RSP Test Specification for the eUICC
SGP.23-1 Version 3.1.2
26 April 2024

Security Classification: Non-Confidential

Access to and distribution of this document is restricted to the persons permitted by the security classification. This document is confidential to the Association and is subject to copyright protection. This document is to be used only for the purposes for which it has been supplied and information contained in it must not be disclosed or in any other way made available, in whole or in part, to persons other than those permitted under the security classification without the prior written approval of the Association.

Copyright Notice

Disclaimer

The GSM Association (“Association”) makes no representation, warranty or undertaking (express or implied) with respect to and does not accept any responsibility for, and hereby disclaims liability for the accuracy or completeness or timeliness of the information contained in this document. The information contained in this document may be subject to change without prior notice.

Antitrust Notice

The information contain herein is in full compliance with the GSM Association’s antitrust compliance policy.

Table of Contents

1 Introduction

1.1 Overview

The main aim of the GSMA Remote SIM Provisioning specifications [2] & [3] is to provide solution for the Remote SIM Provisioning of Consumer Devices. The adoption of this technical solution will provide the basis for global interoperability between different Operator deployment scenarios, for example network equipment (e.g. Subscription Manager Data Preparation (SM-DP+)) and various eUICC platforms.

This Test Plan provides a set of test cases to be used for testing the eUICC implementations related to the provisioning system specifications documents [2] & [3]. This document offers an unified test strategy for checking the eUICC component.

1.2 Scope

This document is intended for:

- Parties which develop test tools
- Vendors (eUICC Manufacturers)
- Operators

The Test Plan consists of a set of relevant test cases for testing the eUICC. The only Implementation Under Test (IUT) within this document is the eUICC. The testing scopes developed in this document are:

- Interface compliance testing: Test cases to verify the compliance of the eUICC
- Interfaces System behaviour testing: Test cases to verify the functional behaviour of the eUICC

Each test case specified within this Test Plan refers to one or more requirements.

The Test Plan contains test cases for the following versions of SGP.22:

- GSMA RSP Technical Specification V3.1 [2]

This document includes an applicability table providing an indication whether test cases are relevant for a specific eUICC.

1.3 Definition of Terms

In addition to the terms which are defined below, the terms defined in SGP.22 [2] also apply.

Term	Description
End User	The person using the Device.
Integrated eUICC Test Interface	An external interface provided by its manufacturer for the purpose of testing eUICC functionality.
Standalone Device	A Device which provides all the capabilities to be able to be used in an RSP environment and needs no other Device for the purpose of Remote SIM Provisioning.

Term	Description
Test Plan	Current document describing the test cases that allow the RSP ecosystem to be tested.

1.4 Abbreviations

In addition to the abbreviations which are defined below, the abbreviations defined in SGP.22 [2] also apply.

Abbreviation	Description
APDU	Application Protocol Data Unit
ATR	Answer To Reset
C-APDU	Command APDU
CCID	(USB) Chip Card Interface Device
CERT.CI.SIG	Certificate of the CI for its Public ECDSA Key
CERT.DPauth.SIG	Certificate of the SM-DP+ for its Public ECDSA key used for SM-DP+ authentication
CERT.DPpb.SIG	Certificate of the SM-DP+ for its Public ECDSA key used for Profile Package Binding
CERT.DSauth.SIG	Certificate of the SM-DS for its Public ECDSA key used for SM-DS authentication
CERT.EUICC.SIG	Certificate of the eUICC for its Public ECDSA key
CERT.EUM.SIG	Certificate of the EUM for its Public ECDSA key
DER TLV	Distinguished Encoding Rules - Tag Length Value
DPI	Delegated Platform Identifier
FCP	File Control Parameters
HW	Hardware
IUT	Implementation Under Test
KVN	Key Version Number
OCE	Off-Card Entity
OS	Operating System
PCMP	Profile Content Management Platform
PIR	Profile Installation Result
POR	Proof Of Receipt
R-APDU	Response APDU
RPR	Load RPM Package Result
SK.CI.SIG	Private key of the CI for signing certificates
SK.DPauth.SIG	Private Key of the of SM-DP+ for creating signatures for SM-DP+ authentication
SK.EUICC.SIG	Private key of the eUICC for creating signatures
SK.EUM.SIG	Private key of the EUM for creating signatures
SoC	System on a Chip

Abbreviation	Description
SP	Service Provider
SSD	Supplemental Security Domain
USB	Universal Serial Bus

1.5 Document Cross-references

Ref	Document Number	Title
[1]	SGP.02	GSMA "Remote Provisioning of Embedded UICC Technical specification" V3.1
[2]	SGP.22	RSP Technical Specification V3.1
[3]	SGP.21	RSP Architecture V3.1
[4]	eUICC Profile Package	Trusted Connectivity Alliance (formerly SIMalliance) eUICC Profile Package: Interoperable Format Technical Specification V2.3.1 or later
[5]	ETSI TS 102 221	Smart Cards; UICC-Terminal interface
[6]	GPC_SPE_034	GlobalPlatform Card Specification v.2.3
[7]	ISO/IEC 7816-4:2013	Identification cards – Integrated circuit cards - Part 4: Organization, security and commands for interchange
[8]	RFC 5639	Elliptic Curve Cryptography (ECC) Brainpool Standard Curves and Curve Generation
[9]	ANSSI ECC FRP256V1	Avis relatif aux paramètres de courbes elliptiques définis par l'Etat français. JORF n°0241 du 16 octobre 2011 page 17533. texte n° 30. 2011
[10]	ITU E.118	The international telecommunication charge card
[11]	NIST SP 800-56A	NIST Special Publication SP 800-56A: Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography (Revision 2), May 2013
[12]	3GPP TS 23.003	Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); Numbering, addressing and identification
[13]	ETSI TS 102 225	Secured packet structure for UICC based applications; Release 12
[14]	ETSI TS 102 226	Remote APDU structure for UICC based applications; Release 9
[15]	TS.26	GSMA NFC Handset Requirements V9.0
[16]	ITU-T X.690 (11/2008)	ASN.1 Encoding Rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER) including Corrigendum 1 and 2
[17]	ETSI TS 102 241	Smart cards; UICC Application Programming Interface (UICC API) for Java Card™
[18]	3GPP TS 31.102	Characteristics of the Universal Subscriber Identity Module (USIM) application

Ref	Document Number	Title
[19]	GPC_SPE_095	GlobalPlatform Card - Digital Letter of Approval - Version 1.0
[20]	RFC 2119	Key words for use in RFCs to Indicate Requirement Levels, S. Bradner http://www.ietf.org/rfc/rfc2119.txt
[22]	3GPP TS 23.040	Technical realization of the Short Message Service (SMS)
[23]	TCA Test	Trusted Connectivity Alliance (TCA) eUICC Profile Package: Interoperable Format Test Specification. See Annex K for applicable version.
[24]	RFC 4492	Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS)
[25]	SGP.26	RSP Test Certificates Definition v3.0.X
[26]	Void	Void
[27]	Void	Void
[28]	CCID Rev 1.1	CCID Specification for Integrated Circuit(s) Cards Interface Devices
[29]	SM2 algorithm	ISO/IEC 14888-3:2018 IT Security techniques – Digital signatures with appendix – Part 3: Discrete logarithm based mechanisms

1.6 Conventions

The key words "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", and "MAY" in this document SHALL be interpreted as described in RFC 2119 [20].

2 Testing Rules

2.1 Applicability

2.1.1 Format of the Optional Features Table

The columns in Table 4 have the following meaning:

Column	Meaning
Option	The optional feature supported or not by the implementation.
Mnemonic	The mnemonic column contains mnemonic identifiers for each item.

Table 1: Format of the Optional Features Table

2.1.2 Format of the Applicability Table

The applicability of every test in Table 5 is formally expressed by the use of a Boolean expression defined in the following clause.

The columns in Table 5have the following meaning:

Column	Meaning
Test case	The "Test case" column gives a reference to the test case number detailed in the present document and is required to validate the implementation of the corresponding item in the "Name" column.
Name	In the "Name" column, a short non-exhaustive description of the test is found.
Version	This column indicates which test cases are applicable for the given SGP.22 version. See clause 2.1.3 'Applicability and Notations'.
Test Env.	Test environment used for executing the test case.

Table 2: Format of the Applicability Table

2.1.3 Applicability and Notations

The following notations are used for the Applicability column:

Applicability code	Meaning
M	mandatory - the capability is required to be supported.
N/A	not applicable - in the given context, it is impossible to use the capability.
Ci	conditional - the requirement on the capability depends on the support of other items. "i" is an integer identifying an unique conditional status expression which is defined immediately following the table. For nested conditional expressions, the syntax "IF ... THEN (IF ... THEN ... ELSE...) ELSE ..." is to be used to avoid ambiguities.

Table 3: Applicability and Notations

2.1.4 Optional Features Table

The supplier of the implementation SHALL state the support of possible options in Table 5.

eUICC Options	Mnemonic
The eUICC supports NIST P-256 [11] for signing and for verification (see NOTE 1)	O_E_NIST
The eUICC supports brainpoolP256r1 [8] for signing and for verification (see NOTE 1)	O_E_BRP
The eUICC supports FRP256V1 [9] for signing and for verification (see NOTE 1)	O_E_FRP
The eUICC supports SM2 [XX] for signing and for verification	O_E_SM2
The eUICC supports the LPAe	O_E_LPAe
The eUICC supports LPA Proxy	O_E_LPA_PROXY
The eUICC stores the otPK.eUICC.ECKA / otSK.eUICC.ECKA from previous unsuccessful download attempt for future retry	O_E_REUSE_OTPK
The eUICC can hold two PIR	O_E_2_PIR
The eUICC supports RPM	O_E_RPM
The eUICC supports Enterprise Profiles	O_E_ENTERPRISE

The eUICC terminates EnableProfile and DisableProfile with error "catBusy" when a proactive session is ongoing and the refresh flag is set	O_E_CATBUSY_EN_DIS_REFRESH
The eUICC terminates EnableProfile and DisableProfile with error "catBusy" when a proactive session is ongoing and the refresh Flag is not set	O_E_CATBUSY_EN_DIS_NOREFRESH
The eUICC terminates eUICCMemoryReset with error "catBusy" when a proactive session is ongoing	O_E_CATBUSY_MR
The eUICC is based on an integrated TRE	O_E_INTEGRATED
The eUICC supports variant O for signing	O_VAR_O
The eUICC supports variant Ov3 for signing	O_VAR_OV3
The eUICC supports variant A for signing	O_VAR_A
The eUICC supports variant B for signing	O_VAR_B
The eUICC supports variant C for signing	O_VAR_C
The eUICC supports the HRI server address in the Profile Metadata	O_E_HRI_ADDRESS_IN_PM
The eUICC supports the service description in the Profile Metadata	O_E_SERVICE_DESCRIPTION_IN_PM
The eUICC supports the features defined for Device Change and Profile Recovery	O_E_DEVICE_CHANGE
The eUICC accepts an estimated Profile size in the Profile Metadata	O_E_PROFILE_SIZE_IN_PM
The eUICC provides an estimated Profile size in ES10c.GetProfilesInfo	O_E_PROFILE_SIZE_IN_PROFILE_INFO
The eUICC supports the OS Update capability	O_E_OS_UPDATE
The eUICC supports “catBusy” error code	O_E_CATBUSY
The eUICC supports MEP	O_E_MEP
The eUICC supports MEP-A1	O_E_MEP_A1
The eUICC supports MEP-A2	O_E_MEP_A2
The eUICC supports MEP-B	O_E_MEP_B
The eUICC supports Refresh Flag not set for MEP-B	O_E_MEP_B_NO_REFRESH
The eUICC supports profiles with non-IMSI SUPI Type	O_E_NON_IMSI_SUPI_TYPE
The eUICC supports eUICC Profile Package Specification v3.x	O_E_ADD_PP_VERSIONS
The eUICC supports the extensibility in the DeviceInfo	O_E_DEVICE_INFO_EXTENSIBILITY_SUPPORT
• NOTE 1: This version of test specification extensively tests an eUICC that supports O_E_NIST or O_E_BRP. Any O_E_FRP or O_E_SM2 test cases are for further study.	

Table 4: Options

2.1.5 Applicability Table

Table 5 specifies the applicability of each test case. See clause 2.1.2 for the format of this table.

Test case	Name	V3.1	Test Env.
eUICC Interfaces Compliance Testing			
4.2.1.2.1	TC_eUICC_ATR_And_ISDR_Selection Only the test sequence #1	C006	TE_eUICC
4.2.1.2.1	TC_eUICC_ATR_And_ISDR_Selection Only the test sequence #2	M	TE_eUICC
4.2.1.2.1	TC_eUICC_ATR_And_ISDR_Selection Only the test sequence #3	C325	TE_eUICC
4.2.1.2.1	TC_eUICC_ATR_And_ISDR_Selection Only the test sequences #4 and #7	C318	TE_eUICC
4.2.1.2.1	TC_eUICC_ATR_And_ISDR_Selection Only the test sequences #5 and #8	C328	TE_eUICC
4.2.1.2.1	TC_eUICC_ATR_And_ISDR_Selection Only the test sequences #6 and #9	C327	TE_eUICC
4.2.2.2.1	TC_eUICC_ES6.UpdateMetadata	C319	TE_eUICC
4.2.2.2.2	TC_eUICC_ES6.UpdateMetadata_EnterpriseProfiles	C309	TE_eUICC
4.2.2.2.3	TC_eUICC_ES6.UpdateMetadata_Service_Specific_Data	C319	TE_eUICC
4.2.2.2.4	TC_eUICC_ES6.UpdateMetadata_V3NotificationConfiguration	C319	TE_eUICC
4.2.2.2.5	TC_eUICC_ES6.UpdateMetadata_V3RPM	C305	TE_eUICC
4.2.2.2.6	TC_eUICC_ES6.UpdateMetadata_V3HRIServerAddress	C355	TE_eUICC
4.2.2.2.7	TC_eUICC_ES6.UpdateMetadata_V3LPRConfiguration	C354	TE_eUICC
4.2.2.2.8	TC_eUICC_ES6.UpdateMetadata_V3DeviceChange	C352	TE_eUICC
4.2.3.2.1	TC_eUICC_ES8+.InitialiseSecureChannel	M	TE_eUICC
4.2.4.2.1	TC_eUICC_ES8+.ConfigureISDP	M	TE_eUICC
4.2.5.2.1	TC_eUICC_ES8+.StoreMetadata All test sequences except the sequences #1, #5 and #11	M	TE_eUICC
4.2.5.2.1	TC_eUICC_ES8+.StoreMetadata Only the test sequences #1 and #5	C319	TE_eUICC
4.2.5.2.1	TC_eUICC_ES8+.StoreMetadata Only the test sequence #11	C356	TE_eUICC
4.2.5.2.2	TC_eUICC_ES8+.StoreMetadata_Service_Specific_Data	M	TE_eUICC
4.2.5.2.3	TC_eUICC_ES8+.StoreMetadata_EnterpriseProfiles All test sequences except the sequence #4	C309	TE_eUICC
4.2.5.2.3	TC_eUICC_ES8+.StoreMetadata_EnterpriseProfiles Only the test sequence #4	C349	TE_eUICC
4.2.6.2.1	TC_eUICC_ES8+.ReplaceSessionKeys	M	TE_eUICC
4.2.7.2.1	TC_eUICC_ES8+.LoadProfileElements except test sequence #11 and #12	M	TE_eUICC
4.2.7.2.1	TC_eUICC_ES8+.LoadProfileElements Only the test sequences #11 and #12	C346	TE_eUICC

Test case	Name	V3.1	Test Env.
4.2.8.2.1	TC_eUICC_ES10a.GetEuiccConfiguredAddresses	M	TE_eUICC
4.2.9.2.1	TC_eUICC_ES10a.SetDefaultDpAddress	M	TE_eUICC
4.2.10.2.1	TC_eUICC_ES10b.PrepareDownloadNIST	C001	TE_eUICC
4.2.10.2.2	TC_eUICC_ES10b.PrepareDownloadBRP	C002	TE_eUICC
4.2.10.2.3	TC_eUICC_ES10b.PrepareDownloadFRP	C003	TE_eUICC
4.2.10.2.4	TC_eUICC_ES10b.PrepareDownloadErrorCases	M	TE_eUICC
4.2.11.2.1	TC_eUICC_ES10b.LoadBoundProfilePackageNIST	C001	TE_eUICC
4.2.11.2.2	TC_eUICC_ES10b.LoadBoundProfilePackageBRP	C002	TE_eUICC
4.2.11.2.3	TC_eUICC_ES10b.LoadBoundProfilePackageFRP	C003	TE_eUICC
4.2.11.2.4	TC_eUICC_ES10b.LoadBoundProfilePackage_ErrorCases	M	TE_eUICC
4.2.12.2.1	TC_eUICC_ES10b.GetEUICCChallenge	M	TE_eUICC
4.2.13.2.1	TC_eUICC_ES10b.GetEUICCInfo1 Only test sequences #01 and #02	M	TE_eUICC
4.2.13.2.1	TC_eUICC_ES10b.GetEUICCInfo1 Only test sequence #03	C310	TE_eUICC
4.2.13.2.1	TC_eUICC_ES10b.GetEUICCInfo1 Only test sequence #04	C317	TE_eUICC
4.2.13.2.1	TC_eUICC_ES10b.GetEUICCInfo1 Only test sequence #05	C313	TE_eUICC
4.2.13.2.1	TC_eUICC_ES10b.GetEUICCInfo1 Only test sequence #06	C314	TE_eUICC
4.2.13.2.2	TC_eUICC_ES10b.GetEUICCInfo2_RSP_V2.1	N/A	TE_eUICC
4.2.13.2.3	TC_eUICC_ES10b.GetEUICCInfo2_RSP_V2.2.x	M	TE_eUICC
4.2.13.2.4	TC_eUICC_ES10b.GetEUICCInfo2	C310	TE_eUICC
4.2.13.2.8	TC_eUICC_ES10b.GetEUICCInfo2_RSP_V3.X	M	TE_eUICC
4.2.13.2.9	TC_eUICC_ES10b.GetEUICCInfo2_RSP_V3.x_Integrated_eUICC	C040	TE_eUICC
4.2.14.2.1	TC_eUICC_ES10b.ListNotification All test sequences except the sequence #5	M	TE_eUICC
4.2.14.2.1	TC_eUICC_ES10b.ListNotification Only the test sequence #5	C025	TE_eUICC
4.2.14.2.2	TC_eUICC_ES10b.ListNotification_RPM	C301	TE_eUICC
4.2.15.2.1	TC_eUICC_ES10b.RetrieveNotificationsList All test sequences except the sequences #5 and #15	M	TE_eUICC
4.2.15.2.1	TC_eUICC_ES10b.RetrieveNotificationsList Only the test sequences #5 and #15	C025	TE_eUICC
4.2.16.2.1	TC_eUICC_ES10b.RemoveNotificationFromList All test sequences except the sequence #5	M	TE_eUICC
4.2.18.2.1	TC_eUICC_ES10b.AuthenticateServer_SM-DP+_NIST_Server_Variant_O	C357	TE_eUICC
4.2.18.2.2	TC_eUICC_ES10b.AuthenticateServer_SM-DP+_BRP_Server_Variant_O	C358	TE_eUICC
4.2.18.2.4	TC_eUICC_ES10b.AuthenticateServer_SM-DP+_ErrorCases_Server_Variant_O	M	TE_eUICC

Test case	Name	V3.1	Test Env.
4.2.18.2.5	TC_eUICC_ES10b.AuthenticateServer_SM-DS_BRP_Server_Variant_O	C358	TE_eUICC
4.2.18.2.6	TC_eUICC_ES10b.AuthenticateServer_SM-DS_NIST_Server_Variant_O	C357	TE_eUICC
4.2.18.2.8	TC_eUICC_ES10b.AuthenticateServer_SM-DS_ErrorCases	M	TE_eUICC
4.2.18.2.9	TC_eUICC_ES10b.AuthenticateServerV3_SM-DP+_NIST_Server_Variants_V3	C316	TE_eUICC
4.2.18.2.10	TC_eUICC_ES10b.AuthenticateServerV3_SM-DP+_ErrorCases_V3_Server_Variant_A	C315	TE_eUICC
4.2.18.2.11	TC_eUICC_ES10b.AuthenticateServer_SM-DP+_ErrorCases_V3_Server_Variant_B	C313	TE_eUICC
4.2.18.2.12	TC_eUICC_ES10b.AuthenticateServer_SM-DP+_ErrorCases_V3_Server_Variant_C	C314	TE_eUICC
4.2.18.2.13	TC_eUICC_ES10b.AuthenticateServerV3_SM-DP+_Independently_Of_Server_Variant	M	TE_eUICC
4.2.18.2.14	TC_eUICC_ES10b.AuthenticateServerV3_SM-DS_Independently_Of_Server_Variant	M	TE_eUICC
4.2.19.2.1	TC_eUICC_ES10b.CancelSessionNIST	C001	TE_eUICC
4.2.19.2.2	TC_eUICC_ES10b.CancelSessionBRP	C002	TE_eUICC
4.2.19.2.3	TC_eUICC_ES10b.CancelSessionFRP	C003	TE_eUICC
4.2.19.2.4	TC_eUICC_ES10b.CancelSession_ErrorCase	M	TE_eUICC
4.2.20.2.1	TC_eUICC_ES10c.GetProfilesInfo	M	TE_eUICC
4.2.20.2.2	TC_eUICC_ES10c.GetProfilesInfo_ErrorCases	M	TE_eUICC
4.2.20.2.3	TC_eUICC_ES10c.GetProfilesInfo_MEPA1	C318	TE_eUICC
4.2.20.2.4	TC_eUICC_ES10c.GetProfilesInfo_MEPA2	C328	TE_eUICC
4.2.20.2.5	TC_eUICC_ES10c.GetProfilesInfo_MEPB	C327	TE_eUICC
4.2.21.2.1	TC_eUICC_ES10c.EnableProfile_Case3 All test sequences except the sequences #7, #8, #9 and #10	C319	TE_eUICC
4.2.21.2.1	TC_eUICC_ES10c.EnableProfile_Case3 Only the test sequences #7 and #9	C033	TE_eUICC
4.2.21.2.1	TC_eUICC_ES10c.EnableProfile_Case3 Only the test sequences #8 and #10	C037	TE_eUICC
4.2.21.2.2	TC_eUICC_ES10c.EnableProfile_ErrorCases_Case3 All test sequences except the sequences #7 and #8	C319	TE_eUICC
4.2.21.2.2	TC_eUICC_ES10c.EnableProfile_ErrorCases_Case3 Only the test sequence #7	C036	TE_eUICC
4.2.21.2.2	TC_eUICC_ES10c.EnableProfile_ErrorCases_Case3 Only the test sequence #8	C032	TE_eUICC
4.2.21.2.3	TC_eUICC_ES10c.EnableProfile_Case4 All test sequences except the sequences #7, #8, #9 and #10	C319	TE_eUICC
4.2.21.2.3	TC_eUICC_ES10c.EnableProfile_Case4 Only the test sequences #7 and #9	C033	TE_eUICC
4.2.21.2.3	TC_eUICC_ES10c.EnableProfile_Case4	C037	TE_eUICC

Test case	Name	V3.1	Test Env.
	Only the test sequence #8 and #10		
4.2.21.2.4	TC_eUICC_ES10c.EnableProfile_ErrorCases_Case4 All test sequences except the sequences #5, #6, #7 and #8	C319	TE_eUICC
4.2.21.2.4	TC_eUICC_ES10c.EnableProfile_ErrorCases_Case4 Only the test sequences #5 and #6	C319	TE_eUICC
4.2.21.2.4	TC_eUICC_ES10c.EnableProfile_ErrorCases_Case4 Only the test sequence #7	C036	TE_eUICC
4.2.21.2.4	TC_eUICC_ES10c.EnableProfile_ErrorCases_Case4 Only the test sequence #8	C032	TE_eUICC
4.2.21.2.5	TC_eUICC_ES10c.EnableProfile_ErrorCases_Case4_catBusySupported	C032	TE_eUICC
4.2.21.2.6	TC_eUICC_ES10c.EnableProfile_Case4_catBusyNotSupported	C033	TE_eUICC
4.2.21.2.7	TC_eUICC_ES10c.EnableProfile_ErrorCases_Case3_catBusySupported	C032	TE_eUICC
4.2.21.2.8	TC_eUICC_ES10c.EnableProfile_Case4_MEPA1 All test sequences except the sequences #7, #8, #15, #16, #17 and #18	C318	TE_eUICC
4.2.21.2.8	TC_eUICC_ES10c.EnableProfile_Case4_MEPA1 Only the test sequences #7, #15, #16, #17 and #18	C333	TE_eUICC
4.2.21.2.8	TC_eUICC_ES10c.EnableProfile_Case4_MEPA1 Only the test sequence #8	C329	TE_eUICC
4.2.21.2.9	TC_eUICC_ES10c.EnableProfile_ErrorCases_Case4_MEPA1 All test sequences except the sequences #5, #6 and #11	C318	TE_eUICC
4.2.21.2.9	TC_eUICC_ES10c.EnableProfile_ErrorCases_Case4_MEPA1 Only the test sequences #6 and #11	C332	TE_eUICC
4.2.21.2.9	TC_eUICC_ES10c.EnableProfile_ErrorCases_Case4_MEPA1 Only the test sequence #5	C336	TE_eUICC
4.2.21.2.10	TC_eUICC_ES10c.EnableProfile_Case4_MEPA2	C328	TE_eUICC
4.2.21.2.11	TC_eUICC_ES10c.EnableProfile_ErrorCases_Case4_MEPA2 All test sequences except the sequences #5 and #10	C328	TE_eUICC
4.2.21.2.11	TC_eUICC_ES10c.EnableProfile_ErrorCases_Case4_MEPA2 Only the test sequence #5 and #10	C330	TE_eUICC
4.2.21.2.12	TC_eUICC_ES10c.EnableProfile_Case4_MEPB All test sequences except the sequences #5, #6, #7, #8, #13, #14, #15 and #16	C327	TE_eUICC
4.2.21.2.12	TC_eUICC_ES10c.EnableProfile_Case4_MEP_B Only the test sequences #5, #6, #13, #14	C345	TE_eUICC
4.2.21.2.12	TC_eUICC_ES10c.EnableProfile_Case4_MEPB Only the test sequences #7, #15 and #16	C335	TE_eUICC
4.2.21.2.12	TC_eUICC_ES10c.EnableProfile_Case4_MEPB Only the test sequence #8	C337	TE_eUICC
4.2.21.2.13	TC_eUICC_ES10c.EnableProfile_ErrorCases_Case4_MEPB All test sequences except the sequences #5, #6 and #11	C327	TE_eUICC

Test case	Name	V3.1	Test Env.
4.2.21.2.13	TC_eUICC_ES10c.EnableProfile_ErrorCases_Case4_MEPB Only the test sequences #6 and #11	C334	TE_eUICC
4.2.21.2.13	TC_eUICC_ES10c.EnableProfile_ErrorCases_Case4_MEPB Only the test sequence #5	C338	TE_eUICC
4.2.22.2.1	TC_eUICC_ES10c.DisableProfile_Case3 All test sequences except the sequences #7, sequence #8, #9 and #10	C319	TE_eUICC
4.2.22.2.1	TC_eUICC_ES10c.DisableProfile_Case3 Only the test sequences #7 and #9	C033	TE_eUICC
4.2.22.2.1	TC_eUICC_ES10c.DisableProfile_Case3 Only the test sequences #8 and #10	C037	TE_eUICC
4.2.22.2.2	TC_eUICC_ES10c.DisableProfile_ErrorCases_Case3 All test sequences except the sequences #7 and #8	C319	TE_eUICC
4.2.22.2.2	TC_eUICC_ES10c.DisableProfile_ErrorCases_Case3 Only the test sequence #7	C036	TE_eUICC
4.2.22.2.2	TC_eUICC_ES10c.DisableProfile_ErrorCases_Case3 Only the test sequence #8	C032	TE_eUICC
4.2.22.2.3	TC_eUICC_ES10c.DisableProfile_Case4 All test sequences except the sequences #7, #8, #9 and #10	C319	TE_eUICC
4.2.22.2.3	TC_eUICC_ES10c.DisableProfile_Case4 Only the test sequences #7 and #9	C033	TE_eUICC
4.2.22.2.3	TC_eUICC_ES10c.DisableProfile_Case4 Only the test sequences #8 and #10	C037	TE_eUICC
4.2.22.2.4	TC_eUICC_ES10c.DisableProfile_ErrorCases_Case4 All test sequences except the sequences #5, #6, #7 and #8	C319	TE_eUICC
4.2.22.2.4	TC_eUICC_ES10c.DisableProfile_ErrorCases_Case4 Only the test sequences #5 and #6	C319	TE_eUICC
4.2.22.2.4	TC_eUICC_ES10c.DisableProfile_ErrorCases_Case4 Only the test sequence #7	C036	TE_eUICC
4.2.22.2.4	TC_eUICC_ES10c.DisableProfile_ErrorCases_Case4 Only the test sequence #8	C032	TE_eUICC
4.2.22.2.5	TC_eUICC_ES10c.DisableProfile_Case4_MEPA1 All test sequences except the sequences #7, #8, #15, #16, #17 and #18	C318	TE_eUICC
4.2.22.2.5	TC_eUICC_ES10c.DisableProfile_Case4_MEPA1 Only the test sequences #7, #15, #16, #17 and #18	C333	TE_eUICC
4.2.22.2.5	TC_eUICC_ES10c.DisableProfile_Case4_MEPA1 Only the test sequence #8	C329	TE_eUICC
4.2.22.2.6	TC_eUICC_ES10c.DisableProfile_ErrorCases_Case4_MEPA1 All test sequences except the sequences #5, #6 and #11	C318	TE_eUICC
4.2.22.2.6	TC_eUICC_ES10c.DisableProfile_ErrorCases_Case4_MEPA1 Only the test sequences #6 and #11	C332	TE_eUICC
4.2.22.2.6	TC_eUICC_ES10c.DisableProfile_ErrorCases_Case4_MEPA1 Only the test sequence #5	C336	TE_eUICC

Test case	Name	V3.1	Test Env.
4.2.22.2.7	TC_eUICC_ES10c.DisableProfile_Case4_MEPA2 All test sequences except the sequence #3	C328	TE_eUICC
4.2.22.2.7	TC_eUICC_ES10c.DisableProfile_Case4_MEPA2 Only the test sequence #3	C331	TE_eUICC
4.2.22.2.8	TC_eUICC_ES10c.DisableProfile_ErrorCases_Case4_MEPA2 All test sequences except the sequences #5 and #10	C328	TE_eUICC
4.2.22.2.8	TC_eUICC_ES10c.DisableProfile_ErrorCases_Case4_MEPA2 Only the test sequences #5 and #10	C330	TE_eUICC
4.2.22.2.9	TC_eUICC_ES10c.DisableProfile_Case4_MEPB All test sequences except the sequences #5, #6, #7, #8, #13, #14, #15 #16, #17 and #18	C327	TE_eUICC
4.2.22.2.9	TC_eUICC_ES10c.DisableProfile_Case4_MEPB All test sequences except the sequences #5, #6, #13, #14	C345	TE_eUICC
4.2.22.2.9	TC_eUICC_ES10c.DisableProfile_Case4_MEPB Only the test sequences #7, #15, #16, #17 and #18	C335	TE_eUICC
4.2.22.2.9	TC_eUICC_ES10c.DisableProfile_Case4_MEPB Only the test sequence #8	C337	TE_eUICC
4.2.22.2.10	TC_eUICC_ES10c.DisableProfile_ErrorCases_Case4_MEPB All test sequences except the sequences #5, #6 and #11	C327	TE_eUICC
4.2.22.2.10	TC_eUICC_ES10c.DisableProfile_ErrorCases_Case4_MEPB Only the test sequences #6 and #11	C334	TE_eUICC
4.2.22.2.10	TC_eUICC_ES10c.DisableProfile_ErrorCases_Case4_MEPB Only the test sequence #5	C338	TE_eUICC
4.2.23.2.1	TC_eUICC_ES10c.DeleteProfile_Case3	M	TE_eUICC
4.2.23.2.2	TC_eUICC_ES10c.DeleteProfile_ErrorCases_Case3	M	TE_eUICC
4.2.23.2.3	TC_eUICC_ES10c.DeleteProfile_Case4	M	TE_eUICC
4.2.23.2.4	TC_eUICC_ES10c.DeleteProfile_ErrorCases_Case4	M	TE_eUICC
4.2.24.2.1	TC_eUICC_ES10c.eUICCMemoryReset All test sequences except the sequences #2, #5 and #6	C319	TE_eUICC
4.2.24.2.1	TC_eUICC_ES10c.eUICCMemoryReset Only the test sequence #2	C319	TE_eUICC
4.2.24.2.1	TC_eUICC_ES10c.eUICCMemoryReset Only the test sequences #5 and #6	C039	TE_eUICC
4.2.24.2.2	TC_eUICC_ES10c.eUICCMemoryReset_ErrorCases Only the test sequence #1	C038	TE_eUICC
4.2.24.2.2	TC_eUICC_ES10c.eUICCMemoryReset_ErrorCases Only the test sequence #2	C319	TE_eUICC
4.2.24.2.3	TC_eUICC_ES10c.eUICCMemoryReset_MEPA1 All test sequences except the sequences #6, #7, #8 and #9	C318	TE_eUICC
4.2.24.2.3	TC_eUICC_ES10c.eUICCMemoryReset_MEPA1 Only the test sequences #6, #7, #8 and #9	C326	TE_eUICC
4.2.24.2.4	TC_eUICC_ES10c.eUICCMemoryReset_ErrorCases_MEPA1 All test sequences except the sequence #2	C318	TE_eUICC
4.2.24.2.4	TC_eUICC_ES10c.eUICCMemoryReset_ErrorCases_MEPA1	C324	TE_eUICC

