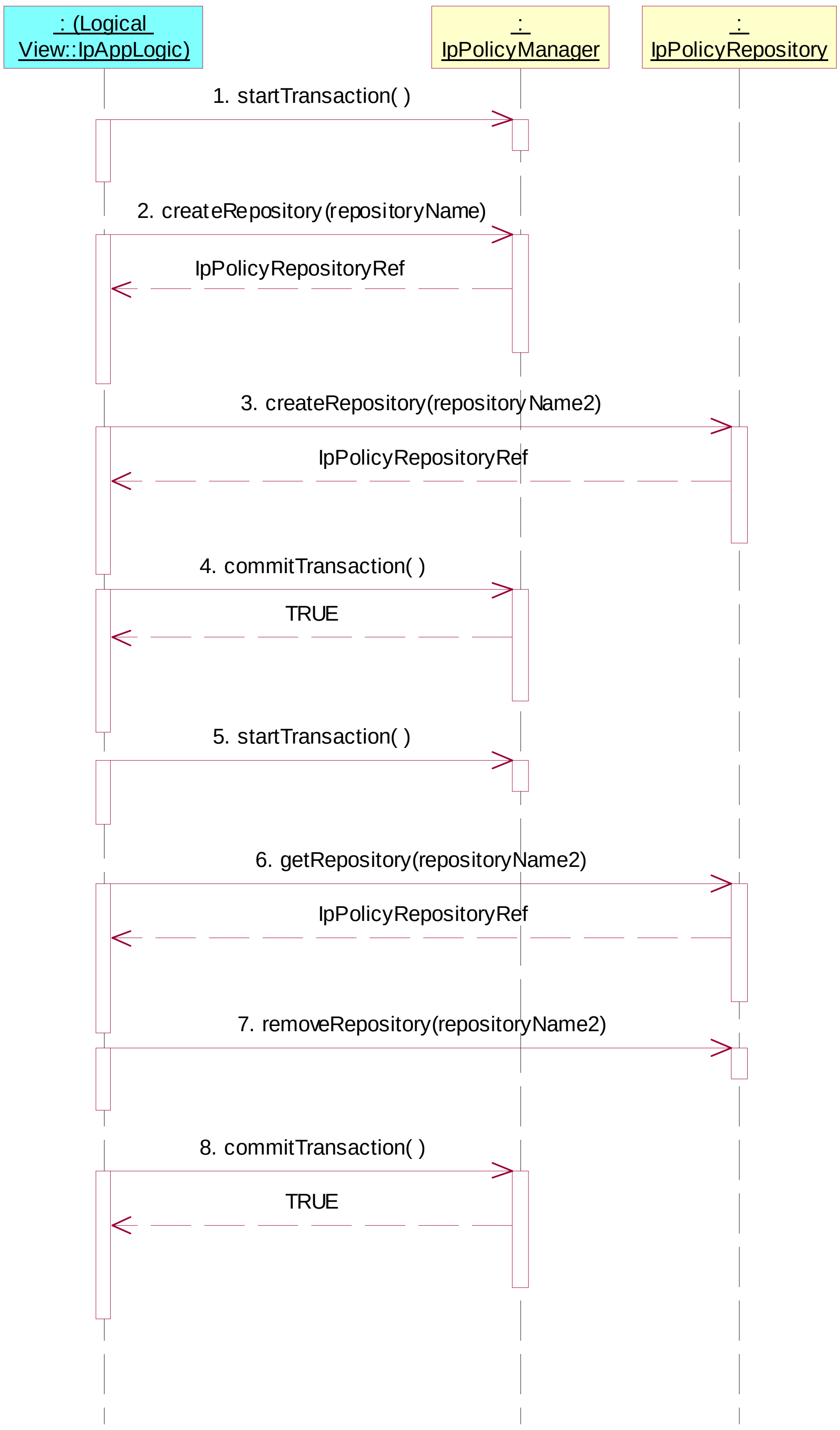
4. Method call **commitTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: value TRUE is returned

5. Method call **startTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: no exception is returned

6. Method call **getRepository()** on the IpRepository interface
   Parameters: repositoryName2
   Check: valid value of IpPolicyRepositoryRef is returned

7. Method call **removeRepository()** on the IpRepository interface
   Parameters: repositoryName2
   Check: no exception is returned

8. Method call **commitTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: value TRUE is returned


: (Logical View::IpAppLogic)
: IpPolicyManager
: IpPolicyRepository
1. startTransaction( )
2. createRepository(repositoryName)
IpPolicyRepositoryRef
3. createRepository(repositoryName2)
IpPolicyRepositoryRef
4. commitTransaction( )
TRUE
5. startTransaction( )
6. getRepository(repositoryName2)
IpPolicyRepositoryRef
7. removeRepository(repositoryName2)
8. commitTransaction( )
TRUE

Test PM_PR_02

Summary: create, get and remove condition.

Reference: ES 202 915-13 [1], clauses 8.1 and 8.5.

Precondition: **createRepository(), createCondition(), getCondition()** and **removeCondition()** are implemented.

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

Test Sequence:

1. Method call **startTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: no exception is returned

2. Method call **createRepository()** on the IpPolicyManager interface
   Parameters: repositoryName
   Check: valid value of IpPolicyDomainRef is returned

3. Method call **createCondition()** on the IpRepository interface
   Parameters: conditionName, conditionType, conditionAttributes
   Check: valid value of IpPolicyConditionRef is returned

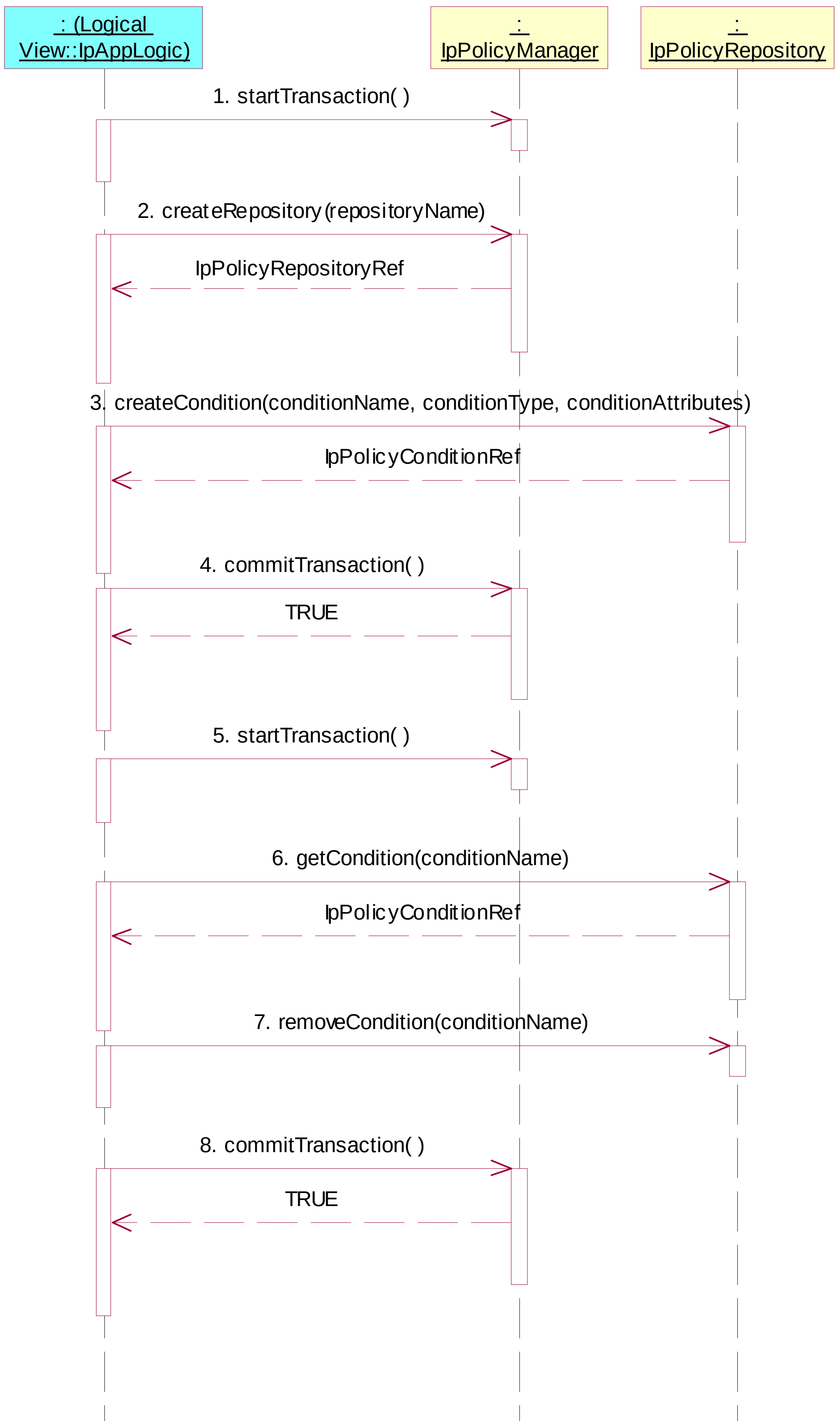
4. Method call **commitTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: value TRUE is returned