Test case	Name	V3.1	Test Env.
	Only the test sequence #2		
4.2.24.2.5	TC_eUICC_ES10c.eUICCMemoryReset_MEPA2 All test sequences except the sequences #6, #7, #8 and #9	C328	TE_eUICC
4.2.24.2.5	TC_eUICC_ES10c.eUICCMemoryReset_MEPA2 Only the test sequences #6, #7, #8 and #9	C340	TE_eUICC
4.2.24.2.6	TC_eUICC_ES10c.eUICCMemoryReset_ErrorCases_MEPA2 All test sequences except the sequence #2	C328	TE_eUICC
4.2.24.2.6	TC_eUICC_ES10c.eUICCMemoryReset_ErrorCases_MEPA2 Only the test sequence #2	C339	TE_eUICC
4.2.24.2.7	TC_eUICC_ES10c.eUICCMemoryReset_MEPB All test sequences except the sequences #6, #7, #8 and #9	C327	TE_eUICC
4.2.24.2.7	TC_eUICC_ES10c.eUICCMemoryReset_MEPB Only the test sequences #6, #7, #8 and #9	C342	TE_eUICC
4.2.24.2.8	TC_eUICC_ES10c.eUICCMemoryReset_ErrorCases_MEPB All test sequences except the sequence #2	C327	TE_eUICC
4.2.24.2.8	TC_eUICC_ES10c.eUICCMemoryReset_ErrorCases_MEPB Only the test sequence #2	C341	TE_eUICC
4.2.25.2.1	TC_eUICC_ES10c.GetEID	M	TE_eUICC
4.2.26.2.1	TC_eUICC_ES10c.SetNickname	M	TE_eUICC
4.2.27.2.1	TC_eUICC_ES10b.GetRAT Only the test Sequence #1	C319	TE_eUICC
4.2.27.2.1	TC_eUICC_ES10b.GetRAT Only the test Sequence #2	C319	TE_eUICC
4.2.28.2.1	TC_eUICC_ES10b.LoadRPMPackage_EnableProfile All test sequences except the sequences #5 and #7	C301	TE_eUICC
4.2.28.2.1	TC_eUICC_ES10b.LoadRPMPackage_EnableProfile Only the test sequences #5 and #7	C305	TE_eUICC
4.2.28.2.2	TC_eUICC_ES10b.LoadRPMPackage_DisableProfile All test sequences except the sequence #6	C301	TE_eUICC
4.2.28.2.2	TC_eUICC_ES10b.LoadRPMPackage_DisableProfile Only the test sequence #6	C305	TE_eUICC
4.2.28.2.3	TC_eUICC_ES10b.LoadRPMPackage_DeleteProfile	C301	TE_eUICC
4.2.28.2.4	TC_eUICC_ES10b.LoadRPMPackage_ListProfileInfo	C305	TE_eUICC
4.2.28.2.5	TC_eUICC_ES10b.LoadRPMPackage_UpdateMetadata All test sequences except the sequence #9	C305	TE_eUICC
4.2.28.2.5	TC_eUICC_ES10b.LoadRPMPackage_UpdateMetadata Only the test sequence #9	C301	TE_eUICC
4.2.28.2.6	TC_eUICC_ES10b.LoadRPMPackage_ContactPCMP All test sequences except the sequence #4	C303	TE_eUICC
4.2.28.2.6	TC_eUICC_ES10b.LoadRPMPackage_ContactPCMP Only the test sequence #4	C350	TE_eUICC
4.2.28.2.7	TC_eUICC_ES10b.LoadRPMPackage_Multiple_RPM All test sequences except the sequences #3, #4, #5, and #7	C301	TE_eUICC

Test case	Name	V3.1	Test Env.
4.2.28.2.7	TC_eUICC_ES10b.LoadRPMPackage_Multiple_RPM Only the test sequences #3, #4 and #5	C350	TE_eUICC
4.2.28.2.7	TC_eUICC_ES10b.LoadRPMPackage_Multiple_RPM Only the test sequence #7	C303	TE_eUICC
4.2.28.2.8	TC_eUICC_ES10b.LoadRPMPackage_ErrorCases	C301	TE_eUICC
4.2.28.2.9	TC_eUICC_ES10b.LoadRPMPackage_Enterprise_Profiles All test sequences except the sequence #13	C302	TE_eUICC
4.2.28.2.9	TC_eUICC_ES10b.LoadRPMPackage_Enterprise_Profiles Only the test sequence #13	C351	TE_eUICC
4.2.28.2.10	TC_eUICC_ES10b.LoadRPMPackage_CatBusy	C304	TE_eUICC
Procedure - Behaviour Testing			
5.2.1.2.1	TC_eUICC_PrepareDownload_Retry_ReuseOTKeys	C019	TE_eUICC
5.2.1.2.2	TC_eUICC_PrepareDownload_Retry_NewOTKeys	C020	TE_eUICC
5.2.2.2.1	TC_eUICC_ForbiddenPPRs	C319	TE_eUICC
5.2.3.2.1	TC_eUICC_GetProfilesInfo_GetRAT_RSPSession	C319	TE_eUICC
5.2.4.2.1	TC_eUICC_Default_FileSystem	M	TE_eUICC
5.2.5.2.1	TC_eUICC_DeleteProfile_ISDP_And_Components	M	TE_eUICC
5.2.6.2.1	TC_eUICC_EnableProfile_Twice_Notifications	M	TE_eUICC
5.2.7.2.1	TC_eUICC_DisableProfile_ApplicationManagement	M	TE_eUICC
5.2.8.2.1	TC_eUICC_Enable_Disable_Delete_Notifications	M	TE_eUICC
Test Specifications			
Annex K	TCA eUICC Profile Package Test Specification	M	See Annex K

Table 5: Applicability of Tests

Conditional item	Condition
C001	IF (O_E_NIST) THEN M ELSE N/A
C002	IF (O_E_BRP) THEN M ELSE N/A
C006	IF (NOT O_E_LPAe) THEN M ELSE N/A
C019	IF (O_E_REUSE_OTPK) THEN M ELSE N/A
C020	IF (NOT O_E_REUSE_OTPK) THEN M ELSE N/A
C025	IF (O_E_2_PIR) THEN M ELSE N/A
C032	IF (O_E_CATBUSY AND NOT O_E_MEP) THEN M ELSE N/A
C033	IF (NOT O_E_CATBUSY AND NOT O_E_MEP) THEN M ELSE N/A
C036	IF (O_E_CATBUSY_EN_DIS_NOREFRESH AND NOT O_E_MEP) THEN M ELSE N/A
C037	IF (NOT O_E_CATBUSY_EN_DIS_NOREFRESH AND NOT O_E_MEP) THEN M ELSE N/A
C038	IF (O_E_CATBUSY_MR AND NOT O_E_MEP) THEN M ELSE N/A
C039	IF (NOT O_E_CATBUSY_MR AND NOT O_E_MEP) THEN M ELSE N/A
C040	IF O_E_INTEGRATED THEN M ELSE N/A
Conditions applicable to SGP.23 v3.1 onwards	

Conditional item	Condition
C301	IF (O_E_RPM) THEN M ELSE N/A
C302	IF (O_E_RPM AND O_E_ENTERPRISE) THEN M ELSE N/A
C303	IF (O_E_RPM AND O_E_LPA_PROXY) THEN M ELSE N/A
C304	IF (O_E_RPM AND O_E_CAT_BUSY) THEN M ELSE N/A
C305	IF (O_E_RPM AND NOT O_E_MEP) THEN M ELSE N/A
C306	IF (O_E_LPAe) THEN M ELSE N/A
C309	IF (O_E_ENTERPRISE) THEN M ELSE N/A
C310	IF (O_VAR_O) THEN M ELSE N/A
C311	IF (O_VAR_OV3) THEN M ELSE N/A
C312	IF (O_VAR_A) THEN M ELSE N/A
C313	IF (O_VAR_B) THEN M ELSE N/A
C314	IF (O_VAR_C) THEN M ELSE N/A
C315	IF (O_VAR_A OR O_VAR_B OR O_VAR_C) THEN M ELSE N/A
C316	IF ((O_VAR_A OR O_VAR_B OR O_VAR_C OR O_VAR_OV3) AND O_E_NIST) THEN M ELSE N/A
C317	IF (O_VAR_A OR O_VAR_B OR O_VAR_C OR O_VAR_OV3) THEN M ELSE N/A
C318	IF (O_E_MEP_A1) THEN M ELSE N/A
C319	IF (NOT O_E_MEP) THEN M ELSE N/A
C320	IF (NOT O_E_CATBUSY_EN_DIS_NOREFRESH) THEN M ELSE N/A
C321	IF (NOT O_E_CATBUSY) THEN M ELSE N/A
C322	IF (O_E_CATBUSY) THEN M ELSE N/A
C323	IF (O_E_CATBUSY_EN_DIS_NOREFRESH) THEN M ELSE N/A
C324	IF (O_E_MEP_A1 AND O_E_CATBUSY_MR) THEN M ELSE N/A
C325	IF O_E_LPAe THEN M ELSE N/A
C326	IF (O_E_MEP_A1 AND NOT O_E_CATBUSY_MR) THEN M ELSE N/A
C327	IF (O_E_MEP_B) THEN M ELSE N/A
C328	IF (O_E_MEP_A2) THEN M ELSE N/A
C329	IF (O_E_MEP_A1 AND NOT O_E_CATBUSY_EN_DIS_NOREFRESH) THEN M ELSE N/A
C330	IF (O_E_MEP_A2 AND O_E_CATBUSY_EN_DIS_NOREFRESH) THEN M ELSE N/A
C331	IF (O_E_MEP_A2 AND NOT O_E_CATBUSY_EN_DIS_NOREFRESH) THEN M ELSE N/A
C332	IF (O_E_MEP_A1 AND O_E_CATBUSY) THEN M ELSE N/A
C333	IF (O_E_MEP_A1 AND NOT O_E_CATBUSY) THEN M ELSE N/A
C334	IF (O_E_MEP_B AND O_E_CATBUSY) THEN M ELSE N/A
C335	IF (O_E_MEP_B AND NOT O_E_CATBUSY) THEN M ELSE N/A

Conditional item	Condition
C336	IF (O_E_MEP_A1 AND O_E_CATBUSY_EN_DIS_NOREFRESH) THEN M ELSE N/A
C337	IF (O_E_MEP_B_NO_REFRESH AND NOT O_E_CATBUSY_EN_DIS_NOREFRESH) THEN M ELSE N/A
C338	IF (O_E_MEP_B_NO_REFRESH AND O_E_CATBUSY_EN_DIS_NOREFRESH) THEN M ELSE N/A
C339	IF (O_E_MEP_A2 AND O_E_CATBUSY_MR) THEN M ELSE N/A
C340	IF (O_E_MEP_A2 AND NOT O_E_CATBUSY_MR) THEN M ELSE N/A
C341	IF (O_E_MEP_B AND O_E_CATBUSY_MR) THEN M ELSE N/A
C342	IF (O_E_MEP_B AND NOT O_E_CATBUSY_MR) THEN M ELSE N/A
C345	IF (O_E_MEP_B_NO_REFRESH) THEN M ELSE N/A
C346	IF (O_E_NON_IMSI_SUPI_TYPE AND O_E_ADD_PP_VERSIONS AND O_E_DEVICE_INFO_EXTENSIBILITY_SUPPORT) THEN M ELSE N/A
C347	IF (O_E_DEVICE_CHANGE) THEN M ELSE N/A
C348	IF (O_E_HRI_ADDRESS_IN_PM) THEN M ELSE N/A
C349	IF (O_E_ENTERPRISE AND NOT O_E_MEP) THEN M ELSE N/A
C350	IF (O_E_RPM AND O_E_LPA_PROXY AND NOT O_E_MEP) THEN M ELSE N/A
C351	IF (O_E_RPM AND O_E_ENTERPRISE AND NOT_O_E_MEP) THEN M ELSE N/A
C352	IF (O_E_DEVICE_CHANGE AND NOT O_E_MEP) THEN M ELSE N/A
C353	IF (O_E_LPA_PROXY) THEN M ELSE N/A
C354	IF (O_E_LPA_PROXY AND NOT O_E_MEP) THEN M ELSE N/A
C355	IF (O_E_HRI_ADDRESS_IN_PM AND NOT O_E_MEP) THEN M ELSE N/A
C356	IF (NOT O_E_ENTERPRISE) THEN M ELSE N/A
C357	IF (O_VAR_O and O_E_NIST) THEN M ELSE N/A
C358	IF (O_VAR_O and O_E_BRP) THEN M ELSE N/A

Table 6: Conditional Items Referenced by Table 5

Note: Conditions C0XX which are missing in Table 6 are present in an earlier version of SGP.23 but are not used in the current version. Conditions RFU are for future use in the next release of this specification.

2.2 General Consideration

This section contains some general considerations about the test cases defined in this document. Note that some external test specifications are referred to in chapter 7. Consequently, the following sub sections SHALL only apply for test cases defined in sections 4 and 5 and 6.

2.2.1 Test Case Definition

Test descriptions are independent.

For each test described in this document, a chapter provides a general description of the initial conditions applicable for the whole test. This description is completed by specific configurations to each individual sub-case.

It is implicitly assumed that the IUT SHALL be compliant with the initial states described in Annex G. An initial state SHALL be considered as a pre-requisite to execute all the test cases described in this Test Plan.

After completing the test, the configuration is reset before the execution of the following test.

2.2.2 Test Cases Format

Here is an explanation of the way to define the test cases in chapters 4, 5 and 6.

4.X.Y.Z Test Cases

4.X.Y.Z.1 TC_IUT_TestName1

General Initial Conditions	
Entity	**Description of the general initial condition**
Entity1	Test case - general condition 1
Entity2	Test case - general condition 2

Test Sequence #01: Short Description

Description of the aim of the test sequence N°1

Initial Conditions	
Entity	**Description of the initial condition**
Entity1	Test sequence N°1 - initial condition 1
Entity2	Test sequence N°1 - initial condition 2

Step	Direction	Sequence / Description	Expected result
IC1	Entity1 → Entity2	Command or Message to send from Entity1 to Entity2	Expected result N°1.1
1	Entity1 → Entity2	Command or Message to send from Entity1 to Entity2	1- expected result N°1.2 2- expected result N°1.3
2	Entity2 → Entity3	Command or Message to send from Entity2 to Entity3	

Test Sequence #02

Description of the aim of the test sequence N°2

Step	Direction	Sequence / Description	Expected result
1	Entity1 → Entity2	Command or Message to send from Entity1 to Entity2	

2	Entity2 → Entity3	Command or Message to send from Entity2 to Entity3	1- expected result N°2.1 2- expected result N°2.2

4.X.Y.Z.2 TC_IUT_TestName2

…

The test cases TC_IUT_TestName1 and TC_IUT_TestName2 are referenced in Table 5 that allows indicating the applicability of the tests.

In the test case TC_IUT_TestName1, the requirements REQ1 and REQ2 are respectively covered by the test sequences #01 and #02.

The test sequence #01 SHALL be executed if and only if these conditions are met:

- Test case - general condition 1
- Test case - general condition 2
- Test sequence N°1 - initial condition 1
- Test sequence N°1 - initial condition 2

The test sequence #02 SHALL be executed if and only if these conditions are met:

- Test case - general condition 1
- Test case - general condition 2

The tables defining the different initial conditions are optional.

Initial Conditions are intended to be reached dynamically using the Test Tool when possible.

No additional operation SHALL be done prior to the test sequence besides those indicated in the Initial Conditions (e.g. no other Profiles SHALL be present on the eUICC besides those defined in the Initial Conditions).

In the test sequence #01:

- the step IC1 corresponds to an additional Initial Condition
- in the step N°1, if the expected results N°1 and N°2 are validated, the requirement REQ1 (or a part of the REQ1) SHALL be considered as implemented

Note that all initial states (described in Annex G) SHALL be implemented by the IUT whatever the test cases to execute.

In addition, following 2.2.2 sub sections present all information (e.g. Methods, Constants…) that MAY be referenced in test sequences.

After execution of each test sequence a clean-up procedure (CU) SHALL be executed to restore the IUT to the Common Initial State as defined in Annex G.

Sections 2.2.2.0 to 2.2.2.6 specify how to factor text among test sequences.

2.2.2.0 Conditional Execution of Test Sequence Steps

Some steps may be relevant only if the IUT supports the corresponding option, even if the intent and general script of the enclosing Test Sequence is applicable to several options.

In that case the Test Sequence uses the following construct:

Step	Direction	Sequence / Description	Expected result
1	Entity1 → Entity2	Command or Message to send from Entity1 to Entity2	Expected result N°1.1
IF (optionsCombination)			
2	Entity1 → Entity2	Command or Message to send from Entity1 to Entity2	1- expected result N°2.1 2- expected result N°2.2
3	Entity2 → Entity3	Command or Message to send from Entity2 to Entity3	
ENDIF			
4	Entity1 → Entity2	Command or Message to send from Entity1 to Entity2	

The test tool SHALL execute the steps enclosed between keywords IF and ENDIF if and only if the expression optionsCombination has value ‘true’.

Every IF is closed by the first ENDIF that comes after it. Nesting is not supported.

The expression `optionsCombination` is a combination of one or more Boolean values that can be:

- options defined in section 2.1.4,
- comparisons on the value of IUT_SETTINGS
- comparisons on the value of dynamic content defined in see Annex B

and using the boolean operators OR, AND, NOT, with brackets to avoid any ambiguity on operator precedence.

2.2.2.1 Methods and Procedures

A method factors the computation of a single value, based on parameters passed by a test step in a test sequence.

A method is referenced as follow:

- `MTD_NAME_OF_THE_METHOD(PARAM1, PARAM2…)`

The key word “NO_PARAM” SHALL be set in method call if the related optional parameter is not used.

All methods and their related parameters are described in Annex C.1.

A procedure factors a generic sub-sequence of test steps. It may take parameters to exert these test steps on different contexts or with different values.

A procedure is referenced as follow:

- `PROC_NAME_OF_THE_PROCEDURE` if it doesn't take parameters
- `PROC_NAME_OF_THE_PROCEDURE(PARAM1, PARAM2…)` if it takes parameters. In that case, the key word “NO_PARAM” SHALL be set if the related optional parameter is not used.

All procedures are described in Annex C.2.

The implementation of these methods and procedures is under the responsibility of the test tool providers.

2.2.2.2 Constants and Dynamic Content

A constant (e.g. text, ASN.1 structure, hexadecimal string, icon, URI, integer, EID, AID…) is referenced as follow:

- `#NAME_OF_THE_CONSTANT`

All constants are defined in Annex A.

When provided as an ASN.1 value notation, a constant SHALL be encoded in DER TLV (as specified in ITU-T X.690 [16]) by the test tool.

A dynamic content (e.g. TLV, ASN.1 structure, signature, integer, AID, one-time key pair…) is referenced as follow:

- `<NAME_OF_THE_VARIABLE>`

All dynamic contents are defined in Annex B.

A dynamic content is either generated by an IUT or by a test tool provider.

2.2.2.3 Requests and Responses

An ASN.1 or a JSON request is referenced as follow:

- `#NAME_OF_THE_REQUEST`

An ASN.1 or a JSON response is referenced as follows:

- `#R_NAME_OF_THE_RESPONSE`

Each ASN.1 or JSON request and response MAY refer to a constant or a dynamic content. All these structures are defined in Annex D.

When provided as an ASN.1 value notation, a request or a response SHALL be encoded in DER TLV (as specified in ITU-T X.690 [16]) by the test tool.

When an ASN.1 element definition contains three points (i.e. "…"), it means that fields MAY be present but SHALL not be checked by the test tool.

In the following example, several fields MAY be part of the `ProfileInfoListResponse` but only the `profileNickname` SHALL be verified.

```
resp ProfileInfoListResponse ::=
  profileInfoListOk :{
    {
       ...
       profileNickname #NICKNAME
       ...
    }
}
```

This rule applies aslo for Constants definition.

Some ASN.1 SEQUENCE components have a DEFAULT value (for example, `profileClass` in `StoreMetadataRequest`). In this specification, when values are specified in ASN.1 syntax and the DEFAULT value is intended, two different formulations (both of which are valid) may be used:

- the relevant component is specified with the DEFAULT value;
- the relevant component is missing entirely.

These are logically equivalent and lead to the same DER encoding. In both cases, the following rules apply:

- When the test tool is sending the DER value, it SHALL NOT include the component (as per DER rules).
- When the test tool is checking a received DER value from the entity under test, it SHALL check that the component is NOT present.

Test tools SHALL consider two BIT STRINGs to be equivalent if the BIT STRINGs have the same DER encoding. For example, '0101'B shall be considered to be equivalent to '010100'B.

NOTE: this is equivalent to removing any trailing zero bits from the BIT STRINGs in "bstring" notation (e.g. '010100'B → '0101'B) and then comparing the strings textually.

NOTE: according to the DER format, the encoding of transmitted values will remove the trailing zeroes. The definition above allows for values which are specified using ASN.1 value notation and are not transmitted, such as values specified in the Annexes of the current document, including IUT settings which might be specified by a user of the current document and may contain trailing zeroes in the ASN.1 value notation.

2.2.2.4 APDUs

A C-APDU is referenced as follow:

- [NAME_OF_THE_CAPDU]

All C-APDUs are defined in Annex D.4, except the APDU STORE DATA which is constructed by one of the methods MTD_STORE_DATA_xxx, see annex C.1.

An R-APDU is referenced as follow:

[R_NAME_OF_THE_RAPDU]
All R-APDUs are defined in Annex D.4.

Each APDU MAY refer to a constant or a dynamic content.

The APDU TERMINAL RESPONSE SHALL be dynamically generated by the test tool according to the related proactive command. Therefore, this particular command is not referenced with brackets in this specification. If not explicitly defined in the step, the general result SHALL be set by default to “Command performed successfully” (i.e. 0x83 01 00).

2.2.2.5 Profiles

In order to execute the test cases described in this document, Operational, Test and Provisioning Profiles are necessary. All these Profiles are defined in Annex E with the Profile Metadata content and the corresponding Profile Package as defined in the eUICC Profile Package Specification [4].

A Profile is referenced as follow:

- PROFILE_OPERATIONALx with x the identifier of the Operational Profile

or

- PROFILE_TESTx with x the identifier of the Test Profile

or

- PROFILE_PROVISIONINGx with x the identifier of the Provisioning Profile

NOTE: Test Profiles and Provisioning Profiles are out of the scope of this version of test specification.

2.2.2.6 IUT Settings

For the purpose of some test cases, eUICC manufacturers need to give some information related to their products to the test tools providers (e.g. supported Java Card version).

An IUT setting is referenced as follow:

- #IUT_NAME_OF_SETTING

All these settings are defined in Annex F.

2.2.2.7 Referenced Requirements

All requirements referenced in this document by their identifiers are present and described in Annex I. These requirements have been extracted from the specifications:

- GSMA RSP Technical Specification [2]
- GSMA RSP Architecture [3]

2.2.3 General Rules for eUICC Testing

2.2.3.1 Default Profile Downloading process

By default, when an Operational Profile needs to be downloaded on the eUICC (e.g. as mentioned in an initial condition), the following rules apply except if it is differently defined in the Test Case.

In order to execute the Common Mutual Authentication procedure and the Sub-procedure Profile Download and Installation (End User Confirmation), the following requests SHALL be sent by the Test Tool:

- `#GET_EUICC_INFO1` and `#GET_EUICC_CHALLENGE`
- `MTD_AUTHENTICATE_SMDP`
    - with the SM-DP+ address `#TEST_DP_ADDRESS1`
    - with `#CTX_PARAMS1_MATCH_ID`
    - If `euiccCiPKIdListForSigningV3` is present in `eUICCInfo1`, select the eSIM CA RootCA Public Key, following the priority order given by the eUICC in `euiccCIPKIdListForSigningV3`. The `paramNewCertVariant` SHALL be set to TRUE.
    - If `euiccCiPKIdListForSigningV3` is not present in `eUICCInfo1`, select the eSIM CA RootCA Public Key, following the priority order given by the eUICC in `euiccCIPKIdListForSigning`. The `paramNewCertVariant` SHALL be set to FALSE.
    - with the `#CERT_S_SM_DPauth_SIG` leading to the same CI as the one chosen for signing
    - with the `#CRL_LIST` included
- `#PREP_DOWNLOAD_NO_CC`
    - with the `#CERT_S_SM_DPpb_SIG` leading to the same CI as the one chosen for signing
- Neither ES10b.GetRAT nor ES10b.GetProfilesInfo requests SHALL be executed

During the Profile Installation, the following SCP03t TLVs SHALL be used by default:

- `#S_INIT_SC_PROF1`
- `#CONF_ISDP_EMPTY`

- no TLV for "ES8+.ReplaceSessionKeys" function SHALL be used (i.e. the Profile SHALL be downloaded by using the session keys <S_ENC> and <S_MAC>)

2.2.3.2 Default Local Profile Management process

By default, when an Operational Profile needs to be enabled, disabled or deleted on the eUICC (e.g. As mentioned in an initial condition), the following rules apply except if it is differently defined in the Test Case.

If 'LSI Support' is not included in the ATR:

The EnableProfileRequest and the DisableProfileRequest SHALL contain the following parameters:

- ICCID of the Profile to Enable or to Disable
- RefreshFlag set to TRUE

The eUICC SHALL send the REFRESH command in "UICC Reset" mode (i.e. the APDU[TERMINAL_PROFILE] indicating the support "UICC Reset" SHALL be used by the Test Tool).

If 'LSI Support' is included in the ATR:

The test tool SHALL confirm 'LSI Support' in the PPS2 procedure and SHALL send MANAGE_LSI(Configure LSI) with the following tags:

- 'Highest LSI proposed by the terminal' set to 3.
- 'LSI options supported by the terminal' set to 0x01.
- 'MEP mode(s) supported by the S_Device in the order of priority' set to (unless otherwise specified by the Test Sequence):
  '01': MEP-A1,
  '02': MEP-A2,
  '03': MEP-B.
- 'Maximum number of LSIs for Enabled Profiles of the Device' set to 2.

Test tool SHALL process the response to MANAGE_LSI(Configure LSI) as follow.

- Accept the 'LSI options' returned by the eUICC. If it is absent, consider it as 0x00.
- Select the MEP mode, <MEP_MODE>, to be used as the 'Jointly supported MEP mode'.

Test tool SHALL select eSIM Port=0 as the command port.

Test tool SHALL indicate support of LSI COMMAND with "Proactive Session Request" and Proactive UICC: LSI COMMAND with "UICC Platform Reset".

If <MEP_MODE> is MEP-A1:

The EnableProfileRequest and the DisableProfileRequest SHALL contain the following parameters:

- ICCID of the Profile to Enable or to Disable

- RefreshFlag set to TRUE
- targetEsimPort set to 1 or 2

The eUICC SHALL send the LSI COMMAND with the action “Proactive session request” with the targetEsimPort on the eSIM Port 0. The test tool SHALL FETCH the LSI COMMAND on port 0, send the TERMINAL RESPONSE on the same port, and then FETCH the REFRESH command on the targetEsimPort.

The eUICC SHALL send the REFRESH command in "eUICC Profile State Change" mode or the “UICC Reset” mode (i.e. the APDU[TERMINAL_PROFILE] indicating the support "eUICC Profile State Change" or “UICC Reset” SHALL be used by the Test Tool) on the targetEsimPort.

If the REFRESH mode is "eUICC Profile State Change", the Test Tool SHALL send the TERMINAL RESPONSE on the targetEsimPort. If the REFRESH mode is "UICC Reset", Test Tool SHALL send the MANAGE_LSI(Reset LSE) on the targetEsimPort.

If <MEP_MODE> is MEP-A2:

The EnableProfileRequest and the DisableProfileRequest SHALL contain the following parameters:

- ICCID of the Profile to Enable or to Disable
- RefreshFlag set to FALSE

The test tool SHALL not send the targetEsimPort, but SHALL extract and verify the targetEsimPort returned by the eUICC in the EnableProfileResponse.

If <MEP_MODE> is MEP-B:

The EnableProfileRequest and the DisableProfileRequest SHALL contain the following parameters:

- ICCID of the Profile to Enable or to Disable
- RefreshFlag set to TRUE

The test tool SHALL not send the targetEsimPort as it is the same as the command port.

The test tool SHALL FETCH the REFRESH command on the command port.

The eUICC SHALL send the REFRESH command in "eUICC Profile State Change" mode or the “UICC Reset” mode (i.e. the APDU[TERMINAL_PROFILE] indicating the support "eUICC Profile State Change" or “UICC Reset” SHALL be used by the Test Tool) on the command port.

If the REFRESH mode is "eUICC Profile State Change", the Test Tool SHALL send the TERMINAL RESPONSE on the same command port. If the REFRESH mode is "UICC Reset", Test Tool SHALL send the MANAGE_LSI(Reset LSE) on the same command port.

The DeleteProfileRequest SHALL contain the following parameter:

- ICCID of the Profile to Delete

2.2.3.3 ASN.1 elements verifications

Each time the eUICC returns an ASN.1 structure containing a SEQUENCE OF elements, the order of elements SHALL be checked by the Test Tool except for the particular responses:

- `notificationMetadataList` of `ListNotificationResponse`
- `profileInfoListOk` of `ProfileInfoListResponse`
- `notificationList` of `RetrieveNotificationsListResponse`

2.2.3.4 Default Remote Profile Management process

By default, when an Operational Profile needs to be enabled, disabled or deleted on the eUICC by using an RPM script (e.g. as mentioned in an initial condition), the following rules apply except if it is differently defined in the Test Case.

In order to execute the Common Mutual Authentication procedure, the following requests SHALL be sent by the Test Tool:

- `#GET_EUICC_INFO1` and `#GET_EUICC_CHALLENGE`
- `MTD_AUTHENTICATE_SMDP`
    - with the SM-DP+ address `#TEST_DP_ADDRESS1`
    - with #CTX_PARAMS1_RPM
    - If `euiccCiPKIdListForSigningV3` is present in `eUICCInfo1`, select the eSIM CA RootCA Public Key, following the priority order given by the eUICC in `euiccCIPKIdListForSigningV3`. The `paramNewCertVariant` SHALL be set to TRUE.
    - If `euiccCiPKIdListForSigningV3` is not present in `eUICCInfo1`, select the eSIM CA RootCA Public Key, following the priority order given by the eUICC in `euiccCIPKIdListForSigning`. The `paramNewCertVariant` SHALL be set to FALSE.
    - with the `#CERT_S_SM_DPauth_SIG` leading to the same CI as the one chosen for signing
    - with the `#CRL_LIST` included

No ES10b.GetProfilesInfo request SHALL be executed.

In order to execute the Sub-procedure RPM Execution, the `LoadRpmPackageRequest` SHALL contain the following parameters:

- only one `RpmCommand` (corresponding to the requested operation)
- the ICCID of the Profile to Enable, to Disable or to Delete
- the parameters `continueOnFailure` and `rpmPending` SHALL not be set

In case the RPM script aims to Disable or Enable a Profile, the eUICC SHALL send the REFRESH command in "UICC Reset" mode (i.e. the APDU [TERMINAL_PROFILE] indicating the support "UICC Reset" SHALL be used by the Test Tool).

2.2.3.5 Default Common Mutual Authentication procedure

By default, when Common Mutual Authentication procedure has to be executed only as a pre-step (e.g. as mentioned in an initial condition for the test cases where only some particular steps of the Profile Download are verified), the following rules apply except if it is differently defined in the Test Case.

The following requests SHALL be sent by the Test Tool:

- `#GET_EUICC_INFO1` and `#GET_EUICC_CHALLENGE`
- `MTD_AUTHENTICATE_SMDP`
    - with the SM-DP+ address `#TEST_DP_ADDRESS1`
    - with `#CTX_PARAMS1_MATCH_ID`
    - If `euiccCiPKIdListForSigningV3` is present in `eUICCInfo1`, select the eSIM CA RootCA Public Key, following the priority order given by the eUICC in `euiccCIPKIdListForSigningV3`. The `paramNewCertVariant` SHALL be set to TRUE.
    - If `euiccCiPKIdListForSigningV3` is not present in `eUICCInfo1`, select the eSIM CA RootCA Public Key, following the priority order given by the eUICC in `euiccCIPKIdListForSigning`. The `paramNewCertVariant` SHALL be set to FALSE.
    - with the `#CERT_S_SM_DPauth_SIG` leading to the same CI as the one chosen for signing
    - with the `#CRL_LIST` included

2.2.3.6 Usage of Logical Channels

All ES10 commands are wrapped in a STORE DATA APDU, and sent in a Logical Channel dedicated to sending commands to the ISD-R.

All other APDus (in particular, APDUs specified in Annex D.4) are sent in the Basic Logical Channel.

2.2.4 VOID

2.2.5 Pass Criteria

A test execution is considered as successful only if the test procedure was fully carried out successfully.

A test execution is considered as failed if the tested feature provides an unexpected behaviour.

A test execution is considered as inconclusive when the pass criteria cannot be evaluated due to issues during the setup of the initial conditions (including the ICx steps) or during the execution of steps in which no requirement is referenced.

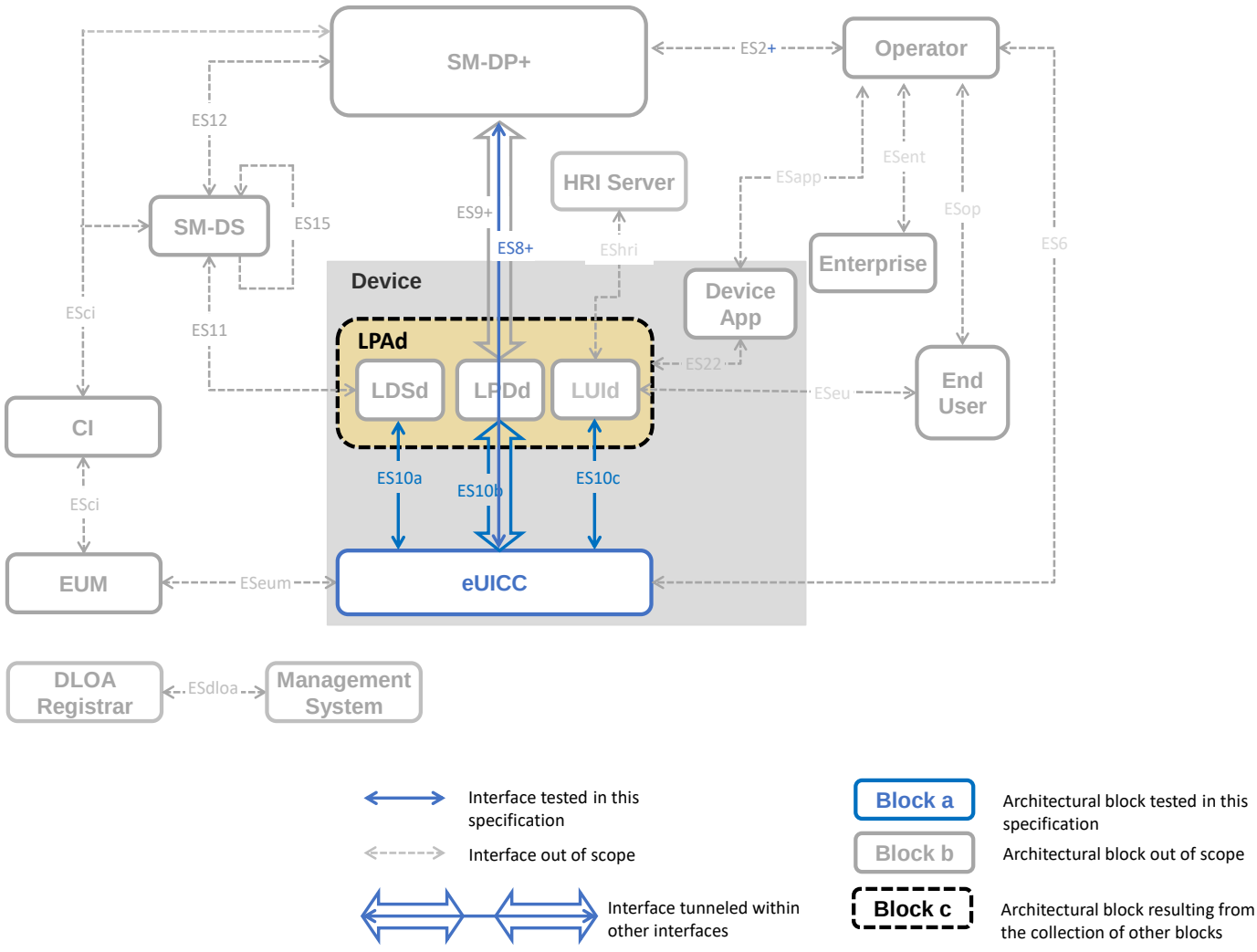
2.2.6 Future Study

Some of the test cases or test sequences described in this Test Plan are FFS (For Future Study). This MAY mean that some clarifications are expected at the requirement level to conclude on a test method. As consequence, the corresponding test SHALL not be executed.