5. Method call **startTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: no exception is returned

6. Method call **getCondition()** on the IpRepository interface
   Parameters: conditionName
   Check: valid value of IpPolicyConditionRef is returned

7. Method call **removeCondition()** on the IpRepository interface
   Parameters: conditionName
   Check: no exception is returned

8. Method call **commitTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: value TRUE is returned


: (Logical View::IpAppLogic)
: IpPolicyManager
: IpPolicyRepository
1. startTransaction( )
2. createRepository(repositoryName)
IpPolicyRepositoryRef
3. createCondition(conditionName, conditionType, conditionAttributes)
IpPolicyConditionRef
4. commitTransaction( )
TRUE
5. startTransaction( )
6. getCondition(conditionName)
IpPolicyConditionRef
7. removeCondition(conditionName)
8. commitTransaction( )
TRUE

Test PM_PR_03

Summary: create, get and remove action.

Reference: ES 202 915-13 [1], clauses 8.1 and 8.5.

Precondition: **createRepository(), createAction(), getAction()** and **removeAction()** are implemented.

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

Test Sequence:

1. Method call **startTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: no exception is returned

2. Method call **createRepository()** on the IpPolicyManager interface
   Parameters: repositoryName
   Check: valid value of IpPolicyDomainRef is returned

3. Method call **createAction()** on the IpRepository interface
   Parameters: actionName, actionType, actionAttributes
   Check: valid value of IpPolicyActionRef is returned

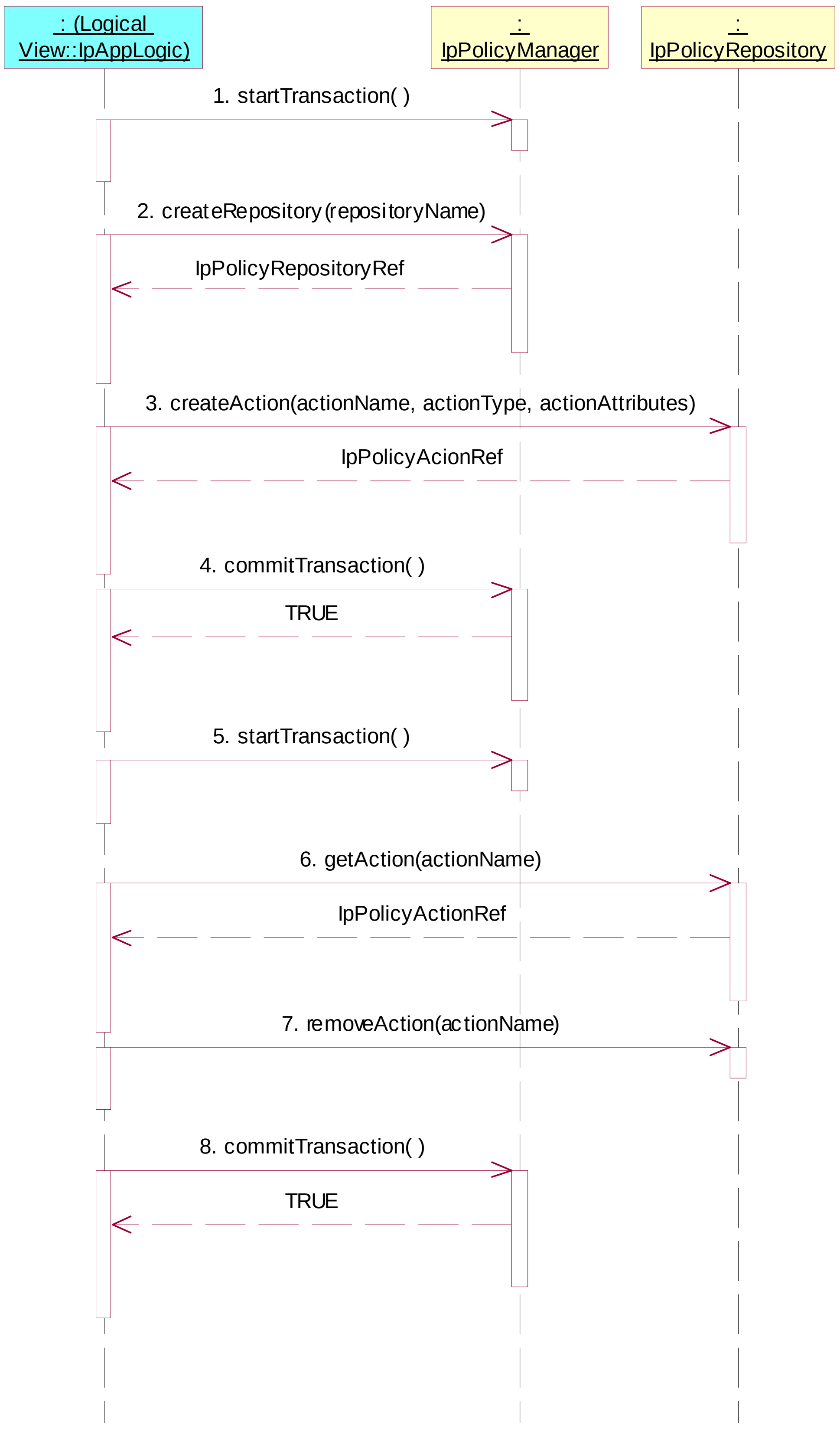
4. Method call **commitTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: value TRUE is returned

5. Method call **startTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: no exception is returned

6. Method call **getAction()** on the IpRepository interface
   Parameters: actionName
   Check: valid value of IpPolicyActionRef is returned

7. Method call **removeAction()** on the IpRepository interface
   Parameters: actionName
   Check: no exception is returned

8. Method call **commitTransaction()** on the IpPolicyManager interface
   Parameters: none
   Check: value TRUE is returned


: (Logical View::IpAppLogic)
: IpPolicyManager
: IpPolicyRepository
1. startTransaction( )
2. createRepository(repositoryName)
IpPolicyRepositoryRef
3. createAction(actionName, actionType, actionAttributes)
IpPolicyAcionRef
4. commitTransaction( )
TRUE
5. startTransaction( )
6. getAction(actionName)
IpPolicyActionRef
7. removeAction(actionName)
8. commitTransaction( )
TRUE

5.2.2 Policy Management, application side

Test PM_APP_01

Summary:	enable and accept notifications.
Reference:	ES 202 915-13 [1], clauses 8.16.
Precondition:	IUT capable of invoking **getDomain()** and **createNotification().**
Preamble:	registration of the IUT (application) and the tester (Policy Management SCF) to the framework. The IUT must have obtained a reference to an instance of the IpPolicyManager interface through selecting that service and signing the required service agreement.