3 Testing Architecture

3.1 Testing Scope

All the interfaces, intended to be tested in the scope of this document, are presented hereafter:



Interface	Between		Description
ES6	Operator	eUICC	Used by the Operator for the management of Operator services via OTA services.
ES8+	SM-DP+	eUICC	Provides a secure end-to-end channel between the SM-DP+ and the eUICC for the administration of the ISD-P and the associated Profile during download and installation. It provides Perfect Forward Secrecy.
ES10a	LDSd	eUICC	Used between the LDSd and the LPA Services to handle a Profile discovery.
ES10b	LPDd	eUICC	Used between the LPDd and the LPA services to transfer a Bound Profile Package to the eUICC. This interface plays no role in the decryption of Profile Packages.
ES10c	LUId	eUICC	Used between the LUId and the LPA services for Local Profile Management by the End User.

Table 7: Tested Interfaces Descriptions

3.2 Testing Execution

This chapter aims to describe the different testing environments and equipments to allow the test cases to be executed.

To permit the execution of the different test cases described in this Test Plan, specifics simulators SHALL be used. The simulators that have been defined are listed hereafter:

- S_Device: the Device Simulator used to send some commands to the eUICC under test using ISO/IEC 7816-4 [7] on the contact interface
- S_SM-DP+: the SM-DP+ Simulator
- S_SM-DS: the SM-DS Simulator
- S_MNO: the MNO Simulator
- S_LPAd: the LPAd Simulator

Implementation of these simulators remains under the responsibility of the test tool providers. The aim of all the test cases is to verify the compliance of an Actor/Component (i.e. eUICC).

Following notations are used:

- `S_ComponentName` for a simulated component
- `ComponentName` for the Implementation Under Test (IUT)

The use of "-- optional" in any ASN.1 elements defined within this document indicate that the test tool SHALL allow for the value either being present with that value, or being absent.

3.2.1 eUICC - Test Environment

The following test environment is used for all eUICC test cases as defined in chapter 4.2 and 5.2 (unless it is specified differently in the specific test case). Following conditions apply:

- Removable eUICC is used

- EUM SHALL provide products with one of the form factors specified in ETSI TS 102 221 [5]
- EUM SHALL provide products compliant with Annex G.2 – eUICC Initial States
- LPAd / MNO / SM-DP+ / SM-DS / Device Simulators SHALL be implemented by the test tools
- The Device Simulator SHALL support both LSI indication options (*MANAGE LSI(Select LSI)* and *T=1 using NAD byte*) allowed for MEP, and use the one indicated by the EUM in IUT_EUICC_MULTIPLEXING_LSI_INDICATION.

The reference of this Test Environment is TE_eUICC.

3.2.2 VOID

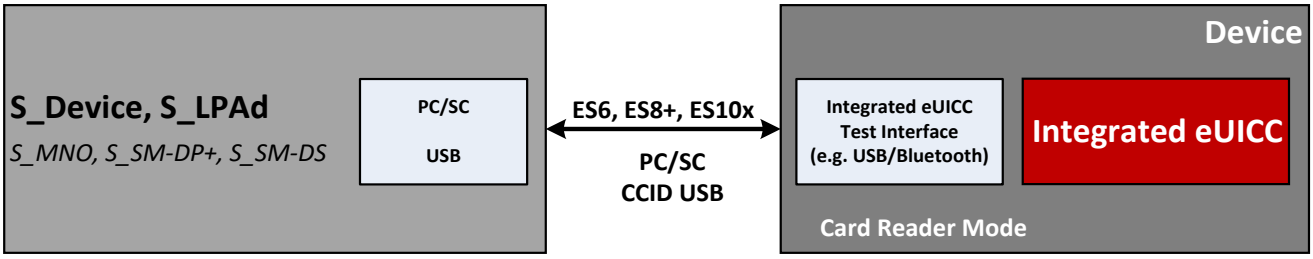
3.2.3 VOID

3.2.4 VOID

3.2.5 Integrated eUICC - Test Environment

The following test environment is used for all eUICC test cases as defined in chapter 4.2 and 5.2 (unless it is specified differently in the specific test case). Following conditions apply:

- EUM SHALL provide products compliant with Annex G.2 – eUICC Initial States
- LPAd / MNO / SM-DP+ / SM-DS / Device Simulators SHALL be implemented by the test tools
    - The Device Simulator SHALL support the same commands, responses, and functionalities as in section 3.2.1, and differs only by the support of a dedicated physical interface.
- Integrated eUICC shall provide a test interface which includes one of the following:
    - ISO/IEC 7816-4 [7]
    - USB CCID [28]
- For Integrated eUICC providing a USB CCID [28] test interface, the provisions of Annex J SHALL apply
- For Integrated eUICC providing ISO/IEC 7816-4 [7], the requirements of 3.2.1 eUICC - Test Environment with implementing shall apply



The reference of this [28] USB CCID based Test Environment is TE_Integrated eUICC.

4 Interface Compliance Testing

4.1 General Overview

This section focuses on the implementation of the different interfaces according to the GSMA RSP Technical Specification [2]. The aim is to verify the compliance of all eUICC interfaces.

4.2 eUICC Interfaces

4.2.1 ATR and ISD-R Selection

4.2.1.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 3.4.1
- Section 5.7
- Annex D

4.2.1.2 Test Cases

4.2.1.2.1 TC_eUICC_ATR_And_ISDR_Selection

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 is loaded on the eUICC and, unless specified otherwise by the test sequence, is in Disabled state.

Test Sequence #01 Nominal: ATR and Select ISD-R

Step	Direction	Sequence / Description	Expected result
1	S_Device → eUICC	RESET	ATR present with the first tBi (i>2) after T = 15 containing b2=1
2	S_Device → eUICC	[SELECT_MF]	FCP Template present SW=0x9000
3	S_Device → eUICC	[TERMINAL_CAPABILITY_LPAd]	SW=0x9000
4	S_Device → eUICC	[TERMINAL_PROFILE]	Toolkit initialization THEN SW=0x9000
5	S_LPAd → eUICC	[MANAGE_CHANNEL_OPEN]	Extract the <CHANNEL_NUMBER> from response data SW=0x9000

6	S_LPAd → eUICC	MTD_SELECT(#ISD_R_AID)	The response data:
			0x6F <L>
			84 <L> #ISD_R_AID
			A5 <L> <PROPRIETARY_DATA> #R_ISDR_SELECTION
			SW=0x9000

Test Sequence #02 Nominal: ATR and Select ISD-R with Enabled Profile

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
1	S_Device → eUICC	RESET	ATR present with the first tBi (i>2) after T = 15 containing b2=1
2	S_Device → eUICC	[SELECT_MF]	FCP Template present SW=0x9000
3	S_Device → eUICC	[TERMINAL_CAPABILITY_LPAd]	SW=0x9000
4	S_Device → eUICC	[TERMINAL_PROFILE]	Toolkit initialization THEN SW=0x9000
5	S_LPAd → eUICC	[MANAGE_CHANNEL_OPEN]	Extract the <CHANNEL_NUMBER> from response data SW=0x9000
6	S_LPAd → eUICC	MTD_SELECT(#ISD_R_AID)	The response data: 0x6F <L> 84 <L> #ISD_R_AID A5 <L> <PROPRIETARY_DATA> #R_ISDR_SELECTION_EN_PROF

			SW=0x9000

Test Sequence #03 Nominal: ATR and Select ISD-R when LPAe supported

Step	Direction	Sequence / Description	Expected result
1	S_Device → eUICC	RESET	ATR present with the first tBi (i>2) after T = 15 containing b2=1
2	S_Device → eUICC	[SELECT_MF]	FCP Template present SW=0x9000
3	S_Device → eUICC	[TERMINAL_CAPABILITY_LPAd]	SW=0x9000
4	S_Device → eUICC	[TERMINAL_PROFILE]	Toolkit initialization THEN SW=0x9000
5	S_LPAd → eUICC	[MANAGE_CHANNEL_OPEN]	Extract the <CHANNEL_NUMBER> from response data SW=0x9000

6	S_LPAd → eUICC	MTD_SELECT(#ISD_R_AID)	The response data:
			0x6F <L>
			84 <L> #ISD_R_AID
			A5 <L> <PROPRIETARY_DATA> #R_ISDR_SELECTION_LPAE
			SW=0x9000

Test Sequence #04 Nominal: ATR and Select ISD-R for MEP-A1

Step	Direction	Sequence / Description	Expected result
1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = '01', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
3	PROC_MEP_LSI_MULTIPLEXING(0)		
4	S_Device → eUICC	[SELECT_MF]	FCP Template present SW=0x9000
5	S_Device → eUICC	[TERMINAL_CAPABILITY_LPAd]	SW=0x9000

6	S_Device → eUICC	[TERMINAL_PROFILE_LSI_COMMAND]	Toolkit initialization THEN SW=0x9000
7	S_LPAd → eUICC	[MANAGE_CHANNEL_OPEN]	Extract the <CHANNEL_NUMBER> from response data SW=0x9000

8	S_LPAd → eUICC	MTD_SELECT(#ISD_R_AID)	The response data:
			0x6F <L>
			84 <L> #ISD_R_AID
			A5 <L>
			<PROPRIETARY_DATA>
			#R_ISDR_SELECTION
			SW=0x9000

Test Sequence #05 Nominal: ATR and Select ISD-R for MEP-A2

Step	Direction	Sequence / Description	Expected result
1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = '02', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
3	PROC_MEP_LSI_MULTIPLEXING(0)		
4	S_Device → eUICC	[SELECT_MF]	FCP Template present SW=0x9000
5	S_Device → eUICC	[TERMINAL_CAPABILITY_LPAd]	SW=0x9000
6	S_Device → eUICC	[TERMINAL_PROFILE]	Toolkit initialization THEN SW=0x9000
7	S_LPAd → eUICC	[MANAGE_CHANNEL_OPEN]	Extract the <CHANNEL_NUMBER> from response data SW=0x9000

8	S_LPAd → eUICC	MTD_SELECT(#ISD_R_AID)	The response data:
			0x6F <L>
			84 <L> #ISD_R_AID
			A5 <L> <PROPRIETARY_DATA>
			#R_ISDR_SELECTION
			SW=0x9000

Test Sequence #06 Nominal: ATR and Select ISD-R for MEP-B

Step	Direction	Sequence / Description	Expected result
1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030102", 2)	Verify <MEP_MODE> = '03', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
3	S_Device → eUICC	[SELECT_MF]	FCP Template present SW=0x9000
4	S_Device → eUICC	[TERMINAL_CAPABILITY_LPAd]	SW=0x9000
5	S_Device → eUICC	[TERMINAL_PROFILE]	Toolkit initialization THEN SW=0x9000
6	S_LPAd → eUICC	[MANAGE_CHANNEL_OPEN]	Extract the <CHANNEL_NUMBER> from response data SW=0x9000

7	S_LPAd → eUICC	MTD_SELECT(#ISD_R_AID)	The response data:
			0x6F <L>
			84 <L> #ISD_R_AID
			A5 <L>
			<PROPRIETARY_DATA>
			#R_ISDR_SELECTION
			SW=0x9000

Test Sequence #07 Nominal: ATR and Select ISD-R with Enabled Profile for MEP-A1

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on Port 1.

Step	Direction	Sequence / Description	Expected result
1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203",	Verify <MEP_MODE> = '01', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS

		2)	
3	PROC_MEP_LSI_MULTIPLEXING(0)		
4	S_Device → eUICC	[SELECT_MF]	FCP Template present SW=0x9000
5	S_Device → eUICC	[TERMINAL_CAPABILITY_LPAd]	SW=0x9000
6	S_Device → eUICC	[TERMINAL_PROFILE_LSI_COMMAND]	Toolkit initialization THEN SW=0x9000
7	S_LPAd → eUICC	[MANAGE_CHANNEL_OPEN]	Extract the <CHANNEL_NUMBER> from response data SW=0x9000

8	S_LPAd → eUICC	MTD_SELECT(#ISD_R_AID)	The response data:
			0x6F <L>
			84 <L> #ISD_R_AID
			A5 <L>
			<PROPRIETARY_DATA>
			#R_ISDR_SELECTION_EN_PROF
			SW=0x9000

Test Sequence #08 Nominal: ATR and Select ISD-R with Enabled Profile for MEP-A2

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = '02', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
3	PROC_MEP_LSI_MULTIPLEXING(0)		
4	S_Device → eUICC	[SELECT_MF]	FCP Template present SW=0x9000
5	S_Device → eUICC	[TERMINAL_CAPABILITY_LPAd]	SW=0x9000

6	S_Device → eUICC	[TERMINAL_PROFILE]	Toolkit initialization THEN SW=0x9000
7	S_LPAd → eUICC	[MANAGE_CHANNEL_OPEN]	Extract the <CHANNEL_NUMBER> from response data SW=0x9000

8	S_LPAd → eUICC	MTD_SELECT(#ISD_R_AID)	The response data:
			0x6F <L>
			84 <L> #ISD_R_AID
			A5 <L> <PROPRIETARY_DATA> #R_ISDR_SELECTION_EN_PROF
			SW=0x9000

Test Sequence #09 Nominal: ATR and Select ISD-R with Enabled Profile for MEP-B

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on Port 1.

Step	Direction	Sequence / Description	Expected result
1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030102", 2)	Verify <MEP_MODE> = '03', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
3	PROC_MEP_LSI_MULTIPLEXING(1)		
4	S_Device → eUICC	[SELECT_MF]	FCP Template present SW=0x9000
5	S_Device → eUICC	[TERMINAL_CAPABILITY_LPAd]	SW=0x9000
6	S_Device → eUICC	[TERMINAL_PROFILE]	Toolkit initialization THEN SW=0x9000
7	S_LPAd → eUICC	[MANAGE_CHANNEL_OPEN]	Extract the <CHANNEL_NUMBER> from response data SW=0x9000
8	S_LPAd → eUICC	MTD_SELECT(#ISD_R_AID)	The response data: 0x6F <L> 84 <L> #ISD_R_AID A5 <L> <PROPRIETARY_DATA> #R_ISDR_SELECTION_EN_PROF

			SW=0x9000

4.2.2 ES6 (Operator -- eUICC): UpdateMetadata

4.2.2.1 Conformance Requirements

References

3GPP TS 23.040 - Technical realization of the Short Message Service (SMS) [22]

GSMA RSP Technical Specification [2]:

- Section 2.4.5
- Section 2.9.1, 2.9.3.2
- Section 5.4
- Section 5.7.15

4.2.2.2 Test Cases

4.2.2.2.1 TC_eUICC_ES6.UpdateMetadata

Throughout all the ES6.UpdateMetadata test cases, SMS is used as the secure OTA channel.

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_WITH_PPRS_AND_ICON is loaded on the eUICC.

Test Sequence #01 Nominal: Unset PPR1

The purpose of this test is to verify that the MNO can unset PPR1 from a Profile and that the eUICC can handle an Update Metadata request with only one field present.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
1	S_Device → eUICC	MTD_SEND_SMS_PP([INSTALL_PERSO_RES_ISDP]; MTD_STORE_DATA_SCRIPT(#REMOVE_PPR1,	SW=0x91XX

		FALSE))	
2	S_Device → eUICC	FETCH "XX"	MTD_CHECK_SMS_POR(0x9000)
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
5	S_LPAd → eUICC	MTD_STORE_DATA(#GET_NEW_METADATA_V3)	#R_GET_UPDATE_N1 SW=0x9000

Test Sequence #02 Nominal: Unset PPR2 and update icon

The purpose of this test is to verify that the MNO can unset PPR2 and update the icon and icon type values from a Profile.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
1	S_Device → eUICC	MTD_SEND_SMS_PP([INSTALL_PERSO_RES_ISDP]; MTD_STORE_DATA_SCRIPT(#UPD_ICON_REM_PPR2, FALSE))	SW=0x91XX
2	S_Device → eUICC	FETCH "XX"	MTD_CHECK_SMS_POR(0x9000)
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
5	S_LPAd → eUICC	MTD_STORE_DATA(#GET_NEW_METADATA_V3)	#R_GET_UPDATE_N2 SW=0x9000

Test Sequence #03 Nominal: Unset PPR1 and PPR2 and update Profile name and Service Provider name

The purpose of this test is to verify that MNO can unset PPR1 and PPR2 from a Profile and can update the Service Provider Name and Profile Name values.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		

1	S_Device → eUICC	MTD_SEND_SMS_PP([INSTALL_PERSO_RES_ISDP]; MTD_STORE_DATA_SCRIPT(#UPD_NAMES_REM_PPRS_V3, FALSE))	SW=0x91XX
2	S_Device → eUICC	FETCH “XX”	MTD_CHECK_SMS_POR(0x9000)
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
5	S_LPAd → eUICC	MTD_STORE_DATA(#GET_NEW_METADATA_V3)	#R_GET_UPDATE_N3 SW=0x9000

Test Sequence #04 Nominal: VOID

Test Sequence #05 Nominal: Delete icon

The purpose of this test is to verify that the MNO can delete the icon and icon type from a Profile.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
1	S_Device → eUICC	MTD_SEND_SMS_PP([INSTALL_PERSO_RES_ISDP]; MTD_STORE_DATA_SCRIPT(#UPD_NAMES_REM_ICON_REM_PPRS_V3, FALSE))	SW=0x91XX
2	S_Device →eUICC	FETCH “XX”	MTD_CHECK_SMS_POR(0x9000)
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
5	S_LPAd → eUICC	MTD_STORE_DATA(#GET_NEW_METADATA_V3)	#R_GET_UPDATE_N7 SW=0x9000

Test Sequence #06 Nominal: Delete Unset PPRs

The purpose of this test is to verify that the MNO can delete already unset PPRs using the Update Metadata request.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	S_Device → eUICC	MTD_SEND_SMS_PP([INSTALL_PERSO_RES_ISDP]; MTD_STORE_DATA_SCRIPT(#REMOVE_ PPRS_V3, FALSE))	SW=0x91XX
IC3	S_Device → eUICC	FETCH “XX”	MTD_CHECK_SMS_POR(0x9000)
IC4	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
1	S_Device → eUICC	MTD_SEND_SMS_PP([INSTALL_PERSO_RES_ISDP]; MTD_STORE_DATA_SCRIPT(#UPD_NAMES_REM_PPRS_V3, FALSE))	SW=0x91XX
2	S_Device → eUICC	FETCH “XX”	MTD_CHECK_SMS_POR(0x9000)
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
5	S_LPAd → eUICC	MTD_STORE_DATA(#GET_NEW_METADATA_V3)	#R_GET_UPDATE_N6 SW=0x9000

Test Sequence #07 Error: Set a pprUpdateControl value to one

The purpose of this test is to verify that the eUICC is correctly handling a pprUpdateControl value error from the MNO request, and return the expected error code status.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
1	S_Device → eUICC	MTD_SEND_SMS_PP([INSTALL_PERSO_RES_ISDP]; MTD_STORE_DATA_SCRIPT(#UPD_PPR_CONTROL, FALSE))	SW=0x91XX
2	S_Device → eUICC	FETCH “XX”	MTD_CHECK_SMS_POR(

			0x6A81)
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
5	S_LPAd → eUICC	MTD_STORE_DATA(#GET_NEW_METADATA_V3)	#R_METADATA_UNCHANGED SW=0x9000

Test Sequence #08 Error: Update Metadata on a Disable Profile

The purpose of this test is to verify that the eUICC is correctly rejecting an Update Metadata request from the MNO when the targeted Profile is Disabled.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
1	S_Device → eUICC	MTD_SEND_SMS_PP([INSTALL_PERSO_RES_ISDP]; MTD_STORE_DATA_SCRIPT(#REMOVE_PPR1, FALSE))	SW=0x91XX or SW=0x9000 (i.e. envelope rejected, see NOTE) or any error SW (i.e. envelope rejected, see NOTE)
2	S_Device → eUICC	FETCH “XX”	SMS POR received SCP80 response status code equal to 0x06 (Unidentified security error) or 0x09 (TAR unknown)
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
5	S_LPAd → eUICC	MTD_STORE_DATA(#GET_NEW_METADATA_V3)	#R_METADATA_UNCHANGED SW=0x9000
NOTE: Depending on the implementation, the eUICC MAY decide to not send back a POR (i.e. SW=0x9000 on the ENVELOPE command). Therefore, the steps 2 and 3 SHALL only be executed in case SW=0x91XX.			

Test Sequence #09 Error: Empty request

The purpose of this test is to verify that the eUICC is correctly rejecting an Update Metadata request from the MNO when no field is present.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
1	S_Device → eUICC	MTD_SEND_SMS_PP([INSTALL_PERSO_RES_ISDP]; MTD_STORE_DATA_SCRIPT(#UPD_NO_METADATA, FALSE))	SW=0x91XX
2	S_Device → eUICC	FETCH “XX”	MTD_CHECK_SMS_POR(<ANY_SW_IN_ERROR>)
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
5	S_LPAd → eUICC	MTD_STORE_DATA(#GET_NEW_METADATA_V3)	#R_METADATA_UNCHANGED SW=0x9000

Test Sequence #10 Error: VOID

Test Sequence #11 Error: VOID

Test Sequence #12 Error: Update Metadata with Enterprise Configuration while the targeted profile is not an Enterprise profile

The purpose of this test is to verify that a non-Enterprise profile cannot accept Update Metadata command UPD_WITH_EC with Enterprise Configuration fields.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
1	S_Device → eUICC	MTD_SEND_SMS_PP([INSTALL_PERSO_RES_ISDP]; MTD_STORE_DATA_SCRIPT(#UPD_WITH_EC, FALSE))	SW=0x91XX
2	S_Device → eUICC	FETCH “XX”	MTD_CHECK_SMS_POR(0x6985)
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
5	S_LPAd → eUICC	MTD_STORE_DATA(#GET_NEW_METADATA_V3)	#R_METADATA_UNCHANGED SW=0x9000

Test Sequence #13 Error: Delete Service Provider Name

The purpose of this test is to verify that the eUICC is correctly handling the deletion of a non deletable Metadata from the MNO request and return the expected error code status.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
1	S_Device → eUICC	MTD_SEND_SMS_PP([INSTALL_PERSO_RES_ISDP]; MTD_STORE_DATA_SCRIPT(#REM_SP_NAME, FALSE))	SW=0x91XX
2	S_Device →eUICC	FETCH “XX”	MTD_CHECK_SMS_POR(0x6A80)
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_NEW_METADATA_V3)
			#R_METADATA_UNCHANGED
			SW=0x9000

4.2.2.2.2 TC_eUICC_ES6.UpdateMetadata_EnterpriseProfiles

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The communication between the S_Device and the eUICC has been initialized by using the [TERMINAL_CAPABILITY_Enterprise] and the S_LPAd has selected the ISD-R.
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_WITH_ER is loaded on the eUICC.

Test Sequence #01 Nominal: Enterprise Rules present in Metadata

The purpose of this test is to update Metadata of the PROFILE_OPERATIONAL1 with new Enterprise rules as defined in UPD_ENT_CONFIG1.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE_Enterprise		
1	S_Device → eUICC	MTD_SEND_SMS_PP([INSTALL_PERSO_RES_ISDP]; MTD_STORE_DATA_SCRIPT(#UPD_ENT_CONFIG1, FALSE))	SW=0x91XX
2	S_Device → eUICC	FETCH “XX”	MTD_CHECK_SMS_POR(0x9000)
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
5	S_LPAd → eUICC	MTD_STORE_DATA(#GET_ENTERPRISE_CONFIG_OP_PROF1)	resp ProfileInfoListResponse ::= profileInfoListOk :{ { iccid #ICCID_OP_PROF1, enterpriseConfiguration #ENTERPRISE_CONFIG3 } } SW=0x9000

Test Sequence #02 Nominal: Switch of Reference Enterprise Rules form one Enterprise profile to another

The purpose of this test is to update Metadata of the PROFILE_OPERATIONAL1 (configured with METADATA_WITH_ER) with UPD_ENT_CONFIG1 (Reference Enterprise Rules bit is set), while PROFILE_OPERATIONAL2 (configured with METADATA_WITH_RER_PROF2) has already the Reference Enterprise Rules bit set using.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL2 with #METADATA_WITH_RER_PROF2 is loaded on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_WITH_ER is Enabled.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE_Enterprise		

1	S_Device → eUICC	MTD_SEND_SMS_PP([INSTALL_PERSO_RES_ISDP]; MTD_STORE_DATA_SCRIPT(#UPD_ENT_CONFIG1, FALSE))	SW=0x91XX
2	S_Device → eUICC	FETCH “XX”	MTD_CHECK_SMS_POR(0x9000)
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
5	S_LPAd → eUICC	MTD_STORE_DATA(#GET_ENTERPRISE_CONFIG_OP_PROF1)	resp ProfileInfoListResponse ::= profileInfoListOk :{ { iccid #ICCID_OP_PROF1, enterpriseConfiguration #ENTERPRISE_CONFIG3 } } SW=0x9000
6	S_LPAd → eUICC	MTD_STORE_DATA(#GET_ENTERPRISE_CONFIG_OP_PROF2)	resp ProfileInfoListResponse ::= profileInfoListOk :{ { iccid #ICCID_OP_PROF2, enterpriseConfiguration #ENTERPRISE_CONFIG4 } } SW=0x9000

Test Sequence #03 Nominal: Update Metadata with Reference Enterprise Rules while a non-Enterprise profile is already installed

The purpose of this test is to update Metadata of the PROFILE_OPERATIONAL1 with Reference Enterprise Rules bit set and Only Enterprise Profiles Installed bit set, while a non-Enterprise profile PROFILE_OPERATIONAL2 is already installed. The command shall be accepted.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL2 is loaded on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_WITH_ER is Enabled.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE_Enterprise		

1	S_Device → eUICC	MTD_SEND_SMS_PP([INSTALL_PERSO_RES_ISDP]; MTD_STORE_DATA_SCRIPT(#UPD_ENT_CONFIG1, FALSE))	SW=0x91XX
2	S_Device → eUICC	FETCH “XX”	MTD_CHECK_SMS_POR(0x9000)
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
5	S_LPAd → eUICC	MTD_STORE_DATA(#GET_ENTERPRISE_CONFIG_OP_PROF1)	resp ProfileInfoListResponse ::= profileInfoListOk :{ { iccid #ICCID_OP_PROF1, enterpriseConfiguration #ENTERPRISE_CONFIG3 } } SW=0x9000

Test Sequence #04 Error: Enterprise Rules present in Metadata

The purpose of this test is to verify that update of Metadata with Enterprise Rules using UPD_ENT_CONFIG1 while the targeted profile with METADATA_WITH_EC does not contain Enterprise rules, will fail.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_WITH_EC is loaded on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE_Enterprise		
1	S_Device → eUICC	MTD_SEND_SMS_PP([INSTALL_PERSO_RES_ISDP]; MTD_STORE_DATA_SCRIPT(#UPD_ENT_CONFIG1, FALSE))	SW=0x91XX
2	S_Device → eUICC	FETCH “XX”	MTD_CHECK_SMS_POR(0x6985)
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		
		#GET_ENTERPRISE_CONFIG_OP_PROF1)
			resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			iccid #ICCID_OP_PROF1,
			enterpriseConfiguration
			#ENTERPRISE_CONFIG5
			}
			}
			SW=0x9000

4.2.2.2.3 TC_eUICC_ES6.UpdateMetadata_Service_Specific_Data

Throughout all the ES6.UpdateMetadata test cases, SMS is used as the secure OTA channel.

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The communication between the S_Device and the eUICC has been initialized by using the [TERMINAL_CAPABILITY_LPA_Alerting] and the S_LPAd has selected the ISD-R.

Test Sequence #01 Nominal: Update Service Specific Data

The purpose of this test is to verify that the MNO can update the Service Specific Data using the Update Metadata request.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_WITH_PPRS_ICON_AND_SPEC_DATA is loaded on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE_LPA_Alerting		
1	S_Device → eUICC	MTD_SEND_SMS_PP([INSTALL_PERSO_RES_ISDP]; MTD_STORE_DATA_SCRIPT(#UPD_SPEC_DATA, FALSE))	SW=0x91XX
2	S_Device →eUICC	FETCH “XX”	MTD_CHECK_SMS_POR(0x9000)

3	S_Device →eUICC	TERMINAL RESPONSE	SW=0x91XX
4	S_Device →eUICC	FETCH “XX”	REFRESH Command (“Application Update”, “BF22”)
5	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
6	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
7	S_LPAd → eUICC	MTD_STORE_DATA(#GET_NEW_METADATA_V3)	#R_GET_UPDATE_N10 SW=0x9000

Test Sequence #02 Nominal: Delete Service Specific Data

The purpose of this test is to verify that the MNO can delete the Service Specific Data Info using the Update Metadata request.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_WITH_PPRS_ICON_AND_SPEC_DATA is loaded on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE_LPA_Alerting		
1	S_Device → eUICC	MTD_SEND_SMS_PP([INSTALL_PERSO_RES_ISDP]; MTD_STORE_DATA_SCRIPT(#REM_SPEC_DATA, FALSE))	SW=0x91XX
2	S_Device →eUICC	FETCH “XX”	MTD_CHECK_SMS_POR(0x9000)
3	S_Device →eUICC	TERMINAL RESPONSE	SW=0x91XX
4	S_Device →eUICC	FETCH “XX”	REFRESH Command (“Application Update”, “BF22”)
5	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
6	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
7	S_LPAd → eUICC	MTD_STORE_DATA(#GET_NEW_METADATA_V3)	#R_GET_UPDATE_N11 SW=0x9000

4.2.2.2.4 TC_eUICC_ES6.UpdateMetadata_V3NotificationConfiguration

Throughout all the ES6.UpdateMetadata test cases, SMS is used as the secure OTA channel.

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The communication between the S_Device and the eUICC has been initialized by using the [TERMINAL_CAPABILITY_LPA_Alerting] and the S_LPAd has selected the ISD-R.

Test Sequence #01 Nominal: Update Notification Configuration Info

The purpose of this test is to verify that the MNO can update the Notification Configuration Info using the Update Metadata request.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_WITH_PPRS_ICON_AND_NOTIF is loaded on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE_LPA_Alerting		
1	S_Device → eUICC	MTD_SEND_SMS_PP([INSTALL_PERSO_RES_ISDP]; MTD_STORE_DATA_SCRIPT(#UPD_NOTIF_CONFIG_INFO, FALSE))	SW=0x91XX
2	S_Device →eUICC	FETCH “XX”	MTD_CHECK_SMS_POR(0x9000)
3	S_Device →eUICC	TERMINAL RESPONSE	SW=0x91XX
4	S_Device →eUICC	FETCH “XX”	REFRESH Command (“Application Update”, “B6”)
5	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
6	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
7	S_LPAd → eUICC	MTD_STORE_DATA(#GET_NEW_METADATA_V3)	#R_GET_UPDATE_N8 SW=0x9000

Test Sequence #02 Nominal: Delete Notification Configuration Info

The purpose of this test is to verify that the MNO can delete elements in the Notification Configuration Info using the Update Metadata request.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_WITH_PPRS_ICON_AND_NOTIF is loaded on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE_LPA_Alerting		
1	S_Device → eUICC	MTD_SEND_SMS_PP([INSTALL_PERSO_RES_ISDP]; MTD_STORE_DATA_SCRIPT(#REM_NOTIF_CONFIG_INFO, FALSE))	SW=0x91XX
2	S_Device →eUICC	FETCH “XX”	MTD_CHECK_SMS_POR(0x9000)
3	S_Device →eUICC	TERMINAL RESPONSE	SW=0x91XX
4	S_Device →eUICC	FETCH “XX”	REFRESH Command (“Application Update”, “B6”)
5	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
6	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
7	S_LPAd → eUICC	MTD_STORE_DATA(#GET_NEW_METADATA_V3)	#R_GET_UPDATE_N9 SW=0x9000

4.2.2.2.5 TC_eUICC_ES6.UpdateMetadata_V3RPM

Throughout all the ES6.UpdateMetadata test cases, SMS is used as the secure OTA channel.

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The communication between the S_Device and the eUICC has been initialized by using the [TERMINAL_CAPABILITY_LPA_Alerting] and the S_LPAd has selected the ISD-R.

Test Sequence #01 Nominal: Update RPM Configuration

The purpose of this test is to verify that the MNO can update the RPM Configuration using the Update Metadata request.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_WITH_PPRS_ICON_AND_RPM_CONFIG is loaded on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE_LPA_Alerting		
1	S_Device → eUICC	MTD_SEND_SMS_PP([INSTALL_PERSO_RES_ISDP]; MTD_STORE_DATA_SCRIPT(#UPD_RPM_CONFIG, FALSE))	SW=0x91XX
2	S_Device →eUICC	FETCH “XX”	MTD_CHECK_SMS_POR(0x9000)
3	S_Device →eUICC	TERMINAL RESPONSE	SW=0x91XX
4	S_Device →eUICC	FETCH “XX”	REFRESH Command (“Application Update”, “BA”)
5	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
6	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
7	S_LPAd → eUICC	MTD_STORE_DATA(#GET_NEW_METADATA_V3)	#R_GET_UPDATE_N12 SW=0x9000

Test Sequence #02 Nominal: Delete RPM Configuration

The purpose of this test is to verify that the MNO can delete elements in the RPM Configuration using the Update Metadata request.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_WITH_PPRS_ICON_AND_RPM_CONFIG is loaded on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE_LPA_Alerting		

1	S_Device → eUICC	MTD_SEND_SMS_PP([INSTALL_PERSO_RES_ISDP]; MTD_STORE_DATA_SCRIPT(#REM_ RPM_CONFIG, FALSE))	SW=0x91XX
2	S_Device →eUICC	FETCH “XX”	MTD_CHECK_SMS_POR(0x9000)
3	S_Device →eUICC	TERMINAL RESPONSE	SW=0x91XX
3	S_Device →eUICC	FETCH “XX”	REFRESH Command (“Application Update”, “BA”)
4	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
5	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
6	S_LPAd → eUICC	MTD_STORE_DATA(#GET_NEW_METADATA_V3)	#R_GET_UPDATE_N13 SW=0x9000

4.2.2.2.6 TC_eUICC_ES6.UpdateMetadata_V3HRIServerAddress

Throughout all the ES6.UpdateMetadata test cases, SMS is used as the secure OTA channel.

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The communication between the S_Device and the eUICC has been initialized by using the [TERMINAL_CAPABILITY_LPA_Alerting] and the S_LPAd has selected the ISD-R.

Test Sequence #01 Nominal: Update HRI Server Address

The purpose of this test is to verify that the MNO can update the HRI Server Address using the Update Metadata request.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_WITH_PPRS_ICON_AND_HRI_ADDR is loaded on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE_LPA_Alerting		
1	S_Device → eUICC	MTD_SEND_SMS_PP([INSTALL_PERSO_RES_ISDP]; MTD_STORE_DATA_SCRIPT(#UPD_HRI_ADDR, FALSE))	SW=0x91XX
2	S_Device →eUICC	FETCH “XX”	MTD_CHECK_SMS_POR(0x9000)
3	S_Device →eUICC	TERMINAL RESPONSE	SW=0x91XX
4	S_Device →eUICC	FETCH “XX”	REFRESH Command (“Application Update”, “9B”)
5	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
6	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
7	S_LPAd → eUICC	MTD_STORE_DATA(#GET_NEW_METADATA_V3)	#R_GET_UPDATE_N14 SW=0x9000

Test Sequence #02 Nominal: Delete HRI Server Address

The purpose of this test is to verify that the MNO can delete the HRI Server Address using the Update Metadata request.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_WITH_PPRS_ICON_AND_HRI_ADDR is loaded on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE_LPA_Alerting		
1	S_Device → eUICC	MTD_SEND_SMS_PP([INSTALL_PERSO_RES_ISDP]; MTD_STORE_DATA_SCRIPT(#REM_HRI_ADDR, FALSE))	SW=0x91XX
2	S_Device →eUICC	FETCH “XX”	MTD_CHECK_SMS_POR(0x9000)
3	S_Device →eUICC	TERMINAL RESPONSE	SW=0x91XX

4	S_Device →eUICC	FETCH “XX”	REFRESH Command (“Application Update”, “9B”)
5	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
6	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
7	S_LPAd → eUICC	MTD_STORE_DATA(#GET_NEW_METADATA_V3)	#R_GET_UPDATE_N11 SW=0x9000

4.2.2.2.7 TC_eUICC_ES6.UpdateMetadata_V3LPRConfiguration

Throughout all the ES6.UpdateMetadata test cases, SMS is used as the secure OTA channel.

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The communication between the S_Device and the eUICC has been initialized by using the [TERMINAL_CAPABILITY_LPA_Alerting] and the S_LPAd has selected the ISD-R.

Test Sequence #01 Nominal: Update LPR Configuration

The purpose of this test is to verify that the MNO can update the LPR Configuration using the Update Metadata request.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_WITH_PPRS_ICON_AND_LPR_CONFIG is loaded on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE_LPA_Alerting		
1	S_Device → eUICC	MTD_SEND_SMS_PP([INSTALL_PERSO_RES_ISDP]; MTD_STORE_DATA_SCRIPT(#UPD_LPR_CONFIG, FALSE))	SW=0x91XX
2	S_Device →eUICC	FETCH “XX”	MTD_CHECK_SMS_POR(0x9000)

3	S_Device →eUICC	TERMINAL RESPONSE	SW=0x91XX
4	S_Device →eUICC	FETCH “XX”	REFRESH Command (“Application Update”, “BC”)
5	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
6	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
7	S_LPAd → eUICC	MTD_STORE_DATA(#GET_NEW_METADATA_V3)	#R_GET_UPDATE_N15 SW=0x9000

Test Sequence #02 Nominal: Delete LPR Configuration

The purpose of this test is to verify that the MNO can delete elements in the LPR Configuration using the Update Metadata request.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_WITH_PPRS_ICON_AND_LPR_CONFIG is loaded on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE_LPA_Alerting		
1	S_Device → eUICC	MTD_SEND_SMS_PP([INSTALL_PERSO_RES_ISDP]; MTD_STORE_DATA_SCRIPT(#REM_LPR_CONFIG, FALSE))	SW=0x91XX
2	S_Device →eUICC	FETCH “XX”	MTD_CHECK_SMS_POR(0x9000)
3	S_Device →eUICC	TERMINAL RESPONSE	SW=0x91XX
4	S_Device →eUICC	FETCH “XX”	REFRESH Command (“Application Update”, “BC”)
5	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
6	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
7	S_LPAd → eUICC	MTD_STORE_DATA(#GET_NEW_METADATA_V3)	#R_GET_UPDATE_N16 SW=0x9000

4.2.2.2.8 TC_eUICC_ES6.UpdateMetadata_V3DeviceChange

Throughout all the ES6.UpdateMetadata test cases, SMS is used as the secure OTA channel.

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The communication between the S_Device and the eUICC has been initialized by using the [TERMINAL_CAPABILITY_LPA_Alerting] and the S_LPAd has selected the ISD-R.

Test Sequence #01 Nominal: Update Device Change Configuration with DP

The purpose of this test is to verify that the MNO can update the Device Change Configuration using the Update Metadata request.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_WITH_PPRS_ICON_AND_DC_CONFIG_DP is loaded on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE_LPA_Alerting		
1	S_Device → eUICC	MTD_SEND_SMS_PP([INSTALL_PERSO_RES_ISDP]; MTD_STORE_DATA_SCRIPT(#UPD_DC_CONFIG_DP, FALSE))	SW=0x91XX
2	S_Device →eUICC	FETCH “XX”	MTD_CHECK_SMS_POR(0x9000)
3	S_Device →eUICC	TERMINAL RESPONSE	SW=0x91XX
4	S_Device →eUICC	FETCH “XX”	REFRESH Command (“Application Update”, “BF20”)
5	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
6	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
7	S_LPAd → eUICC	MTD_STORE_DATA(#GET_NEW_METADATA_V3)	#R_GET_UPDATE_N17 SW=0x9000

Test Sequence #02 Nominal: Delete Device Change Configuration with DP

The purpose of this test is to verify that the MNO can delete elements in the Device Change Configuration using the Update Metadata request.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_WITH_PPRS_ICON_AND_DC_CONFIG_DP is loaded on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE_LPA_Alerting		
1	S_Device → eUICC	MTD_SEND_SMS_PP([INSTALL_PERSO_RES_ISDP]; MTD_STORE_DATA_SCRIPT(#REM_DC_CONFIG_DP, FALSE))	SW=0x91XX
2	S_Device →eUICC	FETCH “XX”	MTD_CHECK_SMS_POR(0x9000)
3	S_Device →eUICC	TERMINAL RESPONSE	SW=0x91XX
4	S_Device →eUICC	FETCH “XX”	REFRESH Command (“Application Update”, “BF20”)
5	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
6	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
7	S_LPAd → eUICC	MTD_STORE_DATA(#GET_NEW_METADATA_V3)	#R_GET_UPDATE_N18 SW=0x9000

Test Sequence #03 Nominal: Update Device Change Configuration with AC

The purpose of this test is to verify that the MNO can update the Device Change Configuration using the Update Metadata request.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_WITH_PPRS_ICON_AND_DC_CONFIG_AC is loaded on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE_LPA_Alerting		

1	S_Device → eUICC	MTD_SEND_SMS_PP([INSTALL_PERSO_RES_ISDP]; MTD_STORE_DATA_SCRIPT(#UPD_DC_CONFIG_AC, FALSE))	SW=0x91XX
2	S_Device →eUICC	FETCH “XX”	MTD_CHECK_SMS_POR(0x9000)
3	S_Device →eUICC	TERMINAL RESPONSE	SW=0x91XX
4	S_Device →eUICC	FETCH “XX”	REFRESH Command (“Application Update”, “BF20”)
5	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
6	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
7	S_LPAd → eUICC	MTD_STORE_DATA(#GET_NEW_METADATA_V3)	#R_GET_UPDATE_N19 SW=0x9000

Test Sequence #04 Nominal: Delete Device Change Configuration with AC

The purpose of this test is to verify that the MNO can delete elements in the Device Change Configuration using the Update Metadata request.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_WITH_PPRS_ICON_AND_DC_CONFIG_AC is loaded on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE_LPA_Alerting		
1	S_Device → eUICC	MTD_SEND_SMS_PP([INSTALL_PERSO_RES_ISDP]; MTD_STORE_DATA_SCRIPT(#REM_DC_CONFIG_AC, FALSE))	SW=0x91XX
2	S_Device →eUICC	FETCH “XX”	MTD_CHECK_SMS_POR(0x9000)
3	S_Device →eUICC	TERMINAL RESPONSE	SW=0x91XX
4	S_Device →eUICC	FETCH “XX”	REFRESH Command (“Application Update”, “BF20”)

5	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
6	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

7	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_NEW_METADATA_V3)
			#R_GET_UPDATE_N20
			SW=0x9000

4.2.3 ES8+ (SM-DP+ -- eUICC): InitialiseSecureChannel

4.2.3.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 2.5.6
- Section 3.1.3.3
- Section 3.5
- Section 5.5.1

4.2.3.2 Test Cases

4.2.3.2.1 TC_eUICC_ES8+.InitialiseSecureChannel

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 is not loaded on the eUICC.
eUICC	The communication between the S_Device and the eUICC has been initialized and the S_LPAd has selected the ISD-R. Common Mutual Authentication procedure has been successfully executed between the eUICC and the S_SM-DP+ Sub-procedure Profile Download and Installation – End User Confirmation has been successfully executed between the eUICC and the S_SM-DP+ • #PREP_DOWNLOAD_NO_CC has been sent to the eUICC

Test Sequence #01 Error: Invalid Remote Operation

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#INIT_SC_INVALID_OP_ID, #CONF_ISDP_PROF1, #METADATA_OP_PROF1,		

	NO_PARAM, #UPP_OP_PROF1)		
IC3	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT> • <BPP_SEG_A0> • <BPP_SEG_A1> • <BPP_SEG_A3>		
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands except the last one SW=0x9000 with the response data #R_PIR_INVALID_OP_ID for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG

Test Sequence #02 Error: Invalid SM-DP+ Signature

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#INIT_SC_INVALID_SIGN, #CONF_ISDP_PROF1, #METADATA_OP_PROF1, NO_PARAM, #UPP_OP_PROF1)		
IC3	Execute the step IC3 of the Test Sequence #01 defined in this section		
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data all STORE DATA commands except for the last one SW=0x9000 with the response data #R_PIR_INVALID_SIGN for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG

Test Sequence #03 Error: Invalid Transaction Identifier

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#INIT_SC_INVALID_TRANS_ID, #CONF_ISDP_PROF1, #METADATA_OP_PROF1, NO_PARAM, #UPP_OP_PROF1)		
IC3	Execute the step IC3 of the Test Sequence #01 defined in this section		

1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands except the last one SW=0x9000 with the response data #R_PIR_INVALID_TRANS_ID for the last STORE DATA command The transactionId returned in the response SHALL not be checked (any value SHALL be accepted) The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG

Test Sequence #04 Error: Invalid CRT Values

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#INIT_SC_INVALID_CRT, #CONF_ISDP_PROF1, #METADATA_OP_PROF1, NO_PARAM, #UPP_OP_PROF1)		
IC3	Execute the step IC3 of the Test Sequence #01 defined in this section		
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for the intermediate STORE DATA commands (if any) SW=0x9000 with the response data #R_PIR_INVALID_CRT for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG

Test Sequence #05 Error: InitialiseSecureChannel request while Secure Channel Session is ongoing

The purpose of this test is to ensure that the eUICC rejects an InitialiseSecureChannel request if a secure channel session is already ongoing.

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1,		
IC3	Execute the step IC3 of the Test Sequence #01 defined in this section		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands

IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x6A88 or 0x6985 or SW=0x9000 with a ProfileInstallationResult containing an ErrorResult

4.2.4 ES8+ (SM-DP+ -- eUICC): ConfigureISDP

4.2.4.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 2.4.4
- Section 2.5.6
- Section 3.1.3.3
- Section 3.5
- Section 5.5.2

4.2.4.2 Test Cases

4.2.4.2.1 TC_eUICC_ES8+.ConfigureISDP

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 is not loaded on the eUICC.
eUICC	The communication between the S_Device and the eUICC has been initialized and the S_LPAd has selected the ISD-R. Common Mutual Authentication procedure has been successfully executed between the eUICC and the S_SM-DP+ Sub-procedure Profile Download and Installation – End User Confirmation has been successfully executed between the eUICC and the S_SM-DP+ • #PREP_DOWNLOAD_NO_CC has been sent to the eUICC

Test Sequence #01 Nominal: Empty Proprietary Data

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_EMPTY, #METADATA_OP_PROF1, NO_PARAM, #UPP_OP_PROF1)		
IC3	Split the <BPP> into several segments arrays named:		

	• <BPP_SEG_INIT> • <BPP_SEG_A0> • <BPP_SEG_A1> • <BPP_SEG_A3>		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
2	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands
3	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x9000 without response data for all STORE DATA commands except the last one SW=0x9000 with the response data #R_PIR_OK for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG.

4	S_LPAd → eUICC	MTD_STORE_DATA(#GET_CONF_OP_PROF1)	resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			isdpAid <ISD_P_AID>
			-- dpProprietaryData SHALL not be
			-- present
			}
			}
			SW=0x9000

Test Sequence #02 Nominal: Proprietary Data with the maximum length authorized (i.e. 128 bytes)

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_MAX_LENGTH, #METADATA_OP_PROF1, NO_PARAM, #UPP_OP_PROF1)		
IC3	Execute the step IC3 of the Test Sequence #01 defined in this section		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
2	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands

3	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x9000 without response data for all STORE DATA commands except the last one SW=0x9000 with the response data #R_PIR_OK for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG.

4	S_LPAd → eUICC	MTD_STORE_DATA(#GET_CONF_OP_PROF1)	#R_CONF_OP_PROF1
			SW=0x9000

Test Sequence #03 Error: Proprietary Data with the maximum length exceeded (i.e. 129 bytes)

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_SIZE_EXCEEDED, #METADATA_OP_PROF1, NO_PARAM, #UPP_OP_PROF1)		
IC3	Execute the step IC3 of the Test Sequence #01 defined in this section		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands except the last one SW=0x9000 with the response data #R_PIR_INVALID_DATA for the last STORE DATA command

4.2.5 ES8+ (SM-DP+ -- eUICC): StoreMetadata

4.2.5.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 2.4.5.1
- Section 2.5.6
- Section 2.9.1
- Section 3.1.3.3
- Section 3.2.5
- Section 5.5.3

4.2.5.2 Test Cases

4.2.5.2.1 TC_eUICC_ES8+.StoreMetadata

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The communication between the S_Device and the eUICC has been initialized and the S_LPAd has selected the ISD-R. Common Mutual Authentication procedure has been successfully executed between the eUICC and the S_SM-DP+ Sub-procedure Profile Download and Installation – End User Confirmation has been successfully executed between the eUICC and the S_SM-DP+ • #PREP_DOWNLOAD_NO_CC has been sent to the eUICC

Test Sequence #01 Nominal: All Metadata fields present (PNG icon used and PPR1 set)

The purpose of this test is to download the PROFILE_OPERATIONAL1 by setting all Metadata fields. In this sequence, a PNG icon is used and PPR1 is set.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	No Operational Profile is present on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #FULL_METADATA, NO_PARAM, #UPP_OP_PROF1)		
IC3	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT> • <BPP_SEG_A0> • <BPP_SEG_A1> • <BPP_SEG_A3>		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands
2	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x9000 without response data for all STORE DATA commands expect the last one

			SW=0x9000 with the response data #R_PIR_OK for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG

3	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_METADATA_OP_PROF1)
			#R_GET_METADATA_OP_PROF1
			SW=0x9000

Test Sequence #02 Nominal: With JPG icon

The purpose of this case is to verify the ability to download JPG icon. The icon size does not allow for the command to fit into one data sequence.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is not loaded on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_WITH_JPG, NO_PARAM, #UPP_OP_PROF1)		
IC3	Execute the step IC3 of the Test Sequence #01 defined in this section		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands
2	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x9000 without response data for all STORE DATA commands except the last one SW=0x9000 with the response data #R_PIR_OK for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG

3	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			…
			iccid #ICCID_OP_PROF1,

			iconType jpg, icon #ICON_JPG, … } } SW=0x9000

Test Sequence #03 Nominal: Without providing Profile Class

The purpose of this test is to download the PROFILE_OPERATIONAL1 by not indicating the Profile Class in the Metadata. In such a case, the default Profile Class 'Operational' SHALL be set by the eUICC.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is not loaded on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_NO_CLASS, NO_PARAM, #UPP_OP_PROF1)		
IC3	Execute the step IC3 of the Test Sequence #01 defined in this section		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands
2	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data #R_PIR_OK for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG

3	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			…
			iccid #ICCID_OP_PROF1,

			profileClass operational … } } SW=0x9000

Test Sequence #04 Nominal: With PPR2 set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is not loaded on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_WITH_PPR2, NO_PARAM, #UPP_OP_PROF1)		
IC3	Execute the step IC3 of the Test Sequence #01 defined in this section		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands
2	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data #R_PIR_OK for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG

3	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PPR_OP_PROF1)
			resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			iccid #ICCID_OP_PROF1,
			profilePolicyRules {ppr2}
			}
			}
			SW=0x9000

Test Sequence #05 Nominal: With PPR1 and PPR2 set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	No Operational Profile is present on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_WITH_PPR1_PPR2, NO_PARAM, #UPP_OP_PROF1)		
IC3	Execute the step IC3 of the Test Sequence #01 defined in this section		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands
2	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data #R_PIR_OK for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG

3	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PPR_OP_PROF1)
			resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			iccid #ICCID_OP_PROF1,
			profilePolicyRules {ppr1,ppr2}
			}
			}
			SW=0x9000

Test Sequence #06 Nominal: With several Notification events configured

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is not loaded on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_WITH_NOTIFS, NO_PARAM, #UPP_OP_PROF1)		
IC3	Execute the step IC3 of the Test Sequence #01 defined in this section		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands
2	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data #R_PIR_OK for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG

3	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_NOTIF_CONF_OP_PROF1)
			#R_GET_PROF_NOTIF_CONF
			SW=0x9000

Test Sequence #07 Error: ICCID already present in the eUICC

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	General Initial Conditions do not apply.
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 is Disabled.

Step	Direction	Sequence / Description	Expected result
IC1	The communication between the S_Device and the eUICC has been initialized and the S_LPAd has selected the ISD-R. Common Mutual Authentication procedure has been successfully executed between the eUICC and the S_SM-DP+ Sub-procedure Profile Download and Installation – End User Confirmation has been successfully executed between the eUICC and the S_SM-DP+ #PREP_DOWNLOAD_NO_CC has been sent to the eUICC		
IC2	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC3	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1,		

	#CONF_ISDP_PROF1, #METADATA_OP_PROF1, NO_PARAM, #UPP_OP_PROF1)		
IC4	Execute the step IC3 of the Test Sequence #01 defined in this section		
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC6	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data #R_PIR_ICCID_ALREADY_EXIST for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG

Test Sequence #08 Error: Profile Policy Rule is set but Profile Owner is not

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is not loaded on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_PPR_NO_OWNER, NO_PARAM, #UPP_OP_PROF1)		
IC3	Execute the step IC3 of the Test Sequence #01 defined in this section		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data #R_PIR_METADATA_INVALID (See NOTE) for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG

NOTE: The errorReason "pprNotAllowed" or "installFailedDueToUnknownError" MAY be also returned by the eUICC.

Test Sequence #09 Error: Profile Owner is set with a wildcard ('E') digits

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is not loaded on the eUICC.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_WILDCARD, NO_PARAM, #UPP_OP_PROF1)		
IC3	Execute the step IC3 of the Test Sequence #01 defined in this section		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data #R_PIR_METADATA_INVALID (See Note) for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG
NOTE: The errorReason "pprNotAllowed" MAY be also returned by the eUICC.			

Test Sequence #10 Error: Icon Type is set but icon is not

The purpose of this test is to verify ASN.1 conditional requirement for icon presence. If icon type is present then icon SHALL also be present.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is not loaded on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_WITHOUT_ICON, NO_PARAM, #UPP_OP_PROF1)		
IC3	Execute the step IC3 of the Test Sequence #01 defined in this section		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data #R_PIR_METADATA_INVALID for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG

Test Sequence #11 Error: Store Metadata with Enterprise Configuration while the targeted eUICC is non Enterprise Capable

The purpose of this test is check that a non-Enterprise capable eUICC cannot accept profile with Enterprise Configuration.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_WITH_EC is not loaded on the eUICC.
S_LPA	The S_LPA does not perform the Cancel Session procedure.

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_WITH_EC, NO_PARAM, #UPP_OP_PROF1)		
IC3	Execute the step IC3 of the Test Sequence #01 defined in this section		

IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data #R_PIR_UNKNOWN_TLV for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG

4.2.5.2.2 TC_eUICC_ES8+.StoreMetadata_Service_Specific_Data

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The communication between the S_Device and the eUICC has been initialized and the S_LPAd has selected the ISD-R. Common Mutual Authentication procedure has been successfully executed between the eUICC and the S_SM-DP+ Sub-procedure Profile Download and Installation – End User Confirmation has been successfully executed between the eUICC and the S_SM-DP+ #PREP_DOWNLOAD_NO_CC has been sent to the eUICC

Test Sequence #01 Nominal: Metadata include service-specific data, stored

The purpose of this test is to download the PROFILE_OPERATIONAL1 with service-specific metadata stored in the eUICC.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	No Operational Profile is present on the eUICC.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_SERVICE_SPECIFIC_STORED, NO_PARAM, #UPP_OP_PROF1)		
IC3	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT>		

	• <BPP_SEG_A0> • <BPP_SEG_A1> • <BPP_SEG_A3>		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands
2	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x9000 without response data for all STORE DATA commands expect the last one SW=0x9000 with the response data #R_PIR_OK for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_ECDSA

3	S_LPAd → eUICC	MTD_STORE_DATA(#GET_METADATA_OP_PROF1_SERVICE_SPECIFIC)	#R_GET_METADATA_OP_PROF1_SERVICE_SPECIFIC
			SW=0x9000

Test Sequence #02 Nominal: Metadata include service-specific data, not stored

The purpose of this test is to download the PROFILE_OPERATIONAL1 with service-specific metadata *not* stored in the eUICC.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	No Operational Profile is present on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_SERVICE_SPECIFIC_NOT_STORED, NO_PARAM, #UPP_OP_PROF1)		
IC3	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT> • <BPP_SEG_A0>		

	• <BPP_SEG_A1> • • <BPP_SEG_A3>		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands
2	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x9000 without response data for all STORE DATA commands expect the last one SW=0x9000 with the response data #R_PIR_OK for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_ECDSA

3	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_METADATA_OP_PROF1_SERVICE_SPECIFIC)
			#R_GET_METADATA_OP_PROF1_NO_SERVICE_SPECIFIC
			SW=0x9000

Test Sequence #03 Nominal: Metadata include service-specific data, stored and not stored

The purpose of this test is to download the PROFILE_OPERATIONAL1 with service-specific metadata stored in the eUICC and other service-specific metadata *not* stored.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	No Operational Profile is present on the eUICC.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_SERVICE_SPECIFIC_STORED_AND_NOT_STORED, NO_PARAM, #UPP_OP_PROF1)		
IC3	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT>		

	• <BPP_SEG_A0> • <BPP_SEG_A1> • <BPP_SEG_A3>		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands
2	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x9000 without response data for all STORE DATA commands expect the last one SW=0x9000 with the response data #R_PIR_OK for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_ECDSA

3	S_LPAd → eUICC	MTD_STORE_DATA(#GET_METADATA_OP_PROF1_SERVICE_SPECIFIC)	#R_GET_METADATA_OP_PROF1_SERVICE_SPECIFIC
			SW=0x9000

4.2.5.2.3 TC_eUICC_ES8+.StoreMetadata_EnterpriseProfiles

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The communication between the S_Device and the eUICC has been initialized by using the [TERMINAL_CAPABILITY_Enterprise] and the S_LPAd has selected the ISD-R. Common Mutual Authentication procedure has been successfully executed between the eUICC and the S_SM-DP+ with enterpriseCapableDevice present in `DeviceInfo` (in `CtxParams1`) Sub-procedure Profile Download and Installation – End User Confirmation has been successfully executed between the eUICC and the S_SM-DP+ • #PREP_DOWNLOAD_NO_CC has been sent to the eUICC

Test Sequence #01 Nominal: Enterprise Rules present in Metadata

The purpose of this test is to download the PROFILE_OPERATIONAL1 with Enterprise rules in Metadata #METADATA_WITH_ER

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	No Operational Profile is present on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_WITH_ER, NO_PARAM, #UPP_OP_PROF1)		
IC3	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT> • <BPP_SEG_A0> • <BPP_SEG_A1> • <BPP_SEG_A3>		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands
2	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x9000 without response data for all STORE DATA commands expect the last one SW=0x9000 with the response data #R_PIR_OK for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG

3	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_ENTERPRISE_CONFIG_OP_PROF1)
			resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			iccid #ICCID_OP_PROF1,
			enterpriseConfiguration
			
			#ENTERPRISE_CONFIG1_ER
			
			}
			}
			SW=0x9000

Test Sequence #02 Nominal: Enterprise Profile installation while another profile with no PPR1 set is installed

The purpose of this test is to download the PROFILE_OPERATIONAL2 with Enterprise configuration in Metadata #METADATA_WITH_EC_PROF2, while a non-enterprise profile PROFILE_OPERATIONAL1 without PPR1 set is installed.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_OP_PROF1 is installed on the eUICC

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF2, #CONF_ISDP_PROF2, #METADATA_WITH_EC_PROF2, NO_PARAM, #UPP_OP_PROF2)		
IC3	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT> • <BPP_SEG_A0> • <BPP_SEG_A1> • <BPP_SEG_A3>		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands
2	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x9000 without response data for all STORE DATA commands expect the last one SW=0x9000 with the response data #R_PIR_OK for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG

3	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_ENTERPRISE_CONFIG_OP_PROF2)
			resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			iccid #ICCID_OP_PROF2,
			enterpriseConfiguration
			
			#ENTERPRISE_CONFIG1_EC
			
			}
			}
			SW=0x9000

Test Sequence #03 Error: Enterprise Rules with Reference Enterprise Rule set

The purpose of this test is to verify that it is not possible to load the PROFILE_OPERATIONAL1 with Reference Enterprise Rule set in Metadata #METADATA_WITH_RER.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	No Operational Profile is present on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_WITH_RER, NO_PARAM, #UPP_OP_PROF1)		
IC3	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT> • <BPP_SEG_A0> • <BPP_SEG_A1> • <BPP_SEG_A3>		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data #R_PIR_RER_NOTALLOWED for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG

Test Sequence #04 Error: Enterprise Profile installation while another profile with PPR1 set is installed

The purpose is to verify that it is not possible to verify that the installation of PROFILE_OPERATIONAL2 with Enterprise configuration in Metadata #METADATA_WITH_EC_PROF2, while a non-enterprise profile PROFILE_OPERATIONAL1 with PPR1 set is installed.

Initial Conditions

Entity	Description of the initial condition
eUICC	The PROFILE_OPERATIONAL1 with #FULL_METADATA is installed on the eUICC

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF2, #CONF_ISDP_PROF2, #METADATA_WITH_EC_PROF2, NO_PARAM, #UPP_OP_PROF2)		
IC3	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT> • <BPP_SEG_A0> • <BPP_SEG_A1> • <BPP_SEG_A3>		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data #R_PIR_EP_NOTALLOWED for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG

Test Sequence #05 Error: non-Enterprise Profile installation while an Enterprise profile with Enterprise rules that does not allow it is installed

The purpose is to verify that it is not possible to verify that the installation of non-enterprise profile PROFILE_OPERATIONAL2 with Metadata #METADATA_OP_PROF2, while an enterprise profile PROFILE_OPERATIONAL1 is installed with Enterprise rules that does not allow it.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_WITH_RER is installed on the eUICC

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		

IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF2, #CONF_ISDP_PROF2, #METADATA_OP_PROF2, NO_PARAM, #UPP_OP_PROF2)		
IC3	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT> • <BPP_SEG_A0> • <BPP_SEG_A1> • <BPP_SEG_A3>		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data #R_PIR_EP_ONLY for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG

Test Sequence #06 Nominal: Store Metadata with Enterprise Configuration while the targeted device is a Non Enterprise Capable device

The purpose of this test is to verify that non-Enterprise capable devices can accept profile with Enterprise Configuration without Enterprise Rules and with Metadata #METADATA_WITH_EC.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is not loaded on the eUICC.
eUICC	The communication between the S_Device and the eUICC has been initialized and the S_LPAd has selected the ISD-R. Common Mutual Authentication procedure has been successfully executed between the eUICC and the S_SM-DP+ with Sub-procedure Profile Download and Installation – End User Confirmation has been successfully executed between the eUICC and the S_SM-DP+ #PREP_DOWNLOAD_NO_CC has been sent to the eUICC

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1,		

	#CONF_ISDP_PROF1, #METADATA_WITH_EC, NO_PARAM, #UPP_OP_PROF1)		
IC3	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT> • <BPP_SEG_A0> • <BPP_SEG_A1> <BPP_SEG_A3>		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands
2	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x9000 without response data for all STORE DATA commands expect the last one SW=0x9000 with the response data #R_PIR_OK for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG

3	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_ENTERPRISE_CONFIG_OP_PROF1)
			resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			iccid #ICCID_OP_PROF1,
			enterpriseConfiguration
			
			#ENTERPRISE_CONFIG1_EC
			
			}
			}
			SW=0x9000

Test Sequence #07 Error: Store Metadata with Enterprise Rules while the targeted device is a Non Enterprise Capable device

The purpose of this test is to verify that only Enterprise capable devices can accept profile with Enterprise Rules, and with Metadata #METADATA_WITH_ER.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is not loaded on the eUICC.
eUICC	The communication between the S_Device and the eUICC has been initialized and the S_LPAd has selected the ISD-R.

	Common Mutual Authentication procedure has been successfully executed between the eUICC and the S_SM-DP+ with Sub-procedure Profile Download and Installation – End User Confirmation has been successfully executed between the eUICC and the S_SM-DP+ #PREP_DOWNLOAD_NO_CC has been sent to the eUICC

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_WITH_ER, NO_PARAM, #UPP_OP_PROF1)		
IC3	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT> • <BPP_SEG_A0> • <BPP_SEG_A1> <BPP_SEG_A3>		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data #R_PIR_ER_NOTALLOWED for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG

Test Sequence #08 Error: Store Metadata with different Enterprise OID

The purpose of this test is to verify that the installation of an Enterprise profile, with an Enterprise OID in METADATA_WITH_EC_OID2 different from the one defined for an already installed profile (METADATA_WITH_EC), will fail.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_WITH_EC is installed on the eUICC

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF2,		

	#CONF_ISDP_PROF2, #METADATA_WITH_EC_OID2, NO_PARAM, #UPP_OP_PROF2)		
IC3	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT> • <BPP_SEG_A0> • <BPP_SEG_A1> • <BPP_SEG_A3>		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data #R_PIR_EOID_MISMATCH for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG

4.2.6 ES8+ (SM-DP+ -- eUICC): ReplaceSessionKeys

4.2.6.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 2.5.6
- Section 2.6.4, 2.6.5
- Section 3.1.3.3
- Section 5.5.4

4.2.6.2 Test Cases

4.2.6.2.1 TC_eUICC_ES8+.ReplaceSessionKeys

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 is not loaded on the eUICC
eUICC	The communication between the S_Device and the eUICC has been initialized and the S_LPAd has selected the ISD-R. Common Mutual Authentication procedure has been successfully executed between the eUICC and the S_SM-DP+ Sub-procedure Profile Download and Installation – End User Confirmation has been successfully executed between the eUICC and the S_SM-DP+ • #PREP_DOWNLOAD_NO_CC has been sent to the eUICC

Test Sequence #01 Error: Incorrect PPK size

The purpose of this test is to verify that the eUICC checks that PPK sizes are the same as session keys.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is not loaded on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_OP_PROF1, #REPLACE_S_KEYS_REQ_INV_SIZE, #UPP_OP_PROF1) MTD_GENERATE_BPP overriding: For this test sequence, the initial session keys SHALL be used for UPP SCP03t protection. Therefore: Encrypt all <UPP_SEG> with <S_ENC> Calculate and add a MAC to all tags 0x86 of sequenceOf86 by using <S_MAC>		
IC3	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT> • <BPP_SEG_A0> • <BPP_SEG_A1> • <BPP_SEG_A2> • <BPP_SEG_A3>		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands

IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
IC6	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A2>)	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data #R_PIR_PPK_INV for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG

4.2.7 ES8+ (SM-DP+ -- eUICC): LoadProfileElements

4.2.7.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 2.5.6
- Section 3.1.3.3
- Section 5.5.5

4.2.7.2 Test Cases

4.2.7.2.1 TC_eUICC_ES8+.LoadProfileElements

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 is not loaded on the eUICC
eUICC	The communication between the S_Device and the eUICC has been initialized and the S_LPAd has selected the ISD-R. Common Mutual Authentication procedure has been successfully executed between the eUICC and the S_SM-DP+ Sub-procedure Profile Download and Installation – End User Confirmation has been successfully executed between the eUICC and the S_SM-DP+ • #PREP_DOWNLOAD_NO_CC has been sent to the eUICC

Test Sequence #01 Error: EF_{ICCID} different from the ICCID provided in the Profile Metadata

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL2 is not loaded on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_ICCID_MISMATCH, NO_PARAM, #UPP_OP_PROF1)		
IC3	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT> • <BPP_SEG_A0> • <BPP_SEG_A1> • <BPP_SEG_A3>		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
IC6	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x9000 with the response data #R_PIR_DATA_MISMATCH for one of the STORE DATA commands The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG
2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(#ICCID_OP_PROF1, NO_PARAM))	resp ProfileInfoListResponse ::= profileInfoListOk :{} SW=0x9000
3	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(#ICCID_OP_PROF2, NO_PARAM))	resp ProfileInfoListResponse ::= profileInfoListOk :{} SW=0x9000

Test Sequence #02 Error: MCC / MNC of EF_{IMSI} different from MCC / MNC of Profile Owner present in Metadata

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_MCCMNC_MISMATCH, NO_PARAM, #UPP_OP_PROF1)		
IC3	Execute the step IC3 of the Test Sequence #01 defined in this section		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands

IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
IC6	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x9000 with the response data #R_PIR_DATA_MISMATCH for one of the STORE DATA commands The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG
2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(#ICCID_OP_PROF1, NO_PARAM))	resp ProfileInfoListResponse ::= profileInfoListOk :{} SW=0x9000

Test Sequence #03 Error: Session MAC chaining used instead of new Initial MAC chaining

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP (#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_OP_PROF1, #REPLACE_S_KEYS_REQ, #UPP_OP_PROF1) MTD_GENERATE_BPP overriding: For this test sequence, <S_MAC_CHAIN> SHALL be used instead of <PPK_INIT_MAC> for UPP SCP03t protection.		
IC3	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT> • <BPP_SEG_A0> • <BPP_SEG_A1> • <BPP_SEG_A2> • <BPP_SEG_A3>		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
IC6	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands
IC7	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A2>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x9000 with the response data #R_PIR_SECU_INVALID for one of the STORE DATA commands The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(#ICCID_OP_PROF1, NO_PARAM))	resp ProfileInfoListResponse ::= profileInfoListOk :{} SW=0x9000

Test Sequence #04 Error: S-MAC used instead of PPK-MAC

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP (#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_OP_PROF1, #REPLACE_S_KEYS_REQ, #UPP_OP_PROF1) MTD_GENERATE_BPP overriding: For this test sequence <S_MAC> SHALL be used instead of <PPK_MAC> for UPP SCP03t protection.		
IC3	Execute the step IC3 of the Test Sequence #03 defined in this section		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
IC6	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands
IC7	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A2>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x9000 with the response data #R_PIR_SECU_INVALID for one of the STORE DATA commands The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG
2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(#ICCID_OP_PROF1, NO_PARAM))	resp ProfileInfoListResponse ::= profileInfoListOk :{} SW=0x9000

Test Sequence #05 Error: S-ENC used instead of PPK-ENC

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP (#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_OP_PROF1, #REPLACE_S_KEYS_REQ,		

	#UPP_OP_PROF1) MTD_GENERATE_BPP overriding: For this test sequence <S_ENC> SHALL be used instead of <PPK_ENC> for UPP SCP03t protection.		
IC3	Execute the step IC3 of the Test Sequence #03 defined in this section		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
IC6	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands
IC7	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A2>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x9000 with the response data #R_PIR_SECU_INVALID for one of the STORE DATA commands The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG
2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(#ICCID_OP_PROF1, NO_PARAM))	resp ProfileInfoListResponse ::= profileInfoListOk :{} SW=0x9000

Test Sequence #06 Error: Profile Downloading stopped by a Reset

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	No pending Notification is present on the eUICC.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP (#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_OP_PROF1, NO_PARAM, #UPP_OP_PROF1)		
IC3	Execute the step IC3 of the Test Sequence #01 defined in this section		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands

IC6	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x9000 without response data for all STORE DATA commands except the last one. Step 2 SHALL be triggered before sending the last STORE DATA
2	PROC_EUICC_INITIALIZATION_SEQUENCE		
3	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
4	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(#ICCID_OP_PROF1, NO_PARAM))	resp ProfileInfoListResponse ::= profileInfoListOk :{} SW=0x9000

Test Sequence #07 Nominal: ICCID in the 'ProfileHeader' PE is ignored by the eUICC

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_OP_PROF1 NO_PARAM, #UPP_OP_PROF1) #UPP_OP_PROF1 overriding: For this sequence, the *iccid* field SHALL be set to #ICCID_OP_PROF2 in the *ProfileHeader* element		
IC3	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT> • <BPP_SEG_A0> • <BPP_SEG_A1> • <BPP_SEG_A3>		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
IC6	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data #R_PIR_OK for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG. <ISD_P_AID> SHALL be in the range as defined SGP.02 [1].