Test Sequence:

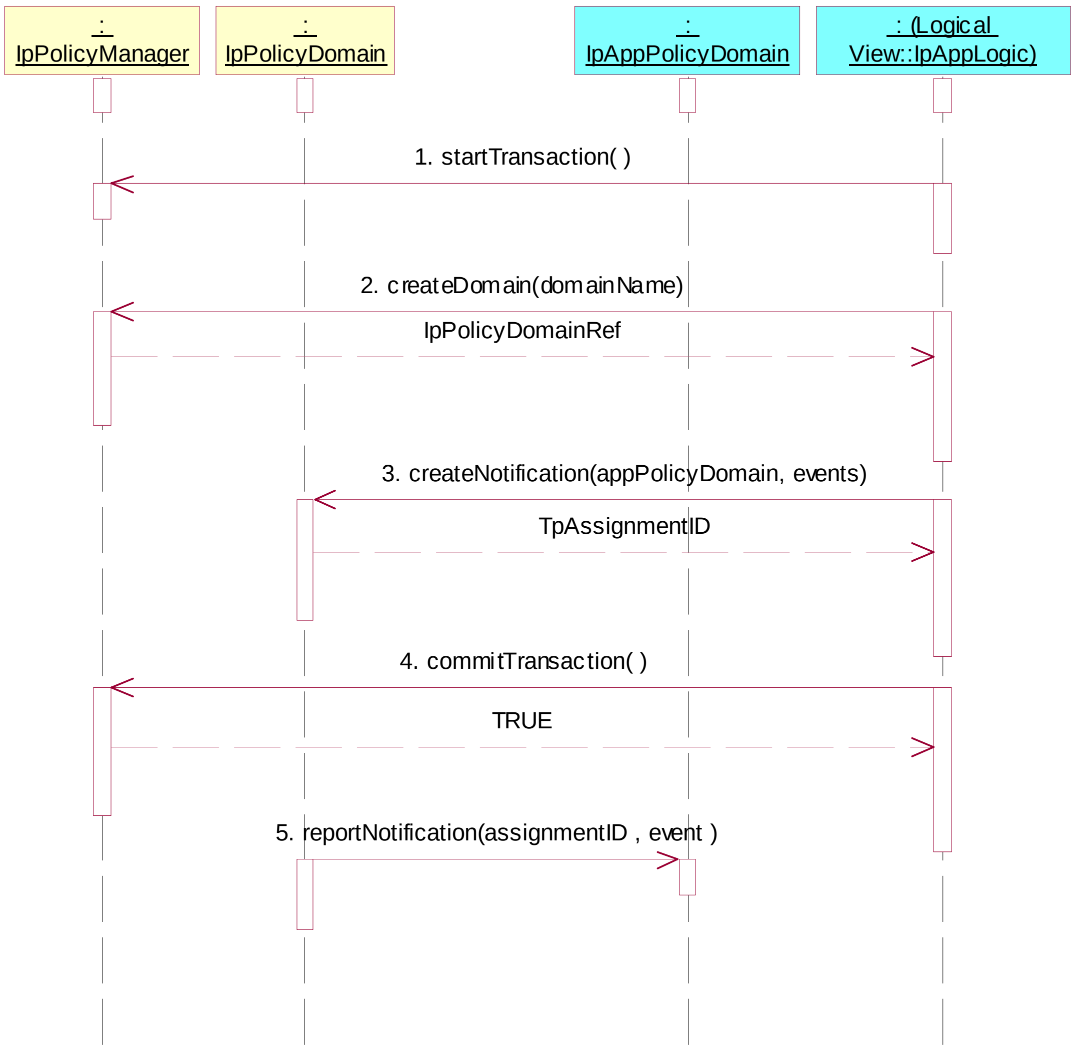
1. Triggered Action: cause IUT to call **startTransaction()** method on the tester's (SCF's) IpPolicyManager interface
   Parameters: none

2. Triggered Action: cause IUT to call **getDomain()** method on the tester's (SCF's) IpPolicyManager interface
   Parameters: domainName

3. Triggered Action: cause IUT to call **createNotification()** method on the tester's (SCF's) IpPolicyDomain interface
   Parameters: appPolicyDomain, events

4. Triggered Action: cause IUT to call **commitTransaction()** method on the tester's (SCF's) IpPolicyManager interface
   Parameters: none

5. Method call **reportNotification()**
   Parameters: assignmentID, event
   Check: no exception is returned


: IpPolicyManager
: IpPolicyDomain
: IpAppPolicyDomain
: (Logical View::IpAppLogic)
1. startTransaction( )
2. createDomain(domainName)
IpPolicyDomainRef
3. createNotification(appPolicyDomain, events)
TpAssignmentID
4. commitTransaction( )
TRUE
5. reportNotification(assignmentID , event)

History

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