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			…
			iccid #ICCID_OP_PROF1,
			isdpAid <ISD_P_AID>,
			profileState disabled,
			…
			}
			}
			SW=0x9000

Test Sequence #08 Nominal: With gid1 and gid2 set

The purpose of this test is to verify that an Operational Profile configured with gid1 and gid2 can be downloaded on the eUICC.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL9 is not loaded on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_OP_PROF9, NO_PARAM, #UPP_OP_PROF9)		
IC3	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT> • <BPP_SEG_A0> • <BPP_SEG_A1> • <BPP_SEG_A3>		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
IC6	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data #R_PIR_OK_PROF9 for the last STORE DATA command

			The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG
2	S_LPAd → eUICC	MTD_STORE_DATA(#GET_PROFILES_OWNERS)	resp ProfileInfoListResponse ::= profileInfoListOk :{ { profileOwner { mccMnc #MCC_MNC9, gid1 #GID1, gid2 #GID2 } } } SW=0x9000

Test Sequence #09 Error: gid1 and gid2 provided in the Profile Metadata but not in the Profile Package

The purpose of this test is to verify that if gid1 and gid2 are provided in the Profile Metadata but ef-gid1 and ef-gid2 are not present and the related services (17 and 18) in ef-ust are not available, the eUICC returns an error during the LoadProfileElements process.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is not loaded on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_OP1_GID1GID2_PRESENT, NO_PARAM, #UPP_OP_PROF1)		
IC3	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT> • <BPP_SEG_A0> • <BPP_SEG_A1> • <BPP_SEG_A3>		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
IC6	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands

1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x9000 with the response data #R_PIR_DATA_MISMATCH for one of the STORE DATA commands The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG
2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(#ICCID_OP_PROF1, NO_PARAM))	resp ProfileInfoListResponse ::= profileInfoListOk :{} SW=0x9000

Test Sequence #10 Error: gid1 and gid2 not provided in the Profile Metadata but Service is available in ef-ust

The purpose of this test is to verify that if gid1 and gid2 are not provided in the Profile Metadata but ef-gid1 and ef-gid2 are present and the related services (17 and 18) in ef-ust are available, the eUICC returns an error during the LoadProfileElements process.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL9 is not loaded on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_OP9_GID1GID2_MISSING, NO_PARAM, #UPP_OP_PROF9)		
IC3	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT> • <BPP_SEG_A0> • <BPP_SEG_A1> • <BPP_SEG_A3>		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
IC6	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x9000 with the response data #R_PIR_DATA_MISMATCH for one of the STORE DATA commands The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG
2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(	resp ProfileInfoListResponse ::= profileInfoListOk :{}

		#ICCID_OP_PROF9, NO_PARAM))	SW=0x9000

Test Sequence #11 Nominal: Profile download is success if profileMetaData does not contains profileOwner and EFIMSI file is not present

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL10 is not loaded on the eUICC.
eUICC	The communication between the S_Device and the eUICC has been initialized and the S_LPAd has selected the ISD-R. Common Mutual Authentication procedure has been successfully executed between the eUICC and the S_SM-DP+ •#GET_EUICC_INFO1, #GET_EUICC_CHALLENGE and #AUTHENTICATE_SMDP_WITH_DEVICE_INFO_NAI have been sent to the eUICC •the same GSMA CI has been chosen for signing and for verification Sub-procedure Profile Download and Installation – End User Confirmation has been successfully executed between the eUICC and the S_SM-DP+ #PREP_DOWNLOAD_NO_CC has been sent to the eUICC

Step	**Direction**	**Sequence / Description**	**Expected result**	**REQ**
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>			
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_OP_PROF10_NO_PROFILE_OWNER NO_PARAM, #UPP_OP_PROF10)			
IC3	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT> • <BPP_SEG_A0> • <BPP_SEG_A1> • <BPP_SEG_A3>			
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands	
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands	
IC6	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands	

1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data #R_PIR_OK_PROF10 for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_ECDSA. <ISD_P_AID> SHALL be in the range as defined SGP.02 [1].	RQ25_023 RQ25_024 RQ55_045 RQ55_048 RQ25_025 RQ25_026 RQ55_044

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF10,
		NO_PARAM))
			resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			…
			iccid #ICCID_OP_PROF10,
			isdpAid <ISD_P_AID>,
			profileState disabled,
			…
			}
			}
			SW=0x9000
				RQ32_071
				RQ55_048

Test Sequence #12 Error: Profile download is failed if ProfileMetaData contains profileOwner but EFIMSI file is not present

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL10 is not loaded on the eUICC.
eUICC	The communication between the S_Device and the eUICC has been initialized and the S_LPAd has selected the ISD-R. Common Mutual Authentication procedure has been successfully executed between the eUICC and the S_SM-DP+ •#GET_EUICC_INFO1, #GET_EUICC_CHALLENGE and #AUTHENTICATE_SMDP_WITH_DEVICE_INFO_NAI have been sent to the eUICC •the same GSMA CI has been chosen for signing and for verification Sub-procedure Profile Download and Installation – End User Confirmation has been successfully executed between the eUICC and the S_SM-DP+ #PREP_DOWNLOAD_NO_CC has been sent to the eUICC

Step	Direction	Sequence / Description	Expected result	REQ
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>			
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1,			

	#CONF_ISDP_PROF1, #METADATA_OP_PROF10, NO_PARAM, #UPP_OP_PROF 10)			
IC3	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT> • <BPP_SEG_A0> • <BPP_SEG_A1> • <BPP_SEG_A3>			
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands	
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands	
IC6	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands	
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x9000 with the response data #R_PIR_DATA_MISMATCH for one of the STORE DATA commands The euiccSignPIR SHALL be verified with the #PK_EUICC_ECDSA	RQ25_023 RQ25_024 RQ55_043 RQ55_047 RQ55_048 RQ25_025 RQ25_026 RQ31_173
2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(#ICCID_OP_PROF10, NO_PARAM))	resp ProfileInfoListResponse ::= profileInfoListOk :{} SW=0x9000	RQ32_071 RQ55_043 RQ55_048

4.2.8 ES10a (LPA -- eUICC): GetEuiccConfiguredData

4.2.8.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 3.1.3
- Section 3.3.4
- Section 5.7.3

4.2.8.2 Test Cases

4.2.8.2.1 TC_eUICC_ES10a.GetEuiccConfiguredAddresses

Test Sequence #01 Nominal: Only Root SM-DS Address

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	Only the Root SM-DS address has been set on the ISD-R (see Annex G.2.1).

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(#GET_EUICC_CONFIGURED_DATA)	#R_ES10a_GECA_DS SW = 0x9000

Test Sequence #02 Nominal: Root SM-DS and Default SM-DP+ Addresses

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The ISD-R is provisioned with the Default SM-DP+ Address #TEST_DP_ADDRESS1 and the Root SM-DS address (see Annex G.2.1).

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(#GET_EUICC_CONFIGURED_DATA)	#R_ES10a_GECA_DS_DP_1 SW = 0x9000

4.2.9 ES10a (LPA -- eUICC): SetDefaultDpAddress

4.2.9.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 3.3.4
- Section 5.7.4

4.2.9.2 Test Cases

4.2.9.2.1 TC_eUICC_ES10a.SetDefaultDpAddress

Test Sequence #01 Nominal: Set SM-DP+ Address with Address Empty in eUICC

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	No value is assigned to the Default SM-DP+ Address field.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(#SET_EUICC_CONFIGURED_ADDRESS_1)	#R_ES10a_SD_DP_A_OK SW = 0x9000
2	S_LPAd → eUICC	MTD_STORE_DATA(#GET_EUICC_CONFIGURED_DATA)	#R_ES10a_GECA_DS_DP_1 SW = 0x9000

Test Sequence #02 Nominal: Set SM-DP+ Address with SM-DP+ Address already in eUICC

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The SM-DP+ address #TEST_DP_ADDRESS1 is provisioned.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(#SET_EUICC_CONFIGURED_ADDRESS_2)	#R_ES10a_SD_DP_A_OK SW = 0x9000
2	S_LPAd → eUICC	MTD_STORE_DATA(#GET_EUICC_CONFIGURED_DATA)	#R_ES10a_GECA_DS_DP_2 SW = 0x9000

Test Sequence #03 Nominal: Set Empty SM-DP+ Address with SM-DP+ Address already in eUICC

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The SM-DP+ address #TEST_DP_ADDRESS1 is provisioned.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(#SET_EUICC_CONFIGURED_ADDRESS_EMPTY)	#R_ES10a_SD_DP_A_OK SW = 0x9000
2	S_LPAd → eUICC	MTD_STORE_DATA(#GET_EUICC_CONFIGURED_DATA)	#R_ES10a_GECA_DS SW = 0x9000

Test Sequence #04 Nominal: Set Empty SM-DP+ Address with Empty SM-DP+ Address in eUICC

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	No value is assigned to the Default SM-DP+ Address field.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(#SET_EUICC_CONFIGURED_ADDRESS_EMPTY)	#R_ES10a_SD_DP_A_OK SW = 0x9000
2	S_LPAd → eUICC	MTD_STORE_DATA(#GET_EUICC_CONFIGURED_DATA)	#R_ES10a_GECA_DS SW = 0x9000

4.2.10 ES10b (LPA -- eUICC): PrepareDownload

4.2.10.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 2.6.2, 2.6.7.1, 2.6.7.2
- Section 3.0.1, 3.1.3.2
- Section 4.5.2.1, 4.5.2.2
- Section 5.7.5

4.2.10.2 Test Cases

4.2.10.2.1 TC_eUICC_ES10b.PrepareDownloadNIST

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 is not loaded on the eUICC.

eUICC	The communication between the S_Device and the eUICC has been initialized and the S_LPAd has selected the ISD-R. Common Mutual Authentication procedure has been successfully executed between the eUICC and the S_SM-DP+ • the same GSMA CI based on NIST P-256 curve has been chosen for signing and for verification

Test Sequence #01 Nominal: Without Confirmation Code

Step	Direction	Sequence / Description	Expected result

1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(#PREP_DOWNLOAD_NO_CC)	#R_PREP_DOWNLOAD_NO_CC
			SW=0x9000
			The <EUICC_SIGNATURE2> SHALL be verified with the #PK_EUICC_SIG.
			Verify that the <S_TRANSACTION_ID> present in the euiccSigned2 is the same as in #PREP_DOWNLOAD_NO_CC.

Test Sequence #02 Nominal: With Confirmation Code

Step	Direction	Sequence / Description	Expected result
IC1	<S_HASHED_CC> = MTD_GENERATE_HASHED_CC(#CONFIRMATION_CODE1, <S_TRANSACTION_ID>)		

1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(#PREP_DOWNLOAD_WITH_CC)	#R_PREP_DOWNLOAD_WITH_CC
			SW=0x9000
			The <EUICC_SIGNATURE2> SHALL be verified with the #PK_EUICC_SIG.
			Verify that the <S_TRANSACTION_ID> present in the euiccSigned2 is the same as in #PREP_DOWNLOAD_WITH_CC.
			Verify that the <S_HASHED_CC> present in the euiccSigned2 is the same as in #PREP_DOWNLOAD_WITH_CC.

Test Sequence #03 Nominal: With an unknown otPK.EUICC.ECKA

The purpose of this test is to verify that the eUICC does not use the one-time key pair given by the SM-DP+ when its value does not correspond to a stored one-time key pair. In this case, the eUICC SHALL generate a new set of key.

Step	Direction	Sequence / Description	Expected result
IC1	<S_HASHED_CC> = MTD_GENERATE_HASHED_CC(#CONFIRMATION_CODE1, <S_TRANSACTION_ID>)		
IC2	S_SM-DP+ generates a random <OTPK_EUICC_ECKA>		

1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(#PREP_DOWNLOAD_RETRY_CC)	#R_PREP_DOWNLOAD_WITH_CC
			SW=0x9000

			The <EUICC_SIGNATURE2> SHALL be verified with the #PK_EUICC_SIG. Verify that the <S_TRANSACTION_ID> present in the euiccSigned2 is the same as in #PREP_DOWNLOAD_RETRY_CC. Verify that the <S_HASHED_CC> present in the euiccSigned2 is the same as in #PREP_DOWNLOAD_RETRY_CC. Verify that the <OTPK_EUICC_ECKA> present in the euiccSigned2 is not the same as in #PREP_DOWNLOAD_RETRY_CC.

4.2.10.2.2 TC_eUICC_ES10b.PrepareDownloadBRP

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 is not loaded on the eUICC.
eUICC	The communication between the S_Device and the eUICC has been initialized and the S_LPAd has selected the ISD-R. Common Mutual Authentication procedure has been successfully executed between the eUICC and the S_SM-DP+ • the same GSMA CI based on BrainpoolP256r1 curve has been chosen for signing and for verification

Test Sequence #01 Nominal: Without Confirmation Code

This test sequence SHALL be the same as the Test Sequence #01 defined in section 4.2.10.2.1 – TC_eUICC_ES10b.PrepareDownloadNIST except that all keys and certificates SHALL be based on BrainpoolP256r1.

Test Sequence #02 Nominal: With Confirmation Code

This test sequence SHALL be the same as the Test Sequence #02 defined in section 4.2.10.2.1 – TC_eUICC_ES10b.PrepareDownloadNIST except that all keys and certificates SHALL be based on BrainpoolP256r1.

Test Sequence #03 Nominal: With an unknown otPK.EUICC.ECKA

This test sequence SHALL be the same as the Test Sequence #03 defined in section 4.2.10.2.1 – TC_eUICC_ES10b.PrepareDownloadNIST except that all keys and certificates SHALL be based on BrainpoolP256r1.

4.2.10.2.3 TC_eUICC_ES10b.PrepareDownloadFRP

This test case is defined as FFS and not applicable for this version of test specification.

4.2.10.2.4 TC_eUICC_ES10b.PrepareDownloadErrorCases

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 is not loaded on the eUICC.
eUICC	The communication between the S_Device and the eUICC has been initialized and the S_LPAd has selected the ISD-R. Common Mutual Authentication procedure has been successfully executed between the eUICC and the S_SM-DP+

Test Sequence #01 Error: VOID

Test Sequence #02 Error: With incorrect CERT.DPpb.SIG (i.e. invalid signature)

Step	Direction	Sequence / Description	Expected result

1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(#PREP_DOWNLOAD_INV_CERT)	#R_PREP_DOWNLOAD_INV_CERT
			SW=0x9000
			Verify that the <S_TRANSACTION_ID> present in the response is the same as in #PREP_DOWNLOAD_INV_CERT.

Test Sequence #03 Error: CERT.DPpb.SIG and CERT.DPauth.SIG not belonging to the same entity

Step	Direction	Sequence / Description	Expected result

1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(#PREP_DOWNLOAD_CERT_SMDP2)	#R_PREP_DOWNLOAD_INV_CERT
			SW=0x9000
			Verify that the <S_TRANSACTION_ID> present in the response is the same as in #PREP_DOWNLOAD_CERT_SMDP2.

Test Sequence #04 Error: With invalid SM-DP+ signature

Step	Direction	Sequence / Description	Expected result

1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(#PREP_DOWNLOAD_INV_SIGN)	#R_PREP_DOWNLOAD_INV_SIGN
			SW=0x9000
			Verify that the <S_TRANSACTION_ID> present in the response is the same as in #PREP_DOWNLOAD_INV_SIGN.

Test Sequence #05 Error: With invalid Transaction ID

Step	Direction	Sequence / Description	Expected result

1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(#PREP_DOWNLOAD_INV_TRANS_ID)	#R_PREP_DOWN_INV_TRANS_ID
			SW=0x9000
			The transactionId returned in the response SHALL not be checked (any value SHALL be accepted)

Test Sequence #06 Error: SM-DP+ has not been previously authenticated

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	No Common Mutual Authentication procedure has been executed between the eUICC and the S_SM-DP+ (this condition overrides the last general initial condition defined in this test case)

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

IC3	S_LPAd → eUICC	MTD_STORE_DATA(#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000
			Extract the highest priority CI from <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION> and choose #CERT_S_SM_DPpb_SIG according to this CI curve.

1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(#PREP_DOWNLOAD_NO_AUTH)	#R_PREP_DOWN_NO_SESSION
			SW=0x9000
			The transactionId returned in the response SHALL not be checked (any value SHALL be accepted)

Test Sequence #07 Error: Unsupported curve

Step	Direction	Sequence / Description	Expected result

1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(#PREP_DOWN_INV_CURVE)	#R_PREP_DOWN_INV_CERT
			SW=0x9000
			Verify that the <S_TRANSACTION_ID> present in the response is the same as in #PREP_DOWN_INV_CERT.

Test Sequence #08 Error: Invalid Certificate Role OID

The purpose of this sequence is to make sure that the eUICC refuses any SM-DP+ Certificate for Profile Package Binding that does not indicate “id-rspRole-dp-pb” in its extension for Certificate Policies.

Step	Direction	Sequence / Description	Expected result

1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(#PREP_DOWNLOAD_INV_OID)	#R_PREP_DOWNLOAD_INV_CERT
			SW=0x9000
			Verify that the <S_TRANSACTION_ID> present in the response is the same as in #PREP_DOWNLOAD_INV_OID.

4.2.11 ES10b (LPA -- eUICC): LoadBoundProfilePackage

4.2.11.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 2.4.3 2.4.5.1
- Section 2.5.3, 2.5.6
- Section 2.6.4, 2.6.5, 2.6.7.1, 2.6.7.2, 2.6.7.3
- Section 3.1.3.2, 3.1.3.3, 3.1.5
- Section 3.2.5
- Section 3.5
- Section 5.5, 5.5.1
- Section 5.7.6
- Annex D
- Annex G

4.2.11.2 Test Cases

4.2.11.2.1 TC_eUICC_ES10b.LoadBoundProfilePackageNIST

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 is not loaded on the eUICC.
eUICC	The communication between the S_Device and the eUICC has been initialized and the S_LPAd has selected the ISD-R. Common Mutual Authentication procedure has been successfully executed between the eUICC and the S_SM-DP+ • the same GSMA CI based on NIST P-256 curve has been chosen for signing and for verification Sub-procedure Profile Download and Installation – End User Confirmation has been successfully executed between the eUICC and the S_SM-DP+ • #PREP_DOWNLOAD_NO_CC has been sent to the eUICC

Test Sequence #01 Nominal: By using S-ENC and S-MAC

The purpose of this test is to download the PROFILE_OPERATIONAL1 by using only the session S-ENC and S-MAC keys resulting from key agreement.

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_OP_PROF1, NO_PARAM,		

	#UPP_OP_PROF1)		
IC3	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT> • <BPP_SEG_A0> • <BPP_SEG_A1> • <BPP_SEG_A3> NOTE: In this test sequence, the data resulting of this operation SHALL be composed of the following TLV arrays: • <BPP_SEG_INIT> contains the tag and length fields of the BoundProfilePackage TLV plus the #S_INIT_SC_PROF1 command • <BPP_SEG_A0> contains the tag and length fields of the firstSequenceOf87 TLV plus the first 0x87 TLV containing #CONF_ISDP_PROF1 command • <BPP_SEG_A1> contains the tag and length fields of the sequenceOf88 TLV and each of the '88' TLVs containing #METADATA_OP_PROF1 command • <BPP_SEG_A3> contains the tag and length fields of the sequenceOf86 TLV and each of the '86' TLVs containing #UPP_OP_PROF1 protected with <S_ENC> and <S_MAC>		
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
2	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
3	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data #R_PIR_OK for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG. <ISD_P_AID> SHALL be in the range as defined SGP.02 [1].

5	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(#ICCID_OP_PROF1, NO_PARAM))	resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			…
			iccid #ICCID_OP_PROF1,
			isdpAid <ISD_P_AID>,
			profileState disabled,
			…
			}
			}
			SW=0x9000

Test Sequence #02 Nominal: By using PPK-ENC and PPK-MAC

The purpose of this test is to download the PROFILE_OPERATIONAL1 by using a new set of random session keys: PPK-ENC, PPK-MAC and Initial MAC chaining value.

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_OP_PROF1, #REPLACE_S_KEYS_REQ, #UPP_OP_PROF1)		
IC3	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT> • <BPP_SEG_A0> • <BPP_SEG_A1> • <BPP_SEG_A2> • <BPP_SEG_A3> NOTE: In this test sequence, the data resulting of this operation SHALL be composed of the following TLV arrays: • <BPP_SEG_INIT> contains the tag and length fields of the BoundProfilePackage TLV plus the #S_INIT_SC_PROF1 command • <BPP_SEG_A0> contains the tag and length fields of the firstSequenceOf87 TLV plus the first 0x87 TLV containing #CONF_ISDP_PROF1 command • <BPP_SEG_A1> contains the tag and length fields of the sequenceOf88 TLV and each of the '88' TLVs containing #METADATA_OP_PROF1 command • <BPP_SEG_A2> contains the tag and length fields of the secondSequenceOf87 TLV plus the first '87' TLV, containing the #REPLACE_S_KEYS_REQ command • <BPP_SEG_A3> contains the tag and length fields of the sequenceOf86 TLV and each of the '86' TLVs containing #UPP_OP_PROF1 protected with PPK-ENC and PPK-MAC		
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
2	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
3	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A2>)	SW=0x9000 without response data for all STORE DATA commands
5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data #R_PIR_OK for the last STORE DATA command The euiccSignPIR SHALL be verified with the #PK_EUICC_SIG. <ISD_P_AID> SHALL be in the range as defined SGP.02 [1].
6	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(#ICCID_OP_PROF1, NO_PARAM))	resp ProfileInfoListResponse ::= profileInfoListOk :{

			{
			…
			iccid #ICCID_OP_PROF1,
			isdpAid <ISD_P_AID>,
			profileState disabled,
			…
			}
			}
			SW=0x9000

4.2.11.2.2 TC_eUICC_ES10b.LoadBoundProfilePackageBRP

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 is not loaded on the eUICC.
eUICC	The communication between the S_Device and the eUICC has been initialized and the S_LPAd has selected the ISD-R. Common Mutual Authentication procedure has been successfully executed between the eUICC and the S_SM-DP+ • the same GSMA CI based on BrainpoolP256r1 curve has been chosen for signing and for verification Sub-procedure Profile Download and Installation – End User Confirmation has been successfully executed between the eUICC and the S_SM-DP+ • #PREP_DOWNLOAD_NO_CC has been sent to the eUICC

Test Sequence #01 Nominal: By using S-ENC and S-MAC

This test sequence SHALL be the same as the Test Sequence #01 defined in section 4.2.11.2.1 – TC_eUICC_ES10b.LoadBoundProfilePackageNIST except that all keys and certificates SHALL be based on BrainpoolP256r1.

Test Sequence #02 Nominal: By using PPK-ENC and PPK-MAC

This test sequence SHALL be the same as the Test Sequence #02 defined in section 4.2.11.2.1 – TC_eUICC_ES10b. LoadBoundProfilePackageNIST except that all keys and certificates SHALL be based on BrainpoolP256r1.

4.2.11.2.3 VOID

4.2.11.2.4 TC_eUICC_ES10b.LoadBoundProfilePackage_ErrorCases

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 is not loaded on the eUICC.
eUICC	The communication between the S_Device and the eUICC has been initialized and the S_LPAd has selected the ISD-R.

	Common Mutual Authentication procedure has been successfully executed between the eUICC and the S_SM-DP+ Sub-procedure Profile Download and Installation – End User Confirmation has been successfully executed between the eUICC and the S_SM-DP+ • #PREP_DOWNLOAD_NO_CC has been sent to the eUICC

Test Sequence #01 Error: Unrecognized leading tag in BPP

The purpose of this test is to ensure that the eUICC rejects any BPP segment with an unrecognized leading tag during Profile download. In such case, the eUICC SHALL return a SW of 0x6A88 and SHALL not discard the download session state.

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_OP_PROF1, NO_PARAM, #UPP_OP_PROF1)		
IC3	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT> • <BPP_SEG_A0> • <BPP_SEG_A1> • <BPP_SEG_A3>		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(#UNKNOWN_BPP_SEGMENT)	SW=0x6A88
2	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands
3	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data #R_PIR_OK for the last STORE DATA command

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			…
			iccid #ICCID_OP_PROF1,
			isdpAid <ISD_P_AID>,
			profileState disabled,
			…

			} } SW=0x9000

Test Sequence #02 Error: GetEUICCChallenge during BPP loading

The purpose of this test is to ensure that the eUICC accepts an ES10b.GetEUICCChallenge request indicating the start of a new RSP session while a BPP is loaded.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	No Notification is stored in the eUICC's Pending Notifications List.

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_OP_PROF1, NO_PARAM, #UPP_OP_PROF1)		
IC3	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT> • <BPP_SEG_A0> • <BPP_SEG_A1> • <BPP_SEG_A3>		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
IC5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
IC6	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 without response data for all STORE DATA commands

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_EUICC_CHALLENGE)
			#R_CHALLENGE
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A3>)	SW=0x6A88 or 0x6985
3	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(#ICCID_OP_PROF1, NO_PARAM))	resp ProfileInfoListResponse ::= profileInfoListOk :{} SW=0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(
		#LIST_NOTIF_ALL)
			#R_LIST_NOTIF_NONE
			SW = 0x9000

4.2.12 ES10b (LPA -- eUICC): GetEUICCChallenge

4.2.12.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 3.0.1
- Section 5.7.7

4.2.12.2 Test Cases

4.2.12.2.1 TC_eUICC_ES10b.GetEUICCChallenge

Test Sequence #01 Nominal

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			<EUICC_CHALLENGE> received in this step is different to the <EUICC_CHALLENGE> in Step 1

4.2.13 ES10b (LPA -- eUICC): GetEUICCInfo

4.2.13.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 4.3
- Section 5.7.8

4.2.13.2 Test Cases

4.2.13.2.1 TC_eUICC_ES10b.GetEUICCInfo1

Test Sequence #01 Nominal

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

Test Sequence #02 Nominal: GetEUICCInfo call after GetEUICCChallenge

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

Test Sequence #03 Nominal: GetEUICCInfo1 call after AuthenticateServer using Server Variant O certificates

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000
			Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> and <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION> from response data and verify if they contain at least one same GSMA CI Key ID

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE>		

	• <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> • Choose the #CERT_S_SM_DPauth_SIG leading to the same Root CI certificate		

4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(
		MTD_AUTHENTICATE_SMDP(
		#TEST_DP_ADDRESS1,
		<S_SMDP_CHALLENGE>,
		#CTX_PARAMS1,
		<S_SMDP_SIGNATURE1>,
		#CERT_S_SM_DPauth_SIG,
		NO_PARAM,
		#CRL_LIST,
		FALSE
		)
		)
			MTD_CHECK_AUTH_SERVER_RESP(
			#TEST_DP_ADDRESS1,
			<S_SMDP_CHALLENGE>,
			#CTX_PARAMS1)
			SW = 0x9000

5	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

Test Sequence #04 Nominal:GetEUICCInfo1 call after AuthenticateServer using V3 Server certificates – Variant A

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000
			Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> and <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION> from response data and verify if they contain at least one same GSMA CI Key ID

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: <S_TRANSACTION_ID> <EUICC_CHALLENGE> <S_SMDP_CHALLENGE> <S_SMDP_SIGNATURE1> Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> Choose the #CERT_S_SM_DPauth_SIG and the remaining part of the Variant A Certificates chain leading to the same Root CI certificate as the one chosen for signing		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1,	MTD_CHECK_AUTH_SERVER_RESP(#TEST_DP_ADDRESS1,

		<S_SMDP_CHALLENGE>,
		#CTX_PARAMS1,
		<S_SMDP_SIGNATURE1>,
		#CERT_S_SM_DPauth_SIG,
		#CERT_S_SM_DP_SubCA_SIG,
		#CRL_LIST,
		TRUE
		)
		)
			<S_SMDP_CHALLENGE>,
			#CTX_PARAMS1)
			SW = 0x9000

5	S_LPAd → eUICC	MTD_STORE_DATA
		(#GET_EUICC_INFO1)
			#R_EUICC_INFO1
			SW = 0x9000

Test Sequence #05 Nominal: GetEUICCInfo1 call after AuthenticateServer using V3 Server certificates – Variant B

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA
		(#GET_EUICC_INFO1)
			#R_EUICC_INFO1
			SW = 0x9000
			Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> and <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION> from response data and verify if they contain at least one same GSMA CI Key ID

2	S_LPAd → eUICC	MTD_STORE_DATA
		(#GET_EUICC_CHALLENGE)
			#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: <S_TRANSACTION_ID> <EUICC_CHALLENGE> <S_SMDP_CHALLENGE> <S_SMDP_SIGNATURE1> Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> Choose the #CERT_S_SM_DPauth_SIG and the remaining part of the Variant B Certificates chain leading to the same Root CI certificate as the one chosen for signing		

4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(
		MTD_AUTHENTICATE_SMDP(
		#TEST_DP_ADDRESS1,
		<S_SMDP_CHALLENGE>,
		#CTX_PARAMS1,
		<S_SMDP_SIGNATURE1>,
		#CERT_S_SM_DPauth_SIG,
		#CERT_CI_SubCA_SIG,
		#CRL_LIST,
		TRUE
			MTD_CHECK_AUTH_SERVER_RESP(
			#TEST_DP_ADDRESS1,
			<S_SMDP_CHALLENGE>,
			#CTX_PARAMS1)
			SW = 0x9000

		))	

5	S_LPAd → eUICC	MTD_STORE_DATA
		(#GET_EUICC_INFO1)
			#R_EUICC_INFO1
			SW = 0x9000

Test Sequence #06 Nominal: GetEUICCInfo1 call after AuthenticateServer using V3 Server certificates – Variant C

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA
		(#GET_EUICC_INFO1)
			#R_EUICC_INFO1
			SW = 0x9000
			Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> and <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION> from response data and verify if they contain at least one same GSMA CI Key ID

2	S_LPAd → eUICC	MTD_STORE_DATA
		(#GET_EUICC_CHALLENGE)
			#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: <S_TRANSACTION_ID> <EUICC_CHALLENGE> <S_SMDP_CHALLENGE> <S_SMDP_SIGNATURE1> Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> Choose the #CERT_S_SM_DPauth_SIG and the remaining part of the Variant C Certificates chain leading to the same Root CI certificate as the one chosen for signing		

4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(
		MTD_AUTHENTICATE_SMDP(
		#TEST_DP_ADDRESS1,
		<S_SMDP_CHALLENGE>,
		#CTX_PARAMS1,
		<S_SMDP_SIGNATURE1>,
		#CERT_S_SM_DPauth_SIG,
		
		#CERT_S_SM_DP_SubCAList_SIG,
		#CRL_LIST,
		TRUE
		)
		)
			MTD_CHECK_AUTH_SERVER_RESP(
			#TEST_DP_ADDRESS1,
			<S_SMDP_CHALLENGE>,
			#CTX_PARAMS1)
			SW = 0x9000

5	S_LPAd → eUICC	MTD_STORE_DATA
		(#GET_EUICC_INFO1)
			#R_EUICC_INFO1
			SW = 0x9000

4.2.13.2.2 VOID

4.2.13.2.3 VOID

4.2.13.2.4 TC_eUICC_ES10b.GetEUICCInfo2

Test Sequence #01 Nominal: GetEUICCInfo2 call after AuthenticateServer using Server Variant O

Step	Direction	Sequence / Description	
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	
IC4	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	
IC5	The following inputs are required for Step IC6 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> • Choose the #CERT_S_SM_DPauth_SIG leading to the same Root CI certificate		

IC6	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(
		MTD_AUTHENTICATE_SMDP(
		#TEST_DP_ADDRESS1,
		<S_SMDP_CHALLENGE>,
		#CTX_PARAMS1,
		<S_SMDP_SIGNATURE1>,
		#CERT_S_SM_DPauth_SIG,
		NO_PARAM,
		#CRL_LIST,
		FALSE
		)
		)
			MTD_CHECK_AUTH_SERVER_RESP(
			#TEST_DP_ADDRESS1,
			<S_SMDP_CHALLENGE>,
			#CTX_PARAMS1)
			SW = 0x9000

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO2)	same EUICCInfo2 data object as in Step IC6 (the extCardResource field SHALL be excluded from the comparison) SW = 0x9000

4.2.13.2.5 VOID

4.2.13.2.6 VOID

4.2.13.2.7 VOID

4.2.13.2.8 TC_eUICC_ES10b.GetEUICCInfo2_RSP_V3.x

Test Sequence #01 Nominal – RSP Version 3.x

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1 Verify if: • <EUICC_RSP_CAPABILITY> is present. #IUT_RSP_VERSION_HIGHEST is equal to 0x030100
2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO2)	#R_EUICC_INFO2 Verify if: • #IUT_RSP_VERSION_HIGHEST is equal to 0x030100 • #IUT_SIMA_VERSION is equal to 0x020301 or higher but less than 0x030000 • #IUT_GLOBALPLATFORM_VERSION is equal to 0x020300 or higher • <EXT_CARD_RESOURCE> contains the "number of installed application" value field is set to '00' • <EUICC_RSP_CAPABILITY> contains ○ crlSupport set to '0' ○ rpmSupport set to '1' if O_E_RPM is supported (otherwise, it SHALL be set to '0') ○ additionalProfile set to '1' ○ deviceInfoExtensibilitySupport set to '1'. ○ serviceSpecificDataSupport bit set to '1' ○ hriServerAddressSupport bit set to '1' if and only if O_E_HRI_ADDRESS_IN_PM is supported. ○ serviceProviderMessageSupport bit set to '1'. ○ lpaProxySupport bit set to '1' if and only if O_E_LPA_PROXY is supported. ○ enterpriseProfilesSupport bit set to '1' if and only if O_E_ENTERPRISE is supported. ○ serviceDescriptionSupport bit set to '1' if and only if

			O_E_SERVICE_DESCRIPTION_IN_PM is supported. ○ deviceChangeSupport bit set to '1' if and only if O_E_DEVICE_CHANGE. ○ estimatedProfileSizeIndicationSupport bit set to '1' if and only if O_E_PROFILE_SIZE_IN_PM is supported. ○ profileSizeInProfilesInfoSupport bit set to '1' if and only if O_E_PROFILE_SIZE_IN_PROFILE_INFO is supported. ○ crlStaplingV3Support bit set to '1'. ○ certChainV3VerificationSupport bit set to '1'. ○ signedSmdsResponseV3Support bit set to '1'. ○ euiccRspCapInInfo1 bit set to '1' and <EUICC_RSP_CAPABILITY> is the same as in Step 1. ○ osUpdateSupport bit set to '1' if and only if O_E_OS_UPDATE is supported. ○ cancelForEmptySpnPnSupport bit set to '1'. ○ updateNotificationConfigurationInfoSupport bit set to '1'. ○ updateMetadataV3Support bit set to '1'. • #IUT_UICC_CAPABILITY contains ○ akaMilenage set to '1' ○ Either akaTuak128 or akaTuak256 set to '1' • #IUT_TRE_PROPERTIES is present and equal to { isDiscrete }, { isIntegrated } or { isIntegrated, usesRemoteMemory } • #IUT_TRE_REFERENCE is not present if #IUT_TRE_PROPERTIES is equal to { isDiscrete } SW = 0x9000

4.2.13.2.9 TC_eUICC_ES10b.GetEUICCInfo2_RSP_V3.x_Integrated_eUICC

Test Sequence #01 Nominal

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO2)	#R_EUICC_INFO2 Verify that:

			• The treProperties field contains ○ isIntegrated set to '1' and ○ isDiscrete set to '0' • The treProductReference field is present and not empty SW = 0x9000 NOTE: usesRemoteMemory can be set to either '0' or '1'.

4.2.14 ES10b (LPA -- eUICC): ListNotification

4.2.14.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 2.5.6
- Section 3.1.3.3
- Section 3.5
- Section 5.7.9

4.2.14.2 Test Cases

Throughout all the ListNotification test cases the maximum number of Notifications simultaneously tested has been set as to two as there is not minimum defined in SGP.21 [3] or SGP.22 [2] for the number of Notifications that can be stored by the eUICC.

4.2.14.2.1 TC_eUICC_ES10b.ListNotification

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	No Operational Profile is installed on the eUICC.
eUICC	No Notifications are stored in the eUICC's Pending Notifications List.

Test Sequence #01 Nominal: Install Notification

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1. Do not remove both the Notifications.		

1	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_IN1_IN1_PIR
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_OMITTED)	#R_LIST_NOTIF_IN1_IN1_PIR
			SW = 0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_NONE)	#R_LIST_NOTIF_NONE
			SW = 0x9000
			OR
			#R_LIST_NOTIF_UNDEFINED_ERROR
			SW = 0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL)	#R_LIST_NOTIF_IN1_IN1_PIR
			SW = 0x9000

5	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ENABLE)	#R_LIST_NOTIF_NONE
			SW = 0x9000

6	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE)	#R_LIST_NOTIF_NONE
			SW = 0x9000

7	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DELETE)	#R_LIST_NOTIF_NONE
			SW = 0x9000

8	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL_ENABLE)	#R_LIST_NOTIF_IN1_IN1_PIR
			SW = 0x9000

9	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE_DELETE)	#R_LIST_NOTIF_NONE
			SW = 0x9000

10	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE_ENABLE)	#R_LIST_NOTIF_NONE
			SW = 0x9000

11	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL_ENABLE_DISABLE)	#R_LIST_NOTIF_IN1_IN1_PIR
			SW = 0x9000

Test Sequence #02 Nominal: Enable Notification

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1. Remove both the Notifications.		
IC4	Enable PROFILE_OPERATIONAL1. Do not remove the Notification.		

1	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_EN1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_OMITTED)	#R_LIST_NOTIF_EN1
			SW = 0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_NONE)	#R_LIST_NOTIF_NONE
			SW = 0x9000
			OR
			#R_LIST_NOTIF_UNDEFINED_ERROR
			SW = 0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL)	#R_LIST_NOTIF_NONE SW = 0x9000
5	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ENABLE)	#R_LIST_NOTIF_EN1 SW = 0x9000
6	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE)	#R_LIST_NOTIF_NONE SW = 0x9000
7	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DELETE)	#R_LIST_NOTIF_NONE SW = 0x9000
8	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL_ENABLE)	#R_LIST_NOTIF_EN1 SW = 0x9000
9	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE_DELETE)	#R_LIST_NOTIF_NONE SW = 0x9000
10	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE_ENABLE)	#R_LIST_NOTIF_EN1 SW = 0x9000
11	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL_ENABLE_DISABLE)	#R_LIST_NOTIF_EN1 SW = 0x9000

Test Sequence #03 Nominal: Disable Notification

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1. Remove both the Notifications.		
IC4	Enable PROFILE_OPERATIONAL1. Remove the Notification.		
IC5	Disable PROFILE_OPERATIONAL1. Do not remove the Notification.		
1	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_DI1 SW = 0x9000
2	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_OMITTED)	#R_LIST_NOTIF_DI1 SW = 0x9000
3	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_NONE)	#R_LIST_NOTIF_NONE SW = 0x9000 OR #R_LIST_NOTIF_UNDEFINED_ERROR SW = 0x9000
4	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL)	#R_LIST_NOTIF_NONE SW = 0x9000
5	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ENABLE)	#R_LIST_NOTIF_NONE SW = 0x9000
6	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE)	#R_LIST_NOTIF_DI1 SW = 0x9000

7	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DELETE)	#R_LIST_NOTIF_NONE SW = 0x9000
8	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL_ENABLE)	#R_LIST_NOTIF_NONE SW = 0x9000
9	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE_DELETE)	#R_LIST_NOTIF_DI1 SW = 0x9000
10	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE_ENABLE)	#R_LIST_NOTIF_DI1 SW = 0x9000
11	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL_ENABLE_DISABLE)	#R_LIST_NOTIF_DI1 SW = 0x9000

Test Sequence #04 Nominal: Delete Notification

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1. Remove both the Notifications.		
IC4	Enable PROFILE_OPERATIONAL1. Remove the Notification.		
IC5	Disable PROFILE_OPERATIONAL1. Remove the Notification.		
IC6	Delete PROFILE_OPERATIONAL1. Do not remove the Notification.		
1	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_DE1 SW = 0x9000
2	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_OMITTED)	#R_LIST_NOTIF_DE1 SW = 0x9000
3	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_NONE)	#R_LIST_NOTIF_NONE SW = 0x9000 OR #R_LIST_NOTIF_UNDEFINED_ERROR SW = 0x9000
4	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL)	#R_LIST_NOTIF_NONE SW = 0x9000
5	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ENABLE)	#R_LIST_NOTIF_NONE SW = 0x9000
6	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE)	#R_LIST_NOTIF_NONE SW = 0x9000
7	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DELETE)	#R_LIST_NOTIF_DE1 SW = 0x9000
8	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL_ENABLE)	#R_LIST_NOTIF_NONE SW = 0x9000
9	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE_DELETE)	#R_LIST_NOTIF_DE1 SW = 0x9000

10	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE_ENABLE)	#R_LIST_NOTIF_NONE SW = 0x9000
11	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ENABLE_DISABLE_DELETE)	#R_LIST_NOTIF_DE1 SW = 0x9000

Test Sequence #05 Nominal: Two Install Notifications (PIR) with different Notification Address

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1 with #METADATA_OP_PROF1_NO_INSTALL. Do not remove the Notification.		
IC4	Install PROFILE_OPERATIONAL2 with #METADATA_OP_PROF2_NO_INSTALL. The default Profile downloading procedure defined in section 2.2.3.1 SHALL be used with the following exceptions: • #CERT_S_SM_DP2auth_SIG SHALL be set in `MTD_AUTHENTICATE_SMDP` rather than #CERT_S_SM_DPauth_SIG • #TEST_DP_ADDRESS2 SHALL be set in `MTD_AUTHENTICATE_SMDP` rather than #TEST_DP_ADDRESS1 • #CERT_S_SM_DP2pb_SIG SHALL be set in `#PREP_DOWNLOAD_NO_CC` rather than #CERT_S_SM_DPpb_SIG Do not remove the Notification.		
1	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_IN1_PIR_IN2_PIR SW = 0x9000
2	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_OMITTED)	#R_LIST_NOTIF_IN1_PIR_IN2_PIR SW = 0x9000
3	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_NONE)	#R_LIST_NOTIF_NONE SW = 0x9000 OR #R_LIST_NOTIF_UNDEFINED_ERROR SW = 0x9000
4	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL)	#R_LIST_NOTIF_IN1_PIR_IN2_PIR SW = 0x9000
5	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ENABLE)	#R_LIST_NOTIF_NONE SW = 0x9000
6	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE)	#R_LIST_NOTIF_NONE SW = 0x9000
7	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DELETE)	#R_LIST_NOTIF_NONE SW = 0x9000
8	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL_ENABLE)	#R_LIST_NOTIF_IN1_PIR_IN2_PIR SW = 0x9000
9	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE_DELETE)	#R_LIST_NOTIF_NONE SW = 0x9000

10	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE_ENABLE)	#R_LIST_NOTIF_NONE SW = 0x9000
11	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL_ENABLE_DISABLE)	#R_LIST_NOTIF_IN1_PIR_IN2_PIR SW = 0x9000

Test Sequence #06 Nominal: Install Notification (PIR) and Enable Notification

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1 with #METADATA_OP_PROF1_NO_INSTALL. Do not remove the Notification.		
IC4	Enable PROFILE_OPERATIONAL1. Do not remove the Notification.		
1	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_IN1_PIR_EN1 SW = 0x9000
2	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_OMITTED)	#R_LIST_NOTIF_IN1_PIR_EN1 SW = 0x9000
3	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_NONE)	#R_LIST_NOTIF_NONE SW = 0x9000 OR #R_LIST_NOTIF_UNDEFINED_ERROR SW = 0x9000
4	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL)	#R_LIST_NOTIF_IN1_PIR SW = 0x9000
5	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ENABLE)	#R_LIST_NOTIF_EN1 SW = 0x9000
6	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE)	#R_LIST_NOTIF_NONE SW = 0x9000
7	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DELETE)	#R_LIST_NOTIF_NONE SW = 0x9000
8	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL_ENABLE)	#R_LIST_NOTIF_IN1_PIR_EN1 SW = 0x9000
9	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE_DELETE)	#R_LIST_NOTIF_NONE SW = 0x9000
10	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE_ENABLE)	#R_LIST_NOTIF_EN1 SW = 0x9000
11	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL_ENABLE_DISABLE)	#R_LIST_NOTIF_IN1_PIR_EN1 SW = 0x9000

Test Sequence #07 Nominal: Disable and Delete Notifications

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1. Remove both the notifications.		
IC4	Enable PROFILE_OPERATIONAL1. Remove the notification		
IC5	Disable PROFILE_OPERATIONAL1. Do not remove the notification		
IC6	Delete PROFILE_OPERATIONAL1. Do not remove the Notification		

1	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_DI1_DE1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_OMITTED)	#R_LIST_NOTIF_DI1_DE1
			SW = 0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_NONE)	#R_LIST_NOTIF_NONE
			SW = 0x9000
			OR
			#R_LIST_NOTIF_UNDEFINED_ERROR
			SW = 0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL)	#R_LIST_NOTIF_NONE
			SW = 0x9000

5	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ENABLE)	#R_LIST_NOTIF_NONE
			SW = 0x9000

6	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE)	#R_LIST_NOTIF_DI1
			SW = 0x9000

7	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DELETE)	#R_LIST_NOTIF_DE1
			SW = 0x9000

8	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL_ENABLE)	#R_LIST_NOTIF_NONE
			SW = 0x9000

9	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE_DELETE)	#R_LIST_NOTIF_DI1_DE1
			SW = 0x9000

10	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE_ENABLE)	#R_LIST_NOTIF_DI1
			SW = 0x9000

11	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL_ENABLE_DISABLE)	#R_LIST_NOTIF_DI1
			SW = 0x9000

Test Sequence #08 Nominal: Install (OtherSignedNotification) and Enable Notification

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		

IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1. Remove the PIR notification, but do not remove the OtherSignedNotification.		
IC4	Enable PROFILE_OPERATIONAL1. Do not remove the Notification.		

1	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_IN1_EN1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_OMITTED)	#R_LIST_NOTIF_IN1_EN1
			SW = 0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_NONE)	#R_LIST_NOTIF_NONE SW = 0x9000 OR #R_LIST_NOTIF_UNDEFINED_ERROR SW = 0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL)	#R_LIST_NOTIF_IN1
			SW = 0x9000

5	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ENABLE)	#R_LIST_NOTIF_EN1
			SW = 0x9000

6	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE)	#R_LIST_NOTIF_NONE
			SW = 0x9000

7	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DELETE)	#R_LIST_NOTIF_NONE
			SW = 0x9000

8	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL_ENABLE)	#R_LIST_NOTIF_IN1_EN1
			SW = 0x9000

9	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE_DELETE)	#R_LIST_NOTIF_NONE
			SW = 0x9000

10	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE_ENABLE)	#R_LIST_NOTIF_EN1
			SW = 0x9000

11	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL_ENABLE_DISABLE)	#R_LIST_NOTIF_IN1_EN1
			SW = 0x9000

Test Sequence #09 Nominal: Enable and Install (PIR) Notifications

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1. Remove both notifications.		
IC4	Enable PROFILE_OPERATIONAL1. Do not remove the Notification.		
IC5	Install PROFILE_OPERATIONAL2 with #METADATA_OP_PROF2_NO_INSTALL. The default Profile downloading procedure defined in section 2.2.3.1 SHALL be used with the following exceptions: • #CERT_S_SM_DP2auth_SIG SHALL be set in `MTD_AUTHENTICATE_SMDP` rather than #CERT_S_SM_DPauth_SIG		

	• #TEST_DP_ADDRESS2 SHALL be set in `MTD_AUTHENTICATE_SMDP` rather than #TEST_DP_ADDRESS1 • #CERT_S_SM_DP2pb_SIG SHALL be set in `#PREP_DOWNLOAD_NO_CC` rather than #CERT_S_SM_DPpb_SIG Do not remove the Notification.		

1	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_EN1_IN2_PIR
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_OMITTED)	#R_LIST_NOTIF_EN1_IN2_PIR
			SW = 0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_NONE)	#R_LIST_NOTIF_NONE
			SW = 0x9000
			OR
			#R_LIST_NOTIF_UNDEFINED_ERROR
			SW = 0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL)	#R_LIST_NOTIF_IN2_PIR
			SW = 0x9000

5	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ENABLE)	#R_LIST_NOTIF_EN1
			SW = 0x9000

6	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE)	#R_LIST_NOTIF_NONE
			SW = 0x9000

7	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DELETE)	#R_LIST_NOTIF_NONE
			SW = 0x9000

8	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL_ENABLE)	#R_LIST_NOTIF_EN1_IN2_PIR
			SW = 0x9000

9	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE_DELETE)	#R_LIST_NOTIF_NONE
			SW = 0x9000

10	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE_ENABLE)	#R_LIST_NOTIF_EN1
			SW = 0x9000

11	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL_ENABLE_DISABLE)	#R_LIST_NOTIF_EN1_IN2_PIR
			SW = 0x9000

Test Sequence #10 Nominal: No Notifications available

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_NONE
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_OMITTED)	#R_LIST_NOTIF_NONE
			SW = 0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_NONE)	#R_LIST_NOTIF_NONE
			SW = 0x9000
			OR

			#R_LIST_NOTIF_UNDEFINED_ERRO R
			SW = 0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL)	#R_LIST_NOTIF_NONE
			SW = 0x9000

5	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ENABLE)	#R_LIST_NOTIF_NONE
			SW = 0x9000

6	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE)	#R_LIST_NOTIF_NONE
			SW = 0x9000

7	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DELETE)	#R_LIST_NOTIF_NONE
			SW = 0x9000

8	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL_ENABLE)	#R_LIST_NOTIF_NONE
			SW = 0x9000

9	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE_DELETE)	#R_LIST_NOTIF_NONE
			SW = 0x9000

10	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE_ENABLE)	#R_LIST_NOTIF_NONE
			SW = 0x9000

11	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL_ENABLE_DISAB LE)	#R_LIST_NOTIF_NONE
			SW = 0x9000

Test Sequence #11 Nominal: Install Notification, different SM-DP+ Addresses in PIR and Install Notification

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1 with #METADATA_OP_PROF1_INST_DIFF instead of #METADATA_OP_PROF1. Do not remove both the Notifications.		

1	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_IN1_DP1_PIR_IN1_DP2_ OSN
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_OMITTED)	#R_LIST_NOTIF_IN1_DP1_PIR_IN1_DP2_ OSN
			SW = 0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_NONE)	#R_LIST_NOTIF_NONE
			SW = 0x9000
			OR
			#R_LIST_NOTIF_UNDEFINED_ERROR
			SW = 0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL)	#R_LIST_NOTIF_IN1_DP1_PIR_IN1_DP2_ OSN
			SW = 0x9000

5	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ENABLE)	#R_LIST_NOTIF_NONE SW = 0x9000
6	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE)	#R_LIST_NOTIF_NONE SW = 0x9000
7	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DELETE)	#R_LIST_NOTIF_NONE SW = 0x9000
8	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL_ENABLE)	#R_LIST_NOTIF_IN1_DP1_PIR_IN1_DP2_OSN SW = 0x9000
9	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE_DELETE)	#R_LIST_NOTIF_NONE SW = 0x9000
10	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE_ENABLE)	#R_LIST_NOTIF_NONE SW = 0x9000
11	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_INSTALL_ENABLE_DISABLE)	#R_LIST_NOTIF_IN1_DP1_PIR_IN1_DP2_OSN SW = 0x9000

4.2.14.2.2 TC_eUICC_ES10b.ListNotification_RPM

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	No Operational Profile is installed on the eUICC.
eUICC	No Notification is stored in the eUICC's Pending Notifications List.

Test Sequence #01 Nominal: RPM Enable Notification

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1 with #METADATA_OP_PROF1_RPM_CONF. Remove the PIR Notification.		
IC4	Enable PROFILE_OPERATIONAL1 via RPM as defined in section 2.2.3.4. Remove the RPR Notification, but do not remove the OtherSignedNotification.		
1	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL_RPM)	#R_LIST_NOTIF_EN1_RPM SW = 0x9000
2	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_OMITTED)	#R_LIST_NOTIF_EN1_RPM SW = 0x9000
5	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ENABLE_RPM)	#R_LIST_NOTIF_EN1_RPM SW = 0x9000
6	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE_RPM)	#R_LIST_NOTIF_NONE SW = 0x9000
7	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DELETE_RPM)	#R_LIST_NOTIF_NONE SW = 0x9000

9	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE_DELETE_RPM)	#R_LIST_NOTIF_NONE SW = 0x9000
10	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE_ENABLE_RPM)	#R_LIST_NOTIF_EN1_RPM SW = 0x9000
11	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DELETE_ENABLE_RPM)	#R_LIST_NOTIF_EN1_RPM SW = 0x9000

Test Sequence #02 Nominal: RPM Enable Notification – RPR and RemoteEnable OSN to different SM-DP+

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1 with #METADATA_OP_PROF1_RPM_CONF_OSN_DP2. Remove the PIR Notification.		
IC4	Enable PROFILE_OPERATIONAL1 via RPM as defined in section 2.2.3.4. Do not remove the RPR Notification and the OtherSignedNotifications.		
1	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL_RPM)	#R_LIST_NOTIF_RPR_EN1_RPM_DP2 SW = 0x9000
2	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_OMITTED)	#R_LIST_NOTIF_RPR_EN1_RPM_DP2 SW = 0x9000
3	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ENABLE_RPM)	#R_LIST_NOTIF_EN1_RPM_DP2 SW = 0x9000
4	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_RPR_ENABLE_RPM)	#R_LIST_NOTIF_RPR_EN1_RPM_DP2 SW = 0x9000
5	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_RPR_DISABLE_RPM)	#R_LIST_NOTIF_RPR SW = 0x9000
6	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_RPR_DELETE_RPM)	#R_LIST_NOTIF_RPR SW = 0x9000
7	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_RPR_DISABLE_DELETE_RPM)	#R_LIST_NOTIF_RPR SW = 0x9000
8	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_RPR_DISABLE_ENABLE_RPM)	#R_LIST_NOTIF_RPR_EN1_RPM_DP2 SW = 0x9000
9	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_RPR_DELETE_ENABLE_RPM)	#R_LIST_NOTIF_RPR_EN1_RPM_DP2 SW = 0x9000
10	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_DISABLE_DELETE_RPM)	#R_LIST_NOTIF_NONE SW = 0x9000

4.2.15 ES10b (LPA -- eUICC): RetrieveNotificationsList

4.2.15.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 2.5.6
- Section 2.6.7.2
- Section 3.1.3.3
- Section 3.5
- Section 5.7.10

4.2.15.2 Test Cases

Throughout all the RetrieveNotificationsList test cases the maximum number of Notifications simultaneously tested has been set to two as there is no minimum defined in SGP.21 [3] or SGP.22 [2] for the number of Notifications that can be stored by the eUICC.

In some test sequences defined below, it is expected to retrieve especially either a `ProfileInstallationResult` or an `OtherSignedNotification` for installation. When both are present in the eUICC, the only way to distinguish these two notifications is to compare their sequence numbers in the `ListNotificationResponse`. The sequence number related to the `ProfileInstallationResult` SHALL be lower than the one linked to the `OtherSignedNotification`.

This assumption is based on the requirement defined in section 5.5.5 of SGP.22 [2] stating that the eUICC SHALL first generate the Profile Installation Result and then as many Notifications as configured in its metadata in the format of `OtherSignedNotification`.

4.2.15.2.1 TC_eUICC_ES10b.RetrieveNotificationsList

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	No Operational Profile is installed on the eUICC.
eUICC	No Notifications are stored in the eUICC's Pending Notifications List.

Test Sequence #01 Nominal: Retrieve by Sequence Number for Install Notification

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1. Do not remove both the notifications.		
IC4	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_IN1_IN1_PIR SW = 0x9000

1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_RETRIEVE_NOTIF_SEQ_NUM(<NOTIF_SEQ_NO_IN1>))	#R_RETRIEVE_NOTIF_IN1_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_RETRIEVE_NOTIF_SEQ_NUM(<NOTIF_SEQ_NO_IN1_PIR>))	#R_RETRIEVE_NOTIF_IN1_PIR
			SW = 0x9000
			• Verify the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG

Test Sequence #02 Nominal: Retrieve by Sequence Number for Enable Notification

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1. Remove both the notifications.		
IC4	Enable PROFILE_OPERATIONAL1. Do not remove the Notification.		

IC5	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_EN1
			SW = 0x9000

1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_RETRIEVE_NOTIF_SEQ_NUM(<NOTIF_SEQ_NO_EN1>))	#R_RETRIEVE_NOTIF_EN1_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

Test Sequence #03 Nominal: Retrieve by Sequence Number for Disable Notification

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1. Remove both the notifications.		
IC4	Enable PROFILE_OPERATIONAL1. Remove the Notification.		
IC5	Disable PROFILE_OPERATIONAL1. Do not remove the Notification.		

IC6	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_DI1
			SW = 0x9000

1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_RETRIEVE_NOTIF_SEQ_NUM(<NOTIF_SEQ_NO_DI1>))	#R_RETRIEVE_NOTIF_DI1_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

Test Sequence #04 Nominal: Retrieve by Sequence Number for Delete Notification

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1. Remove both the notifications.		
IC4	Enable PROFILE_OPERATIONAL1. Remove the Notification.		
IC5	Disable PROFILE_OPERATIONAL1. Remove the Notification.		
IC6	Delete PROFILE_OPERATIONAL1. Do not remove the Notification.		
IC7	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_DE1 SW = 0x9000
1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_RETRIEVE_NOTIF_SEQ_NUM(<NOTIF_SEQ_NO_DE1>))	#R_RETRIEVE_NOTIF_DE1_V3 SW = 0x9000 • Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

Test Sequence #05 Nominal: Retrieve by Sequence Number for Two Install (PIR) Notifications with different Notification Addresses

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1 with #METADATA_OP_PROF1_NO_INSTALL. Do not remove the Notification.		
IC4	Install PROFILE_OPERATIONAL2 with #METADATA_OP_PROF2_NO_INSTALL. The default Profile downloading procedure defined in section 2.2.3.1 SHALL be used with the following exceptions: • #CERT_S_SM_DP2auth_SIG SHALL be set in `MTD_AUTHENTICATE_SMDP` rather than #CERT_S_SM_DPauth_SIG • #TEST_DP_ADDRESS2 SHALL be set in `MTD_AUTHENTICATE_SMDP` rather than #TEST_DP_ADDRESS1 • #CERT_S_SM_DP2pb_SIG SHALL be set in `#PREP_DOWNLOAD_NO_CC` rather than #CERT_S_SM_DPpb_SIG Do not remove the Notification.		
IC5	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_IN1_PIR_IN2_PIR SW = 0x9000
1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_RETRIEVE_NOTIF_SEQ_NUM(<NOTIF_SEQ_NO_IN1_PIR>))	#R_RETRIEVE_NOTIF_IN1_PIR SW = 0x9000 • Verify the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG
2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_RETRIEVE_NOTIF_SEQ_NUM(<NOTIF_SEQ_NO_IN2_PIR>))	#R_RETRIEVE_NOTIF_IN2_PIR SW = 0x9000

			• Verify the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG

Test Sequence #06 Nominal: Retrieve by Sequence Number for Install (PIR) and Enable Notifications

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1 with #METADATA_OP_PROF1_NO_INSTALL. Do not remove the Notification.		
IC4	Enable PROFILE_OPERATIONAL1. Do not remove the Notification.		

IC5	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_IN1_PIR_EN1
			SW = 0x9000

1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_RETRIEVE_NOTIF_SEQ_NUM(<NOTIF_SEQ_NO_IN1_PIR>))	#R_RETRIEVE_NOTIF_IN1_PIR
			SW = 0x9000
			• Verify the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_RETRIEVE_NOTIF_SEQ_NUM(<NOTIF_SEQ_NO_EN1>))	#R_RETRIEVE_NOTIF_EN1_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

Test Sequence #07 Nominal: Retrieve by Sequence Number for Disable and Delete Notifications

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1. Remove both the notifications.		
IC4	Enable PROFILE_OPERATIONAL1. Remove the notification		
IC5	Disable PROFILE_OPERATIONAL1. Do not remove the notification		
IC6	Delete PROFILE_OPERATIONAL1. Do not remove the Notification		

IC7	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_DI1_DE1
			SW = 0x9000

1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_RETRIEVE_NOTIF_SEQ_NUM(<NOTIF_SEQ_NO_DI1>))	#R_RETRIEVE_NOTIF_DI1_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_RETRIEVE_NOTIF_SEQ_NUM(<NOTIF_SEQ_NO_DE1>))	#R_RETRIEVE_NOTIF_DE1_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

Test Sequence #08 Nominal: Retrieve by Sequence Number for Install (OtherSignedNotification) and Enable Notifications

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1. Remove the PIR notification, but do not remove the OtherSignedNotification.		
IC4	Enable PROFILE_OPERATIONAL1. Do not remove the Notification.		

IC5	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_IN1_EN1
			SW = 0x9000

1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_RETRIEVE_NOTIF_SEQ_NUM(<NOTIF_SEQ_NO_IN1>))	#R_RETRIEVE_NOTIF_IN1_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_RETRIEVE_NOTIF_SEQ_NUM(<NOTIF_SEQ_NO_EN1>))	#R_RETRIEVE_NOTIF_EN1_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

Test Sequence #09 Nominal: Retrieve by Sequence Number for Enable and Install (PIR) notifications

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1. Remove both notifications.		
IC4	Enable PROFILE_OPERATIONAL1. Do not remove the Notification.		
IC5	Install PROFILE_OPERATIONAL2 with #METADATA_OP_PROF2_NO_INSTALL. The default Profile downloading procedure defined in section 2.2.3.1 SHALL be used with the following exceptions: • #CERT_S_SM_DP2auth_SIG SHALL be set in `MTD_AUTHENTICATE_SMDP` rather than #CERT_S_SM_DPauth_SIG • #TEST_DP_ADDRESS2 SHALL be set in `MTD_AUTHENTICATE_SMDP` rather than #TEST_DP_ADDRESS1		

	• #CERT_S_SM_DP2pb_SIG SHALL be set in `#PREP_DOWNLOAD_NO_CC` rather than #CERT_S_SM_DPpb_SIG Do not remove the Notification.		

IC6	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_EN1_IN2_PIR
			SW = 0x9000

1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_RETRIEVE_NOTIF_SEQ_NUM(<NOTIF_SEQ_NO_IN2_PIR>))	#R_RETRIEVE_NOTIF_IN2_PIR
			SW = 0x9000
			• Verify the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_RETRIEVE_NOTIF_SEQ_NUM(<NOTIF_SEQ_NO_EN1>))	#R_RETRIEVE_NOTIF_EN1_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

Test Sequence #10 Nominal: Retrieve Sequence Numbers that are not present

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1 with #METADATA_OP_PROF1_NO_INSTALL. Do not remove the Notification.		

IC4	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_IN1_PIR
			SW = 0x9000

1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_RETRIEVE_NOTIF_SEQ_NUM(<NOTIF_SEQ_NO_IN1_PIR>))	#R_RETRIEVE_NOTIF_IN1_PIR
			SW = 0x9000
			• Verify the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_RETRIEVE_NOTIF_SEQ_NUM(<NOTIF_SEQ_NO_IN1_PIR> +1))	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

Test Sequence #11 Nominal: Retrieve by Notification Type for Install Notifications

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1. Do not remove both the notifications.		

1	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_ALL)	#R_RETRIEVE_NOTIF_IN1_IN1_PIR
			SW = 0x9000
			• Verify the euiccNotificationSignature

			<TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG • Verify the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_OMITTED)	#R_RETRIEVE_NOTIF_IN1_IN1_ PIR
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG
			• Verify the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG

3	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_NONE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL)	#R_RETRIEVE_NOTIF_IN1_IN1_ PIR
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG
			• Verify the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG

5	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_ENABLE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

6	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

7	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DELETE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

8	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL _ENABLE)	#R_RETRIEVE_NOTIF_IN1_IN1_ PIR
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG
			• Verify the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG

9	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE_DELETE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

10	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE_ENABLE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

11	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL_ENABLE_DIS ABLE)	#R_RETRIEVE_NOTIF_IN1_IN1_ PIR
			SW = 0x9000
			• Verify the euiccNotificationSignature

			<TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG • Verify the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG

Test Sequence #12 Nominal: Retrieve by Notification Type for Enable Notification

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1. Remove both the notifications.		
IC4	Enable PROFILE_OPERATIONAL1. Do not remove the Notification.		
1	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_ALL)	#R_RETRIEVE_NOTIF_EN1_V3 SW = 0x9000 • Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG
2	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_OMITTED)	#R_RETRIEVE_NOTIF_EN1_V3 SW = 0x9000
3	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_NONE)	#R_RETRIEVE_NOTIF_NONE SW = 0x9000
4	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL)	#R_RETRIEVE_NOTIF_NONE SW = 0x9000
5	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_ENABLE)	#R_RETRIEVE_NOTIF_EN1_V3 SW = 0x9000
6	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE)	#R_RETRIEVE_NOTIF_NONE SW = 0x9000
7	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DELETE)	#R_RETRIEVE_NOTIF_NONE SW = 0x9000
8	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL_ENABLE)	#R_RETRIEVE_NOTIF_EN1_V3 SW = 0x9000
9	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE_DELETE)	#R_RETRIEVE_NOTIF_NONE SW = 0x9000
10	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE_ENABLE)	#R_RETRIEVE_NOTIF_EN1_V3 SW = 0x9000
11	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL_ENABLE_ DISABLE)	#R_RETRIEVE_NOTIF_EN1_V3 SW = 0x9000

Test Sequence #13 Nominal: Retrieve by Notification Type for Disable Notification

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1. Remove both the notifications.		
IC4	Enable PROFILE_OPERATIONAL1. Remove the Notification.		

IC5	Disable PROFILE_OPERATIONAL1. Do not remove the Notification.		

1	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_ALL)	#R_RETRIEVE_NOTIF_DI1_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_OMITTED)	#R_RETRIEVE_NOTIF_DI1_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

3	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_NONE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

5	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_ENABLE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

6	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE)	#R_RETRIEVE_NOTIF_DI1_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

7	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DELETE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

8	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL_ENABLE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

9	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE_DELETE)	#R_RETRIEVE_NOTIF_DI1_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

10	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE_ENABLE)	#R_RETRIEVE_NOTIF_DI1_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

11	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL_ENABLE_DISABLE)	#R_RETRIEVE_NOTIF_DI1_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature

			<TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

Test Sequence #14 Nominal: Retrieve by Notification Type for Delete Notification

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1. Remove both the notifications.		
IC4	Enable PROFILE_OPERATIONAL1. Remove the Notification.		
IC5	Disable PROFILE_OPERATIONAL1. Remove the Notification.		
IC6	Delete PROFILE_OPERATIONAL1. Do not remove the Notification.		

1	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_ALL)	#R_RETRIEVE_NOTIF_DE1_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_OMITTED)	#R_RETRIEVE_NOTIF_DE1_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

3	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_NONE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

5	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_ENABLE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

6	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

7	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DELETE)	#R_RETRIEVE_NOTIF_DE1_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

8	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL_ENABLE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

9	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE_DELETE)	#R_RETRIEVE_NOTIF_DE1_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature

			<TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG
10	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE_ENABLE)	#R_RETRIEVE_NOTIF_NONE SW = 0x9000
11	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL_ENABLE_DISABLE)	#R_RETRIEVE_NOTIF_NONE SW = 0x9000

Test Sequence #15 Nominal: Retrieve by Notification Type for Two Install (PIR) Notifications with different Notification Addresses

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1 with #METADATA_OP_PROF1_NO_INSTALL. Do not remove the Notification.		
IC4	Install PROFILE_OPERATIONAL2 with #METADATA_OP_PROF2_NO_INSTALL. The default Profile downloading procedure defined in section 2.2.3.1 SHALL be used with the following exceptions: • #CERT_S_SM_DP2auth_SIG SHALL be set in `MTD_AUTHENTICATE_SMDP` rather than #CERT_S_SM_DPauth_SIG • #TEST_DP_ADDRESS2 SHALL be set in `MTD_AUTHENTICATE_SMDP` rather than #TEST_DP_ADDRESS1 • #CERT_S_SM_DP2pb_SIG SHALL be set in #PREP_DOWNLOAD_NO_CC rather than #CERT_S_SM_DPpb_SIG • Do not remove the Notification.		

1	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_ALL)	#R_RETRIEVE_NOTIF_IN1_PIR_IN2_PIR
			SW = 0x9000
			• Verify both the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_OMITTED)	#R_RETRIEVE_NOTIF_IN1_PIR_IN2_PIR
			SW = 0x9000
			• Verify both the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG

3	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_NONE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL)	#R_RETRIEVE_NOTIF_IN1_PIR_IN2_PIR
			SW = 0x9000
			• Verify both the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG

5	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_ENABLE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

6	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

7	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DELETE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

8	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL_ENABLE)	#R_RETRIEVE_NOTIF_IN1_PIR_IN2_PIR
			SW = 0x9000
			• Verify both the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG

9	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE_DELETE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

10	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE_ENABLE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

11	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL_ENABLE_DISABLE)	#R_RETRIEVE_NOTIF_IN1_PIR_IN2_PIR
			SW = 0x9000
			• Verify both the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG

Test Sequence #16 Nominal: Retrieve by Notification Type for Install (PIR) and Enable Notifications

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1 with #METADATA_OP_PROF1_NO_INSTALL. Do not remove the Notification.		
IC4	Enable PROFILE_OPERATIONAL1. Do not remove the Notification.		

1	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_ALL)	#R_RETRIEVE_NOTIF_IN1_PIR_EN1_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG
			• Verify the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_OMITTED)	#R_RETRIEVE_NOTIF_IN1_PIR_EN1_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG
			• Verify the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG

3	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_NONE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL)	#R_RETRIEVE_NOTIF_IN1_PIR
			SW = 0x9000

			• Verify the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG
5	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_ENABLE)	#R_RETRIEVE_NOTIF_EN1_V3 SW = 0x9000 • Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG
6	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE)	#R_RETRIEVE_NOTIF_NONE SW = 0x9000
7	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DELETE)	#R_RETRIEVE_NOTIF_NONE SW = 0x9000
8	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL_ENABLE)	#R_RETRIEVE_NOTIF_IN1_PIR_EN1 _V3 SW = 0x9000 • Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG • Verify the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG
9	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE_DELETE)	#R_RETRIEVE_NOTIF_NONE SW = 0x9000
10	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE_ENABLE)	#R_RETRIEVE_NOTIF_EN1_V3 SW = 0x9000 • Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG
11	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL_ENABLE_ DISABLE)	#R_RETRIEVE_NOTIF_IN1_PIR_EN1 _V3 SW = 0x9000 • Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG • Verify the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG

Test Sequence #17 Nominal: Retrieve by Notification Type for Disable and Delete Notifications

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1. Remove both the notifications.		
IC4	Enable PROFILE_OPERATIONAL1. Remove the notification		
IC5	Disable PROFILE_OPERATIONAL1. Do not remove the notification		
IC6	Delete PROFILE_OPERATIONAL1. Do not remove the Notification		

1	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_ALL)	#R_RETRIEVE_NOTIF_DI1_DE1 _V3
			SW = 0x9000
			• Verify both the euiccNotificationSignatures <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_OMITTED)	#R_RETRIEVE_NOTIF_DI1_DE1 _V3
			SW = 0x9000
			• Verify both the euiccNotificationSignatures <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

3	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_NONE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

5	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_ENABLE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

6	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE)	#R_RETRIEVE_NOTIF_DI1_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

7	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DELETE)	#R_RETRIEVE_NOTIF_DE1_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

8	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL_ENABLE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

9	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE_DELETE)	#R_RETRIEVE_NOTIF_DI1_DE1 _V3
			SW = 0x9000
			• Verify both the euiccNotificationSignatures <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

10	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE_ENABLE)	#R_RETRIEVE_NOTIF_DI1_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

11	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL_ENABLE_DISA BLE)	#R_RETRIEVE_NOTIF_DI1_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

Test Sequence #18 Nominal: Retrieve by Notification Type for Install (OtherSignedNotification) and Enable Notifications

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1. Remove the PIR notification, but do not remove the OtherSignedNotification.		
IC4	Enable PROFILE_OPERATIONAL1. Do not remove the Notification.		

1	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_ALL)	#R_RETRIEVE_NOTIF_IN1_EN1_V3
			SW = 0x9000
			• Verify both the euiccNotificationSignatures <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_OMITTED)	#R_RETRIEVE_NOTIF_IN1_EN1_V3
			SW = 0x9000
			• Verify both the euiccNotificationSignatures <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

3	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_NONE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL)	#R_RETRIEVE_NOTIF_IN1_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

5	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_ENABLE)	#R_RETRIEVE_NOTIF_EN1_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

6	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

7	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DELETE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

8	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL_ENABLE)	#R_RETRIEVE_NOTIF_IN1_EN1_V3
			SW = 0x9000
			• Verify both the euiccNotificationSignatures <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

9	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE_DELETE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

10	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE_ENABLE)	#R_RETRIEVE_NOTIF_EN1_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

11	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL_ENABLE_DISABLE)	#R_RETRIEVE_NOTIF_IN1_EN1_V3
			SW = 0x9000
			• Verify both the euiccNotificationSignatures <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

Test Sequence #19 Nominal: Retrieve by Notification Type for Enable and Install (PIR) notifications

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1. Remove both notifications.		
IC4	Enable PROFILE_OPERATIONAL1. Do not remove the Notification.		
IC5	Install PROFILE_OPERATIONAL2 with #METADATA_OP_PROF2_NO_INSTALL. The default Profile downloading procedure defined in section 2.2.3.1 SHALL be used with the following exceptions: • #CERT_S_SM_DP2auth_SIG SHALL be set in `MTD_AUTHENTICATE_SMDP` rather than #CERT_S_SM_DPauth_SIG • #TEST_DP_ADDRESS2 SHALL be set in `MTD_AUTHENTICATE_SMDP` rather than #TEST_DP_ADDRESS1 • #CERT_S_SM_DP2pb_SIG SHALL be set in `#PREP_DOWNLOAD_NO_CC` rather than #CERT_S_SM_DPpb_SIG Do not remove the Notification.		

1	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_ALL)	#R_RETRIEVE_NOTIF_EN1_IN2_PIR_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG
			• Verify the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_OMITTED)	#R_RETRIEVE_NOTIF_EN1_IN2_PIR_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

			• Verify the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG
3	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_NONE)	#R_RETRIEVE_NOTIF_NONE SW = 0x9000
4	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL)	#R_RETRIEVE_NOTIF_IN2_PIR SW = 0x9000 • Verify the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG
5	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_ENABLE)	#R_RETRIEVE_NOTIF_EN1_V3 SW = 0x9000 • Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG
6	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE)	#R_RETRIEVE_NOTIF_NONE SW = 0x9000
7	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DELETE)	#R_RETRIEVE_NOTIF_NONE SW = 0x9000
8	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL_ENABLE)	#R_RETRIEVE_NOTIF_EN1_IN2_ PIR_V3 SW = 0x9000 • Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG • Verify the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG
9	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE_DELETE)	#R_RETRIEVE_NOTIF_NONE SW = 0x9000
10	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE_ENABLE)	#R_RETRIEVE_NOTIF_EN1_V3 SW = 0x9000 • Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG
11	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL_ENABLE_DISABLE)	#R_RETRIEVE_NOTIF_EN1_IN2_ PIR_V3 SW = 0x9000 • Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG • Verify the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG

Test Sequence #20 Nominal: Retrieve by Notification Type for No Notifications available

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_ALL)	#R_RETRIEVE_NOTIF_NONE SW = 0x9000
2	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_OMITTED)	#R_RETRIEVE_NOTIF_NONE SW = 0x9000
3	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_NONE)	#R_RETRIEVE_NOTIF_NONE SW = 0x9000
4	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL)	#R_RETRIEVE_NOTIF_NONE SW = 0x9000
5	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_ENABLE)	#R_RETRIEVE_NOTIF_NONE SW = 0x9000
6	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE)	#R_RETRIEVE_NOTIF_NONE SW = 0x9000
7	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DELETE)	#R_RETRIEVE_NOTIF_NONE SW = 0x9000
8	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL_ENABLE)	#R_RETRIEVE_NOTIF_NONE SW = 0x9000
9	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE_DELETE)	#R_RETRIEVE_NOTIF_NONE SW = 0x9000
10	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE_ENABLE)	#R_RETRIEVE_NOTIF_NONE SW = 0x9000
11	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL_ENABLE_DISABLE)	#R_RETRIEVE_NOTIF_NONE SW = 0x9000

Test Sequence #21 Nominal: Retrieve by Sequence Number for Install Notification, different SM-DP+ Addresses in PIR and Install Notification

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1 with #METADATA_OP_PROF1_INST_DIFF instead of #METADATA_OP_PROF1. Do not remove both the notifications.		
IC4	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_IN1_DP1_PIR_IN 1_DP2_OSN SW = 0x9000
1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_RETRIEVE_NOTIF_SEQ_NUM(<NOTI F_SEQ_NO_IN1_DP1>))	#R_RETRIEVE_NOTIF_IN1_DP1_ PIR SW = 0x9000 • Verify the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_RETRIEVE_NOTIF_SEQ_NUM(<NOTIF_SEQ_NO_IN1_DP2>))	#R_RETRIEVE_NOTIF_IN1_DP2_OSN_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG

Test Sequence #22 Nominal: Retrieve by Notification Type for Install Notifications, different SM-DP+ Addresses in PIR and Install Notification

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1 with #METADATA_OP_PROF1_INST_DIFF instead of #METADATA_OP_PROF1. Do not remove both the notifications.		

1	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_ALL)	#R_RETRIEVE_NOTIF_IN1_DP1_PIR_IN1_DP2_OSN_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG
			• Verify the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_OMITTED)	#R_RETRIEVE_NOTIF_IN1_DP1_PIR_IN1_DP2_OSN_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG
			• Verify the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG

3	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_NONE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL)	#R_RETRIEVE_NOTIF_IN1_DP1_PIR_IN1_DP2_OSN_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG
			• Verify the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG

5	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_ENABLE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

6	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

7	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DELETE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

8	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL _ENABLE)	#R_RETRIEVE_NOTIF_IN1_DP1_PIR_IN1_ DP2_OSN_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG
			• Verify the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG

9	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE_DEL ETE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

10	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_DISABLE_ENA BLE)	#R_RETRIEVE_NOTIF_NONE
			SW = 0x9000

11	S_LPAd → eUICC	MTD_STORE_DATA(#RETRIEVE_NOTIF_INSTALL_ENA BLE_DISABLE)	#R_RETRIEVE_NOTIF_IN1_DP1_PIR_IN1_ DP2_OSN_V3
			SW = 0x9000
			• Verify the euiccNotificationSignature <TBS_EUICC_NOTIF_SIG> using the #PK_EUICC_SIG
			• Verify the euiccSignPIR <EUICC_SIGN_PIR> using the #PK_EUICC_SIG

4.2.16 ES10b (LPA -- eUICC): RemoveNotificationFromList

4.2.16.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 2.5.6
- Section 3.1.3.3
- Section 3.5
- Section 5.7.11

4.2.16.2 Test Cases

Throughout all the RemoveNotificationFromList test cases the maximum number of Notifications simultaneously tested has been set as to two as there is no minimum defined in SGP.21 [3] or SGP.22 [2] for the number of Notifications that can be stored by the eUICC.

The rule specified in section 4.2.15.2 explaining the way to distinguish a `ProfileInstallationResult` from an `OtherSignedNotification` for installation also applies for the test cases defined below.

4.2.16.2.1 TC_eUICC_ES10b.RemoveNotificationFromList

General Initial Conditions	
Entity	**Description of the general initial condition**

eUICC	No Operational Profile is installed on the eUICC.
eUICC	No Notifications are stored in the eUICC's Pending Notifications List.

Test Sequence #01 Nominal: Install Notification

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1. Do not remove both the notifications.		
IC4	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_IN1_IN1_PIR SW = 0x9000
1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_REMOVE_NOTIF(<NOTIF_SEQ_NO_IN1>))	#R_REMOVE_NOTIF_OK SW = 0x9000
2	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_IN1_PIR SW = 0x9000
3	S_LPAd → eUICC	MTD_STORE_DATA(MTD_REMOVE_NOTIF(<NOTIF_SEQ_NO_IN1_PIR>))	#R_REMOVE_NOTIF_OK SW = 0x9000
4	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_NONE SW = 0x9000

Test Sequence #02 Nominal: Enable Notification

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1. Remove both the notifications.		
IC4	Enable PROFILE_OPERATIONAL1. Do not remove the Notification.		
IC5	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_EN1 SW = 0x9000
1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_REMOVE_NOTIF(<NOTIF_SEQ_NO_EN1>))	#R_REMOVE_NOTIF_OK SW = 0x9000
2	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_NONE SW = 0x9000

Test Sequence #03 Nominal: Disable Notification

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

IC3	Install PROFILE_OPERATIONAL1. Remove both the notifications.		
IC4	Enable PROFILE_OPERATIONAL1. Remove the Notification.		
IC5	Disable PROFILE_OPERATIONAL1. Do not remove the Notification.		

IC6	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_DI1
			SW = 0x9000

1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_REMOVE_NOTIF(<NOTIF_SEQ_NO_DI1>))	#R_REMOVE_NOTIF_OK
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_NONE
			SW = 0x9000

Test Sequence #04 Nominal: Delete Notification

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1. Remove both the notifications.		
IC4	Enable PROFILE_OPERATIONAL1. Remove the Notification.		
IC5	Disable PROFILE_OPERATIONAL1. Remove the Notification.		
IC6	Delete PROFILE_OPERATIONAL1. Do not remove the Notification.		

IC7	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_DE1
			SW = 0x9000

1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_REMOVE_NOTIF(<NOTIF_SEQ_NO_DE1>))	#R_REMOVE_NOTIF_OK
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_NONE
			SW = 0x9000

Test Sequence #05 Nominal: Two Install (PIR) Notifications with different Notification Addresses

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1 with #METADATA_OP_PROF1_NO_INSTALL. Do not remove the Notification.		
IC4	Install PROFILE_OPERATIONAL2 with #METADATA_OP_PROF2_NO_INSTALL. The default Profile downloading procedure defined in section 2.2.3.1 SHALL be used with the following exceptions: • #CERT_S_SM_DP2auth_SIG SHALL be set in `MTD_AUTHENTICATE_SMDP` rather than #CERT_S_SM_DPauth_SIG • #TEST_DP_ADDRESS2 SHALL be set in `MTD_AUTHENTICATE_SMDP` rather than #TEST_DP_ADDRESS1		

	• #CERT_S_SM_DP2pb_SIG SHALL be set in `#PREP_DOWNLOAD_NO_CC` rather than #CERT_S_SM_DPpb_SIG Do not remove the Notification.		

IC5	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_IN1_PIR_IN2_PIR
			SW = 0x9000

1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_REMOVE_NOTIF(<NOTIF_SEQ_NO_IN1_PIR>))	#R_REMOVE_NOTIF_OK
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_IN2_PIR
			SW = 0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(MTD_REMOVE_NOTIF(<NOTIF_SEQ_NO_IN2_PIR>))	#R_REMOVE_NOTIF_OK
			SW = 0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_NONE
			SW = 0x9000

Test Sequence #06 Nominal: Install (PIR) and Enable Notifications

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1 with #METADATA_OP_PROF1_NO_INSTALL. Do not remove the Notification.		
IC4	Enable PROFILE_OPERATIONAL1. Do not remove the Notification.		

IC5	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_IN1_PIR_EN1
			SW = 0x9000

1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_REMOVE_NOTIF(<NOTIF_SEQ_NO_IN1_PIR>))	#R_REMOVE_NOTIF_OK
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_EN1
			SW = 0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(MTD_REMOVE_NOTIF(<NOTIF_SEQ_NO_EN1>))	#R_REMOVE_NOTIF_OK
			SW = 0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_NONE
			SW = 0x9000

Test Sequence #07 Nominal: Disable and Delete Notifications

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1. Remove both the Notifications.		
IC4	Enable PROFILE_OPERATIONAL1. Remove the Notification		

IC5	Disable PROFILE_OPERATIONAL1. Do not remove the Notification		
IC6	Delete PROFILE_OPERATIONAL1. Do not remove the Notification		
IC7	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_DI1_DE1 SW = 0x9000
1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_REMOVE_NOTIF(<NOTIF_SEQ_NO_DI1>))	#R_REMOVE_NOTIF_OK SW = 0x9000
2	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_DE1 SW = 0x9000
3	S_LPAd → eUICC	MTD_STORE_DATA(MTD_REMOVE_NOTIF(<NOTIF_SEQ_NO_DE1>))	#R_REMOVE_NOTIF_OK SW = 0x9000
4	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_NONE SW = 0x9000

Test Sequence #08 Nominal: Install (OtherSignedNotification) and Enable Notifications

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1. Remove the PIR notification, but do not remove the OtherSignedNotification.		
IC4	Enable PROFILE_OPERATIONAL1. Do not remove the Notification.		
IC5	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_IN1_EN1 SW = 0x9000
1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_REMOVE_NOTIF(<NOTIF_SEQ_NO_IN1>))	#R_REMOVE_NOTIF_OK SW = 0x9000
2	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_EN1 SW = 0x9000
3	S_LPAd → eUICC	MTD_STORE_DATA(MTD_REMOVE_NOTIF(<NOTIF_SEQ_NO_EN1>))	#R_REMOVE_NOTIF_OK SW = 0x9000
4	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_NONE SW = 0x9000

Test Sequence #09 Nominal: Enable and Install (PIR) notifications

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1. Remove both notifications.		

IC4	Enable PROFILE_OPERATIONAL1. Do not remove the Notification.		
IC5	Install PROFILE_OPERATIONAL2 with METADATA_OP_PROF2_NO_INSTALL. The default Profile downloading procedure defined in section 2.2.3.1 SHALL be used with the following exceptions: • #CERT_S_SM_DP2auth_SIG SHALL be set in `MTD_AUTHENTICATE_SMDP` rather than #CERT_S_SM_DPauth_SIG • #TEST_DP_ADDRESS2 SHALL be set in `MTD_AUTHENTICATE_SMDP` rather than #TEST_DP_ADDRESS1 • #CERT_S_SM_DP2pb_SIG SHALL be set in `#PREP_DOWNLOAD_NO_CC` rather than #CERT_S_SM_DPpb_SIG Do not remove the Notification.		

IC6	S_LPAd → eUICC	MTD_STORE_DATA(
		#LIST_NOTIF_ALL)
			#R_LIST_NOTIF_EN1_IN2_PIR
			SW = 0x9000

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_REMOVE_NOTIF(
		<NOTIF_SEQ_NO_IN2_PIR>))
			#R_REMOVE_NOTIF_OK
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#LIST_NOTIF_ALL)
			#R_LIST_NOTIF_EN1
			SW = 0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_REMOVE_NOTIF(
		<NOTIF_SEQ_NO_EN1>))
			#R_REMOVE_NOTIF_OK
			SW = 0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(
		#LIST_NOTIF_ALL)
			#R_LIST_NOTIF_NONE
			SW = 0x9000

Test Sequence #10 Nominal: Removing Sequence Numbers that are not present

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL1 with #METADATA_OP_PROF1_NO_INSTALL. Do not remove the Notification.		

IC4	S_LPAd → eUICC	MTD_STORE_DATA(
		#LIST_NOTIF_ALL)
			#R_LIST_NOTIF_IN1_PIR
			SW = 0x9000

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_REMOVE_NOTIF(
		<NOTIF_SEQ_NO_IN1_PIR> - 1))
			#R_REMOVE_NOTIF_NOTHING_TO_DELETE
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#LIST_NOTIF_ALL)
			#R_LIST_NOTIF_IN1_PIR
			SW = 0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_REMOVE_NOTIF(
		<NOTIF_SEQ_NO_IN1_PIR> + 1))
			#R_REMOVE_NOTIF_NOTHING_TO_DELETE
			SW = 0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(
		#LIST_NOTIF_ALL)
			#R_LIST_NOTIF_IN1_PIR
			SW = 0x9000

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_REMOVE_NOTIF(
		<NOTIF_SEQ_NO_IN1_PIR>))
			#R_REMOVE_NOTIF_OK
			SW = 0x9000

6	S_LPAd → eUICC	MTD_STORE_DATA(
		#LIST_NOTIF_ALL)
			#R_LIST_NOTIF_NONE
			SW = 0x9000

7	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_REMOVE_NOTIF(
		<NOTIF_SEQ_NO_IN1_PIR>))
			#R_REMOVE_NOTIF_NOTHING_TO_DELETE
			SW = 0x9000

8	S_LPAd → eUICC	MTD_STORE_DATA(
		#LIST_NOTIF_ALL)
			#R_LIST_NOTIF_NONE
			SW = 0x9000

9	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_REMOVE_NOTIF(
		<NOTIF_SEQ_NO_IN1_PIR> + 1))
			#R_REMOVE_NOTIF_NOTHING_TO_DELETE
			SW = 0x9000

10	S_LPAd → eUICC	MTD_STORE_DATA(
		#LIST_NOTIF_ALL)
			#R_LIST_NOTIF_NONE
			SW = 0x9000

4.2.17 ES10b (LPA -- eUICC): LoadCRL

This section is defined as FFS.

4.2.18 ES10b (LPA -- eUICC): AuthenticateServer

4.2.18.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 2.4.2
- Section 2.6.2, 2.6.7.1, 2.6.7.2
- Section 3.0.1, 3.1.3
- Section 3.6.2
- Section 4.5.2
- Section 5.5
- Section 5.6.1
- Section 5.7.13
- Section 5.8.1

4.2.18.2 Test Cases

4.2.18.2.1 TC_eUICC_ES10b.AuthenticateServer_SM-DP+_NIST_Server_Variant_O

Test Sequence #01 Nominal: Without MatchingID in CtxParams1

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000
			Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> and <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION> from response data and verify if they contain at least one same GSMA CI Key ID based on NIST P-256 curve

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED> to the CI Key ID based on NIST P-256 curve • Choose the #CERT_S_SM_DPauth_SIG leading to the same Root CI certificate		

4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(
		MTD_AUTHENTICATE_SMDP(
		#TEST_DP_ADDRESS1,
		<S_SMDP_CHALLENGE>,
		#CTX_PARAMS1,
		<S_SMDP_SIGNATURE1>,
		#CERT_S_SM_DPauth_SIG,
		NO_PARAM,
		#CRL_LIST,
		FALSE
		)
		)
			MTD_CHECK_AUTH_SERVER_RESP(
			#TEST_DP_ADDRESS1,
			<S_SMDP_CHALLENGE>,
			#CTX_PARAMS1)
			SW = 0x9000

Test Sequence #02 Nominal: With MatchingID in CtxParams1

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000
			Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> and <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION> from response data and verify if they contain at least one same GSMA CI Key ID based on NIST P-256 curve

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED> to the CI Key ID based on NIST P-256 curve • Choose the #CERT_S_SM_DPauth_SIG leading to the same Root CI certificate		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_MATCH_ID, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_SIG, NO_PARAM, #CRL_LIST, FALSE))	MTD_CHECK_AUTH_SERVER_RESP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_MATCH_ID) SW = 0x9000

Test Sequence #03 Nominal: With IMEI in Device Capabilities

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000
			Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> and <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION> from response data and verify if they contain at least one same GSMA CI Key ID based on NIST P-256 curve

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED> to the CI Key ID based on NIST P-256 curve • Choose the #CERT_S_SM_DPauth_SIG leading to the same Root CI certificate		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>,	MTD_CHECK_AUTH_SERVER_RESP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_IMEI

		#CTX_PARAMS1_IMEI, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_SIG, NO_PARAM, #CRL_LIST, FALSE))	) SW = 0x9000

4.2.18.2.2 TC_eUICC_ES10b.AuthenticateServer_SM-DP+_BRP_Server_Variant_O

Test Sequence #01 Nominal: Without MatchingID in CtxParams1

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000
			Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> and <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION> from response data and verify if they contain at least one same GSMA CI Key ID based on BrainpoolP256r1 curve

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED> to the CI Key ID based on BrainpoolP256r1 curve • Choose the #CERT_S_SM_DPauth_SIG leading to the same Root CI certificate		

4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(
		#TEST_DP_ADDRESS1,
		<S_SMDP_CHALLENGE>,
		#CTX_PARAMS1,
		<S_SMDP_SIGNATURE1>,
		#CERT_S_SM_DPauth_SIG,
		NO_PARAM,
		#CRL_LIST,
		FALSE
		)
		)
			MTD_CHECK_AUTH_SERVER_RESP(
			#TEST_DP_ADDRESS1,
			<S_SMDP_CHALLENGE>,
			#CTX_PARAMS1)
			SW = 0x9000

Test Sequence #02 Nominal: With MatchingID in CtxParams1

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000
			Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> and <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION> from response data and verify if they contain at least one same GSMA CI Key ID based on BrainpoolP256r1 curve

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED> to the CI Key ID based on BrainpoolP256r1 curve • Choose the #CERT_S_SM_DPauth_SIG leading to the same Root CI certificate		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_MATCH_ID, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_SIG, NO_PARAM, #CRL_LIST, FALSE))	MTD_CHECK_AUTH_SERVER_RESP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_MATCH_ID) SW = 0x9000

Test Sequence #03 Nominal: With IMEI in Device Capabilities

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000
			Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> and

			<EUICC_CI_PK_ID_LIST_FOR_VERIFICATION> from response data and verify if they contain at least one same GSMA CI Key ID based on BrainpoolP256r1 curve

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED> to the CI Key ID based on BrainpoolP256r1 curve • Choose the #CERT_S_SM_DPauth_SIG leading to the same Root CI certificate		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_IMEI, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_SIG, NO_PARAM, #CRL_LIST, FALSE))	MTD_CHECK_AUTH_SERVER_RESP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_IMEI) SW = 0x9000

4.2.18.2.3 TC_eUICC_ES10b.AuthenticateServer_SM-DP+_FRP

This test case is defined as FFS and not applicable for this version of test specification.

4.2.18.2.4 TC_eUICC_ES10b.AuthenticateServer_SM-DP+_ErrorCases_Server_Variant_O

Test Sequence #01 Error: With Incorrect SM-DPauth certificate (i.e. invalid signature)

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE>		

	• <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> • Choose the #CERT_S_SM_DPauth_INV_SIGN leading to the same Root CI certificate apart from the signature		

4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT (MTD_AUTHENTICATE_SMDP(
		#TEST_DP_ADDRESS1,
		<S_SMDP_CHALLENGE>,
		#CTX_PARAMS1,
		<S_SMDP_SIGNATURE1>,
		#CERT_S_SM_DPauth_INV_SIG,
		NO_PARAM,
		#CRL_LIST,
		FALSE
		)
		)
			#R_AUTH_SERVER_INV_CERT
			SW = 0x9000
			
			• Verify that the <S_TRANSACTION_ID> present in the `AuthenticateResponseError` is the same as <S_TRANSACTION_ID> in MTD_AUTHENTICATE_SMDP.

Test Sequence #02 Error: With Invalid SM-DP+ Signature

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> NOT computed with the #SK_S_SM_DPauth_SIG but SHALL have the same length as for a valid signature • Set the <EUICC_CI_PK_ID_TO_BE_USED> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> • Choose the #CERT_S_SM_DPauth_SIG leading to the same Root CI certificate		

4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(
		#TEST_DP_ADDRESS1,
		<S_SMDP_CHALLENGE>,
		#CTX_PARAMS1,
		<S_SMDP_SIGNATURE1>,
		#CERT_S_SM_DPauth_SIG,
		NO_PARAM,
		#CRL_LIST,
			#R_AUTH_SERVER_INV_SIGN
			SW = 0x9000
			
			• Verify that the <S_TRANSACTION_ID> present in the `AuthenticateResponseError` is the same as

		FALSE))	<S_TRANSACTION_ID> in MTD_AUTHENTICATE_SMDP

Test Sequence #03 Error: Unsupported Curve

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <RANDOM_SM_DP+_SIGN> • Set the <EUICC_CI_PK_ID_TO_BE_USED> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> • #CERT_S_SM_DPauth_INV_CURVE		

4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT
		(MTD_AUTHENTICATE_SMDP(
		#TEST_DP_ADDRESS1,
		<S_SMDP_CHALLENGE>,
		#CTX_PARAMS1,
		<S_SMDP_SIGNATURE1>,
		#CERT_S_SM_DPauth_INV_CURVE,
		NO_PARAM,
		#CRL_LIST,
		FALSE
		)
		)
			#R_AUTH_SERVER_INV_CURV
			SW = 0x9000
			
			• Verify that the <S_TRANSACTION_ID> present in the AuthenticateResponseError is the same as <S_TRANSACTION_ID> in MTD_AUTHENTICATE_SMDP.

Test Sequence #04 Error: eUICC Challenge Mismatch

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • Change the value of <EUICC_CHALLENGE> (retrieved in step 1) to a random value different from <EUICC_CHALLENGE>• <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> • Choose the #CERT_S_SM_DPauth_SIG leading to the same Root CI certificate		

4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT
		(MTD_AUTHENTICATE_SMDP(
		#TEST_DP_ADDRESS1,
		<S_SMDP_CHALLENGE>,
		#CTX_PARAMS1,
		<S_SMDP_SIGNATURE1>,
		#CERT_S_SM_DPauth_SIG,
		NO_PARAM,
		#CRL_LIST,
		FALSE
		)
		)
			#R_AUTH_SERVER_INV_CHALLENGE
			SW = 0x9000
			
			• Verify that the <S_TRANSACTION_ID> present in the AuthenticateResponseError is the same as <S_TRANSACTION_ID> in MTD_AUTHENTICATE_SMDP.

Test Sequence #05 Error: Unknown CI PK

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA
		(#GET_EUICC_INFO1)
			#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA
		(#GET_EUICC_CHALLENGE)
			#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED> to a CI Key ID not present in the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> (a random SubjectKeyIdentifier can be used) • Choose the #CERT_S_SM_DPauth_SIG leading to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION>		

4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT
		(MTD_AUTHENTICATE_SMDP(
		#TEST_DP_ADDRESS1,
		<S_SMDP_CHALLENGE>,
		#CTX_PARAMS1,
		<S_SMDP_SIGNATURE1>,
		#CERT_S_SM_DPauth_SIG,
			#R_AUTH_SERVER_INV_CI
			SW = 0x9000
			
			• Verify that the <S_TRANSACTION_ID> present in the AuthenticateResponseError is the same as <S_TRANSACTION_ID> in MTD_AUTHENTICATE_SMDP.

		NO_PARAM, #CRL_LIST, FALSE))	

Test Sequence #06 Error: Invalid Certificate Role OID

The purpose of this sequence is to make sure that the eUICC refuses any SM-DP+ Certificate for authentication that does not indicate “id-rspRole-dp-auth” in its extension for Certificate Policies.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA
		(#GET_EUICC_INFO1)
			#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA
		(#GET_EUICC_CHALLENGE)
			#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> • Choose the #CERT_S_SM_DPpb_SIG (instead of #CERT_S_SM_DPauth_SIG) leading to the same Root CI certificate		

4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT
		(MTD_AUTHENTICATE_SMDP(
		#TEST_DP_ADDRESS1,
		<S_SMDP_CHALLENGE>,
		#CTX_PARAMS1,
		<S_SMDP_SIGNATURE1>,
		#CERT_S_SM_DPpb_SIG,
		NO_PARAM,
		#CRL_LIST,
		FALSE
		)
		)
			#R_AUTH_SERVER_INV_OID
			SW = 0x9000
			OR
			#R_AUTH_SERVER_INV_CERT
			SW = 0x9000
			
			• Verify that the <S_TRANSACTION_ID> present in the AuthenticateResponseError is the same as <S_TRANSACTION_ID> in MTD_AUTHENTICATE_SMDP.

Test Sequence #07 Error: No RSP session on-going

Initial Conditions	
Entity	**Description of the initial state**
eUICC	No RSP session is on-going (i.e. no ES10b.getEUICCChallenge has been sent to the eUICC).

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

2	The following inputs are required for Step 3 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • Change the value of <EUICC_CHALLENGE> (retrieved in step 1) to a random value different from <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> • Choose the #CERT_S_SM_DPauth_SIG leading to the same Root CI certificate		

3	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT
		(MTD_AUTHENTICATE_SMDP(
		#TEST_DP_ADDRESS1,
		<S_SMDP_CHALLENGE>,
		#CTX_PARAMS1,
		<S_SMDP_SIGNATURE1>,
		#CERT_S_SM_DPauth_SIG,
		NO_PARAM,
		#CRL_LIST,
		FALSE
		)
		)
			#R_AUTH_SERVER_NO_SESSION
			SW = 0x9000
			The transactionId returned in the response SHALL not be checked (any value SHALL be accepted)

4.2.18.2.5 TC_eUICC_ES10b.AuthenticateServer_SM-DS_BRP_Server_Variant_O

Test Sequence #01 Nominal: With EventID in CtxParams1

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000
			Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> and <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION>

			from response data and verify if they contain at least one same GSMA CI Key ID based on BrainpoolP256r1 curve

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDS_CHALLENGE> • <S_SMDS_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED> to the CI Key ID based on BrainpoolP256r1 curve • Choose the #CERT_S_SM_DSauth_SIG leading to the same Root CI certificate		

4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(
		
		MTD_AUTHENTICATE_SMDS(
		#TEST_ROOT_DS_ADDRESS,
		<S_SMDS_CHALLENGE>,
		#CTX_PARAMS1_EVENT_ID,
		<S_SMDS_SIGNATURE1>,
		#CERT_S_SM_DSauth_SIG,
		NO_PARAM,
		#CRL_LIST,
		FALSE
		)
		)
			MTD_CHECK_AUTH_SERVER_RESP(
			#TEST_ROOT_DS_ADDRESS,
			<S_SMDS_CHALLENGE>,
			#CTX_PARAMS1_EVENT_ID)
			SW = 0x9000

Test Sequence #02 Nominal: With IMEI and EventID in Device Capabilities

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000
			Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> and <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION> from response data and verify if they contain at least one same GSMA CI Key ID based on BrainpoolP256r1 curve

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDS_CHALLENGE>		

	• <S_SMDS_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED> to the CI Key ID based on BrainpoolP256r1 curve • Choose the #CERT_S_SM_DSauth_SIG leading to the same Root CI certificate		

4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(
		MTD_AUTHENTICATE_SMDS(
		#TEST_ROOT_DS_ADDRESS,
		<S_SMDS_CHALLENGE>,
		#CTX_PARAMS1_EVENT_ID_IMEI,
		<S_SMDS_SIGNATURE1>,
		#CERT_S_SM_DSauth_SIG,
		NO_PARAM,
		#CRL_LIST,
		FALSE
		)
		)
			MTD_CHECK_AUTH_SERVER_RESP(
			#TEST_ROOT_DS_ADDRESS,
			<S_SMDS_CHALLENGE>,
			#CTX_PARAMS1_EVENT_ID_IMEI)
			SW = 0x9000

4.2.18.2.6 TC_eUICC_ES10b.AuthenticateServer_SM-DS_NIST_Server_Variant_O

Test Sequence #01 Nominal: With EventID in CtxParams1

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000
			Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> and <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION> from response data and verify if they contain at least one same GSMA CI Key ID based on NIST P-256 curve

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDS_CHALLENGE> • <S_SMDS_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED> to the CI Key ID based on NIST P-256 curve • Choose the #CERT_S_SM_DSauth_SIG leading to the same Root CI certificate		

4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(
		MTD_AUTHENTICATE_SMDS(
		#TEST_ROOT_DS_ADDRESS,
		<S_SMDS_CHALLENGE>,
		#CTX_PARAMS1_EVENT_ID,
			MTD_CHECK_AUTH_SERVER_RESP(
			#TEST_ROOT_DS_ADDRESS,
			<S_SMDS_CHALLENGE>,
			#CTX_PARAMS1_EVENT_ID)
			SW = 0x9000

		<S_SMDS_SIGNATURE1>, #CERT_S_SM_DSauth_SIG, NO_PARAM, #CRL_LIST, FALSE))	

Test Sequence #02 Nominal: With IMEI and EventID in Device Capabilities

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000
			Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> and <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION> from response data and verify if they contain at least one same GSMA CI Key ID based on NIST P-256 curve

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDS_CHALLENGE> • <S_SMDS_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED> to the CI Key ID based on NIST P-256 curve • Choose the #CERT_S_SM_DSauth_SIG leading to the same Root CI certificate		

4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDS(
		#TEST_ROOT_DS_ADDRESS,
		<S_SMDS_CHALLENGE>,
		#CTX_PARAMS1_EVENT_ID_IMEI,
		<S_SMDS_SIGNATURE1>,
		#CERT_S_SM_DSauth_SIG,
		NO_PARAM,
		#CRL_LIST,
		FALSE
		)
		)
			MTD_CHECK_AUTH_SERVER_RESP(
			#TEST_ROOT_DS_ADDRESS,
			<S_SMDS_CHALLENGE>,
			#CTX_PARAMS1_EVENT_ID_IMEI)
			SW = 0x9000

4.2.18.2.7 TC_eUICC_ES10b.AuthenticateServer_SM-DS_FRP

This test case is defined as FFS and not applicable for this version of test specification.

4.2.18.2.8 TC_eUICC_ES10b.AuthenticateServer_SM-DS_ErrorCases_Server_Variant_O

Test Sequence #01 Error: With Incorrect SM-DSauth certificate (i.e. invalid signature)

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDS_CHALLENGE> • <S_SMDS_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> • Choose the #CERT_S_SM_DSauth_INV_SIGN leading to the same Root CI certificate		

4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(
		MTD_AUTHENTICATE_SMDS(
		#TEST_ROOT_DS_ADDRESS,
		<S_SMDS_CHALLENGE>,
		#CTX_PARAMS1_EVENT_ID,
		<S_SMDS_SIGNATURE1>,
		#CERT_S_SM_DSauth_INV_SIG,
		NO_PARAM,
		#CRL_LIST,
		FALSE
		)
		)
			#R_AUTH_SERVER_INV_CERT
			SW = 0x9000
			
			• Verify that the <S_TRANSACTION_ID> present in the `AuthenticateResponseError` is the same as <S_TRANSACTION_ID> in MTD_AUTHENTICATE_SMDS.

Test Sequence #02 Error: With Invalid SM-DS Signature

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID>		

	• <EUICC_CHALLENGE> • <S_SMDS_CHALLENGE> • <S_SMDS_SIGNATURE1> NOT computed with the #SK_S_SM_DSauth_SIG but SHALL have the same length as for a valid signature • Set the <EUICC_CI_PK_ID_TO_BE_USED> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> • Choose the #CERT_S_SM_DSauth_SIG leading to the same Root CI certificate		

4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(
		MTD_AUTHENTICATE_SMDS(
		#TEST_ROOT_DS_ADDRESS,
		<S_SMDS_CHALLENGE>,
		#CTX_PARAMS1_EVENT_ID,
		<S_SMDS_SIGNATURE1>,
		#CERT_S_SM_DSauth_SIG,
		NO_PARAM,
		#CRL_LIST,
		FALSE
		)
		)
			#R_AUTH_SERVER_INV_SIGN
			SW = 0x9000
			
			• Verify that the <S_TRANSACTION_ID> present in the `AuthenticateResponseError` is the same as <S_TRANSACTION_ID> in MTD_AUTHENTICATE_SMDS

Test Sequence #03 Error: Unsupported Curve

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDS_CHALLENGE> • <RANDOM_SM_DS_SIGN> • Set the <EUICC_CI_PK_ID_TO_BE_USED> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> • #CERT_S_SM_DSauth_INV_CURVE		

4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT (MTD_AUTHENTICATE_SMDS(
		#TEST_ROOT_DS_ADDRESS,
		<S_SMDS_CHALLENGE>,
		#CTX_PARAMS1_EVENT_ID,
		<S_SMDS_SIGNATURE1>,
		#CERT_S_SM_DSauth_INV_CURVE,
		NO_PARAM,
			#R_AUTH_SERVER_INV_CURV
			SW = 0x9000
			
			• Verify that the <S_TRANSACTION_ID> present in the `AuthenticateResponseError` is the same as <S_TRANSACTION_ID> in MTD_AUTHENTICATE_SMDS.

		#CRL_LIST, FALSE))	

Test Sequence #04 Error: eUICC Challenge Mismatch

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA
		(#GET_EUICC_INFO1)
			#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA
		(#GET_EUICC_CHALLENGE)
			#R_CHALLENGE
			SW = 0x9000

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • Change the value of <EUICC_CHALLENGE> (retrieved in step 1) to a random value different from <EUICC_CHALLENGE> • <S_SMDS_CHALLENGE> • <S_SMDS_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> • Choose the #CERT_S_SM_DSauth_SIG leading to the same Root CI certificate		

4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT (
		MTD_AUTHENTICATE_SMDS(
		#TEST_ROOT_DS_ADDRESS,
		<S_SMDS_CHALLENGE>,
		#CTX_PARAMS1_EVENT_ID,
		<S_SMDS_SIGNATURE1>,
		#CERT_S_SM_DSauth_SIG,
		NO_PARAM,
		#CRL_LIST,
		FALSE
		)
		)
			#R_AUTH_SERVER_INV_CHALLENGE
			SW = 0x9000
			
			• Verify that the <S_TRANSACTION_ID> present in the `AuthenticateResponseError` is the same as <S_TRANSACTION_ID> in MTD_AUTHENTICATE_SMDS .

Test Sequence #05 Error: Unknown CI PK

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA
		(#GET_EUICC_INFO1)
			#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA
		(#GET_EUICC_CHALLENGE)
			#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function:		

	• <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDS_CHALLENGE> • <S_SMDS_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED> to a CI Key ID not present in the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> (a random SubjectKeyIdentifier can be used) • Choose the #CERT_S_SM_DSauth_SIG leading to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION>		

4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT
		(MTD_AUTHENTICATE_SMDS(
		#TEST_ROOT_DS_ADDRESS,
		<S_SMDS_CHALLENGE>,
		#CTX_PARAMS1_EVENT_ID,
		<S_SMDS_SIGNATURE1>,
		#CERT_S_SM_DSauth_SIG,
		NO_PARAM,
		#CRL_LIST,
		FALSE
		)
		)
			#R_AUTH_SERVER_INV_CI
			SW = 0x9000
			
			• Verify that the <S_TRANSACTION_ID> present in the `AuthenticateResponseError` is the same as <S_TRANSACTION_ID> in MTD_AUTHENTICATE_SMDS .

Test Sequence #06 Error: No RSP session on-going

Initial Conditions	
Entity	**Description of the initial state**
eUICC	No RSP session is on-going (i.e. no ES10b.getEUICCChallenge has been sent to the eUICC).

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA
		(#GET_EUICC_INFO1)
			#R_EUICC_INFO1
			SW = 0x9000

2	The following inputs are required for Step 3 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • Change the value of <EUICC_CHALLENGE> (retrieved in step 1) to a random value different from <EUICC_CHALLENGE> • <S_SMDS_CHALLENGE> • <S_SMDS_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> • Choose the #CERT_S_SM_DSauth_SIG leading to the same Root CI certificate		

3	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(
		MTD_AUTHENTICATE_SMDS(
		#TEST_ROOT_DS_ADDRESS,
		<S_SMDS_CHALLENGE>,
		#CTX_PARAMS1_EVENT_ID,
			#R_AUTH_SERVER_NO_SESSION
			SW = 0x9000
			The transactionId returned in the response SHALL not be checked (any value SHALL be accepted)

		<S_SMDS_SIGNATURE1>, #CERT_S_SM_DSauth_SIG, NO_PARAM, #CRL_LIST, FALSE))	

4.2.18.2.9 TC_eUICC_ES10b.AuthenticateServer_SM-DP+_NIST_Server_Variants_V3

Test Sequence #01 Nominal: Server Variant A without MatchingID in CtxParams1

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000
			Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> and <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION> from response data and verify if they contain at least one same GSMA CI Key ID based on NIST P-256 curve

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID based on NIST P-256 curve • Choose the #CERT_S_SM_DPauth_SIG and the remaining part of the Variant A Certificates chain leading to the same Root CI certificate as the one chosen for signing		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMPD (#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_SIG, #CERT_S_SM_DP_SubCA_SIG, #CRL_LIST, TRUE))	MTD_CHECK_AUTH_SERVER_RESP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1) SW = 0x9000

Test Sequence #02 Nominal: Server Variant A with MatchingID in CtxParams1

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000
			Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> and <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION> from response data and verify if they contain at least one same GSMA CI Key ID based on NIST P-256 curve

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID based on NIST P-256 curve • Choose the #CERT_S_SM_DPauth_SIG and the remaining part of the Variant A Certificates chain leading to the same Root CI certificate as the one chosen for signing		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_MATCH_ID, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_SIG, #CERT_S_SM_DP_SubCA_SIG, #CRL_LIST, TRUE))	MTD_CHECK_AUTH_SERVER_RESP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_MATCH_ID) SW = 0x9000

Test Sequence #03 Nominal: Server Variant A with IMEI in Device Capabilities

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000
			Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> and <EUICC_CI_PK_ID_LIST_FOR_VERIFIC

			ATION> from response data and verify if they contain at least one same GSMA CI Key ID based on NIST P-256 curve

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID based on NIST P-256 curve • Choose the #CERT_S_SM_DPauth_SIG and the remaining part of the Variant A Certificates chain leading to the same Root CI certificate as the one chosen for signing		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_IMEI, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_SIG, #CERT_S_SM_DP_SubCA_SIG, #CRL_LIST, TRUE))	MTD_CHECK_AUTH_SERVER_RESP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_IMEI) SW = 0x9000

Test Sequence #04 Nominal: Server Variant B without MatchingID in CtxParams1

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000
			Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> and <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION> from response data and verify if they contain at least one same GSMA CI Key ID based on NIST P-256 curve

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1>		

	• Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID based on NIST P-256 curve • Choose the #CERT_S_SM_DPauth_SIG and the remaining part of the Variant B Certificates chain leading to the same Root CI certificate as the one chosen for signing		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_SIG, #CERT_CI_SubCA_SIG, #CRL_LIST, TRUE))	MTD_CHECK_AUTH_SERVER_RESP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1) SW = 0x9000

Test Sequence #05 Nominal: Server Variant B with MatchingID in CtxParams1

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000
			Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> and <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION> from response data and verify if they contain at least one same GSMA CI Key ID based on NIST P-256 curve

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: <S_TRANSACTION_ID> <EUICC_CHALLENGE> <S_SMDP_CHALLENGE> <S_SMDP_SIGNATURE1> Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID based on NIST P-256 curve Choose the #CERT_S_SM_DPauth_SIG and the remaining part of the Variant B Certificates chain leading to the same Root CI certificate as the one chosen for signing		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_MATCH_ID, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_SIG, #CERT_CI_SubCA_SIG,	MTD_CHECK_AUTH_SERVER_RESP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_MATCH_ID) SW = 0x9000

		#CRL_LIST, TRUE))	

Test Sequence #06 Nominal: Server Variant B with IMEI in Device Capabilities

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000
			Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> and <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION> from response data and verify if they contain at least one same GSMA CI Key ID based on NIST P-256 curve

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: <S_TRANSACTION_ID> <EUICC_CHALLENGE> <S_SMDP_CHALLENGE> <S_SMDP_SIGNATURE1> Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID based on NIST P-256 curve Choose the #CERT_S_SM_DPauth_SIG and the remaining part of the Variant B Certificates chain leading to the same Root CI certificate as the one chosen for signing		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_IMEI, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_SIG, #CERT_CI_SubCA_SIG, #CRL_LIST, TRUE))	MTD_CHECK_AUTH_SERVER_RESP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_IMEI) SW = 0x9000

Test Sequence #07 Nominal: Server Variant C without MatchingID in CtxParams1

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		

IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000
			Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> and <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION> from response data and verify if they contain at least one same GSMA CI Key ID based on NIST P-256 curve

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID based on NIST P-256 curve • Choose the #CERT_S_SM_DPauth_SIG and the remaining part of the Variant C Certificates chain leading to the same Root CI certificate as the one chosen for signing		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_SIG, #CERT_S_SM_DP_SubCAList_SIG, #CRL_LIST, TRUE))	MTD_CHECK_AUTH_SERVER_RESP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1) SW = 0x9000

Test Sequence #08 Nominal: Server Variant C with MatchingID in CtxParams1

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000
			Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> and <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION> from response data and verify if they contain at least one same GSMA CI Key ID based on NIST P-256 curve

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: <S_TRANSACTION_ID> <EUICC_CHALLENGE> <S_SMDP_CHALLENGE> <S_SMDP_SIGNATURE1> Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID based on NIST P-256 curve Choose the #CERT_S_SM_DPauth_SIG and the remaining part of the Variant C Certificates chain leading to the same Root CI certificate as the one chosen for signing		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_MATCH_ID, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_SIG, #CERT_S_SM_DP_SubCAList_SIG, #CRL_LIST, TRUE))	MTD_CHECK_AUTH_SERVER_RESP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_MATCH_ID) SW = 0x9000

Test Sequence #09 Nominal: Server Variant C with IMEI in Device Capabilities

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000
			Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> and <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION> from response data and verify if they contain at least one same GSMA CI Key ID based on NIST P-256 curve

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: <S_TRANSACTION_ID> <EUICC_CHALLENGE> <S_SMDP_CHALLENGE> <S_SMDP_SIGNATURE1> Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID based on NIST P-256 curve		

	Choose the #CERT_S_SM_DPauth_SIG and the remaining part of the Variant C Certificates chain leading to the same Root CI certificate as the one chosen for signing		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_IMEI, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_SIG, #CERT_S_SM_DP_SubCAList_SIG, #CRL_LIST, TRUE))	MTD_CHECK_AUTH_SERVER_RESP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_IMEI) SW = 0x9000

Test Sequence #10 Nominal: Server Variant A without MatchingID and OperationType RPM in CtxParams1

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000
			Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> and <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION> from response data and verify if they contain at least one same GSMA CI Key ID based on NIST P-256 curve

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID based on NIST P-256 curve • Choose the #CERT_S_SM_DPauth_SIG and the remaining part of the Variant A Certificates chain leading to the same Root CI certificate as the one chosen for signing		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_RPM_ICCID1, <S_SMDP_SIGNATURE1>,	MTD_CHECK_AUTH_SERVER_RESP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_RPM_ICCID1) SW = 0x9000

		#CERT_S_SM_DPauth_SIG, #CERT_S_SM_DP_SubCA_ECDSA, #CRL_LIST, TRUE))	

4.2.18.2.10 TC_eUICC_ES10b.AuthenticateServer_SM-DP+_ErrorCases_V3_Server_Variant A

In order to execute the error cases defined in this section, the variant A of the SM-DP+ Certificates chain is used by default.

Test Sequence #01 Error: With Incorrect SM-DPauth certificate (i.e. invalid signature)

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> • Choose the #CERT_S_SM_DPauth_INV_SIGN and the remaining part of the Variant A Certificates chain leading to the same Root CI certificate as the one chosen for signing		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_INV_SIGN, #CERT_S_SM_DP_SubCA_SIG, #CRL_LIST, TRUE))	MTD_CHECK_AUTH_ERROR_RESP(`invalidCertificate`) SW = 0x9000

Test Sequence #02 Error: With Invalid SM-DP+ Signature

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> NOT computed with the #SK_S_SM_DPauth_SIG but SHALL have the same length as for a valid signature • Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> • Choose the #CERT_S_SM_DPauth_SIG and the remaining part of the Variant A Certificates chain leading to the same Root CI certificate as the one chosen for signing		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_SIG, #CERT_S_SM_DP_SubCA_SIG, #CRL_LIST, TRUE))	MTD_CHECK_AUTH_ERROR_RESP(`invalidSignature`) SW = 0x9000

Test Sequence #03 Error: Unsupported Curve

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1>		

	• Set the < EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> • Use #CERT_S_SM_DPauth_INV_CURVE and the remaining part of the Variant A Certificates chain leading to the CI Key ID set in highest priority in the <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION>		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_INV_CURVE, #CERT_S_SM_DP_SubCA_SIG, #CRL_LIST, TRUE))	MTD_CHECK_AUTH_ERROR_RESP(`unsupportedCurve`) SW = 0x9000

Test Sequence #04 Error: eUICC Challenge Mismatch

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • Change the value of <EUICC_CHALLENGE> (retrieved in step 1) to a random value different from <EUICC_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> • Choose the #CERT_S_SM_DPauth_SIG and the remaining part of the Variant A Certificates chain leading to the same Root CI certificate as the one chosen for signing		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_SIG, #CERT_S_SM_DP_SubCA_SIG, #CRL_LIST, TRUE))	MTD_CHECK_AUTH_ERROR_RESP(`euiccChallengeMismatch`) SW = 0x9000

Test Sequence #05 Error: Unknown CI PK

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to a CI Key ID not present in the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> (a random SubjectKeyIdentifier can be used) • Use #CERT_S_SM_DPauth_INV_SIG and the remaining part of the Variant A Certificates chain leading to the CI Key ID set in highest priority in the <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION>		

4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT
		(MTD_AUTHENTICATE_SMDP(
		#TEST_DP_ADDRESS1,
		<S_SMDP_CHALLENGE>,
		#CTX_PARAMS1,
		<S_SMDP_SIGNATURE1>,
		#CERT_S_SM_DPauth_SIG,
		#CERT_S_SM_DP_SubCA_SIG,
		#CRL_LIST,
		TRUE
		)
		)
			#R_AUTH_SERVER_INV_CI
			SW = 0x9000
			
			• Verify that the <S_TRANSACTION_ID> present in the AuthenticateResponseError is the same as <S_TRANSACTION_ID> in MTD_AUTHENTICATE_SMDP.

Test Sequence #06 Error: Invalid Certificate Role OID

The purpose of this sequence is to make sure that the eUICC refuses any SM-DP+ Certificate for authentication that does not indicate “id-rspRole-dp-auth” in its extension for Certificate Policies.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		

IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> • Choose the #CERT_S_SM_DPpb_SIG (instead of #CERT_S_SM_DPauth_SIG) and the remaining part of the Variant A Certificates chain leading to the same Root CI certificate as the one chosen for signing		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPpb_SIG, #CERT_S_SM_DP_SubCA_SIG, #CRL_LIST, TRUE))	MTD_CHECK_AUTH_ERROR_RESP(`invalidOid`) SW = 0x9000

Test Sequence #07 Error: No RSP session on-going

Initial Conditions	
Entity	**Description of the initial state**
eUICC	No RSP session is on-going (i.e. no ES10b.getEUICCChallenge has been sent to the eUICC).

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

2	The following inputs are required for Step 3 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • Set <EUICC_CHALLENGE> to a random value • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1>		

	• Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> • Choose the #CERT_S_SM_DPauth_SIG and the remaining part of the Variant A Certificates chain leading to the same Root CI certificate as the one chosen for signing		

3	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(
		MTD_AUTHENTICATE_SMDP(
		#TEST_DP_ADDRESS1,
		<S_SMDP_CHALLENGE>,
		#CTX_PARAMS1,
		<S_SMDP_SIGNATURE1>,
		#CERT_S_SM_DPauth_SIG,
		#CERT_S_SM_DP_SubCA_SIG,
		#CRL_LIST,
		TRUE
		)
		)
			#R_AUTH_SERVER_NO_SESSION
			SW = 0x9000
			The transactionId returned in the response SHALL not be checked (any value SHALL be accepted)

Test Sequence #08 Error: With Incorrect SM-DPauth certificate – Variant A (i.e. invalid signature) – CtxParam1 with OperationType RPM

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> • Choose the #CERT_S_SM_DPauth_INV_SIGN and the remaining part of the Variant A Certificates chain leading to the same Root CI certificate as the one chosen for signing		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_RPM_ICCID1, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_INV_SIGN, #CERT_S_SM_DP_SubCA_SIG, #CRL_LIST,	MTD_CHECK_AUTH_ERROR_RESP(`invalidCertificate`) SW = 0x9000

		TRUE))	

4.2.18.2.11 TC_eUICC_ES10b.AuthenticateServer_SM-DP+_ErrorCases_V3_Server_Variant_B

In order to execute the error cases defined in this section, the variant B of the SM-DP+ Certificates chain is used by default.

Test Sequence #01 Error: With Incorrect SM-DPauth certificate (i.e. invalid signature)

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: <S_TRANSACTION_ID> <EUICC_CHALLENGE> <S_SMDP_CHALLENGE> <S_SMDP_SIGNATURE1> Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> Choose the #CERT_S_SM_DPauth_INV_SIGN and the remaining part of the Variant B Certificates chain leading to the same Root CI certificate as the one chosen for signing		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_INV_SIGN, #CERT_CI_SubCA_SIG, #CRL_LIST, TRUE))	MTD_CHECK_AUTH_ERROR_RESP(invalidCertificate) SW = 0x9000

Test Sequence #02 Error: With Invalid SM-DP+ Signature

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: <S_TRANSACTION_ID> <EUICC_CHALLENGE> <S_SMDP_CHALLENGE> <S_SMDP_SIGNATURE1> NOT computed with the #SK_S_SM_DPauth_SIG but SHALL have the same length as for a valid signature Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> Choose the #CERT_S_SM_DPauth_SIG and the remaining part of the Variant B Certificates chain leading to the same Root CI certificate as the one chosen for signing		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_SIG, #CERT_CI_SubCA_SIG, #CRL_LIST, TRUE))	MTD_CHECK_AUTH_ERROR_RESP(`invalidSignature`) SW = 0x9000

Test Sequence #03 Error: Unsupported Curve

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: <S_TRANSACTION_ID> <EUICC_CHALLENGE> <S_SMDP_CHALLENGE> <S_SMDP_SIGNATURE1> Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> Choose the #CERT_S_SM_DPauth_INV_CURVE and the remaining part of the Variant B Certificates chain leading to the CI Key ID set in highest priority in the <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION>		

4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_INV_CURVE, #CERT_CI_SubCA_SIG, #CRL_LIST, TRUE))	MTD_CHECK_AUTH_ERROR_RESP(`unsupportedCurve`) SW = 0x9000

Test Sequence #04 Error: eUICC Challenge Mismatch

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA
		(#GET_EUICC_INFO1)
			#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA
		(#GET_EUICC_CHALLENGE)
			#R_CHALLENGE
			SW = 0x9000

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: <S_TRANSACTION_ID> Change the value of <EUICC_CHALLENGE> (retrieved in step 1) to a random value different from <EUICC_CHALLENGE> <S_SMDP_SIGNATURE1> Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> Choose the #CERT_S_SM_DPauth_SIG and the remaining part of the Variant B Certificates chain leading to the same Root CI certificate as the one chosen for signing		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_SIG, #CERT_CI_SubCA_SIG, #CRL_LIST, TRUE))	MTD_CHECK_AUTH_ERROR_RESP(`euiccChallengeMismatch`) SW = 0x9000

Test Sequence #05 Error: Unknown CI PK

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to a CI Key ID not present in the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> (a random SubjectKeyIdentifier can be used) • Use #CERT_S_SM_DPauth_SIG and the remaining part of the Variant B Certificates chain leading to the CI Key ID set in highest priority in the <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION>		

4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT
		(MTD_AUTHENTICATE_SMDP(
		#TEST_DP_ADDRESS1,
		<S_SMDP_CHALLENGE>,
		#CTX_PARAMS1,
		<S_SMDP_SIGNATURE1>,
		#CERT_S_SM_DPauth_SIG,
		#CERT_CI_SubCA_SIG,
		#CRL_LIST,
		TRUE
		)
		)
			#R_AUTH_SERVER_INV_CI
			SW = 0x9000
			
			• Verify that the <S_TRANSACTION_ID> present in the AuthenticateResponseError is the same as <S_TRANSACTION_ID> in MTD_AUTHENTICATE_SMDP.

Test Sequence #06 Error: Invalid Certificate Role OID

The purpose of this sequence is to make sure that the eUICC refuses any SM-DP+ Certificate for authentication that does not indicate “id-rspRole-dp-auth” in its extension for Certificate Policies.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA
		(#GET_EUICC_CHALLENGE)
			#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: <S_TRANSACTION_ID> <EUICC_CHALLENGE> <S_SMDP_CHALLENGE> <S_SMDP_SIGNATURE1> Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> Choose the #CERT_S_SM_DPpb_SIG (instead of #CERT_S_SM_DPauth_SIG) and the remaining part of the Variant B Certificates chain leading to the same Root CI certificate as the one chosen for signing		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPpb_SIG, #CERT_CI_SubCA_SIG, #CRL_LIST, TRUE))	MTD_CHECK_AUTH_ERROR_RESP(`invalidOid`) SW = 0x9000

Test Sequence #07 Error: No RSP session on-going

Initial Conditions	
Entity	**Description of the initial state**
eUICC	No RSP session is on-going (i.e. no ES10b.getEUICCChallenge has been sent to the eUICC).

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA
		(#GET_EUICC_INFO1)
			#R_EUICC_INFO1
			SW = 0x9000

2	The following inputs are required for Step 3 as described in the InitiateAuthentication function: <S_TRANSACTION_ID> Set <EUICC_CHALLENGE> to a random value <S_SMDP_CHALLENGE> <S_SMDP_SIGNATURE1> Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> Choose the #CERT_S_SM_DPauth_SIG and the remaining part of the Variant B Certificates chain leading to the same Root CI certificate as the one chosen for signing		

3	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(
		MTD_AUTHENTICATE_SMDP(
		#TEST_DP_ADDRESS1,
		<S_SMDP_CHALLENGE>,
		#CTX_PARAMS1,
		<S_SMDP_SIGNATURE1>,
		#CERT_S_SM_DPauth_SIG,
		#CERT_CI_SubCA_SIG,
		#CRL_LIST,
		TRUE)
		)
			#R_AUTH_SERVER_NO_SESSION
			SW = 0x9000
			The transactionId returned in the response SHALL not be checked (any value SHALL be accepted)

Test Sequence #08 Error: With Incorrect SM-DPauth certificate – (i.e. invalid signature) – CtxParam1 with OperationType RPM

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> • Choose the #CERT_S_SM_DPauth_INV_SIGN and the remaining part of the Variant B Certificates chain leading to the same Root CI certificate as the one chosen for signing		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_RPM_ICCID1, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_INV_SIGN, #CERT_CI_SubCA_SIG, #CRL_LIST, TRUE))	MTD_CHECK_AUTH_ERROR_RESP(`invalidCertificate`) SW = 0x9000

4.2.18.2.12 TC_eUICC_ES10b.AuthenticateServer_SM-DP+_ErrorCases_V3_Server_Variant_C

In order to execute the error cases defined in this section, the variant C of the SM-DP+ Certificates chain is used by default.

Test Sequence #01 Error: With Incorrect SM-DPauth certificate (i.e. invalid signature)

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: <S_TRANSACTION_ID> <EUICC_CHALLENGE> <S_SMDP_CHALLENGE> <S_SMDP_SIGNATURE1> Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> Choose the #CERT_S_SM_DPauth_INV_SIGN and the remaining part of the Variant C Certificates chain leading to the same Root CI certificate as the one chosen for signing		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_INV_SIGN, #CERT_S_SM_DP_SubCAList_SIG, #CRL_LIST, TRUE))	MTD_CHECK_AUTH_ERROR_RESP(`invalidCertificate`) SW = 0x9000

Test Sequence #02 Error: With Invalid SM-DP+ Signature

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function:		

	<S_TRANSACTION_ID> <EUICC_CHALLENGE> <S_SMDP_CHALLENGE> <S_SMDP_SIGNATURE1> NOT computed with the #SK_S_SM_DPauth_SIG but SHALL have the same length as for a valid signature Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> Choose the #CERT_S_SM_DPauth_SIG and the remaining part of the Variant C Certificates chain leading to the same Root CI certificate as the one chosen for signing		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_SIG, #CERT_S_SM_DP_SubCAList_SIG, #CRL_LIST, TRUE))	MTD_CHECK_AUTH_ERROR_RESP(`invalidSignature`) SW = 0x9000

Test Sequence #03 Error: Unsupported Curve

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA
		(#GET_EUICC_INFO1)
			#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA
		(#GET_EUICC_CHALLENGE)
			#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: <S_TRANSACTION_ID> <EUICC_CHALLENGE> <S_SMDP_CHALLENGE> <S_SMDP_SIGNATURE1> Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> Choose the #CERT_S_SM_DPauth_INV_CURVE and the remaining part of the Variant C Certificates chain leading to the CI Key ID set in highest priority in the <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION>		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1, <S_SMDP_SIGNATURE1>, #CERT_S_SM_Dpauth_INV_CURVE,	MTD_CHECK_AUTH_ERROR_RESP(`unsupportedCurve`) SW = 0x9000

		#CERT_S_SM_DP_SubCAList_SIG, #CRL_LIST, TRUE))	

Test Sequence #04 Error: eUICC Challenge Mismatch

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA
		(#GET_EUICC_INFO1)
			#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA
		(#GET_EUICC_CHALLENGE)
			#R_CHALLENGE
			SW = 0x9000

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: <S_TRANSACTION_ID> Change the value of <EUICC_CHALLENGE> (retrieved in step 1) to a random value different from <EUICC_CHALLENGE> <S_SMDP_SIGNATURE1> Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> Choose the #CERT_S_SM_DPauth_SIG and the remaining part of the Variant C Certificates chain leading to the same Root CI certificate as the one chosen for signing		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1, <S_SMDP_SIGNATURE1>, #CERT_S_SM_Dpauth_SIG, #CERT_S_SM_DP_SubCAList_SIG, #CRL_LIST, TRUE))	MTD_CHECK_AUTH_ERROR_RESP(`euiccChallengeMismatch`) SW = 0x9000

Test Sequence #05 Error: Unknown CI PK

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA
		(#GET_EUICC_INFO1)
			#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA
		(#GET_EUICC_CHALLENGE)
			#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to a CI Key ID not present in the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> (a random SubjectKeyIdentifier can be used) • Use #CERT_S_SM_DPauth_SIG and the remaining part of the Variant C Certificates chain leading to the CI Key ID set in highest priority in the <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION>		

4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT
		(MTD_AUTHENTICATE_SMDP(
		#TEST_DP_ADDRESS1,
		<S_SMDP_CHALLENGE>,
		#CTX_PARAMS1,
		<S_SMDP_SIGNATURE1>,
		#CERT_S_SM_DPauth_SIG,
		
		#CERT_S_SM_DP_SubCAList_SIG
		,
		#CRL_LIST,
		TRUE
		)
		)
			#R_AUTH_SERVER_INV_CI
			SW = 0x9000
			
			• Verify that the <S_TRANSACTION_ID> present in the AuthenticateResponseError is the same as <S_TRANSACTION_ID> in MTD_AUTHENTICATE_SMDP.

Test Sequence #06 Error: Invalid Certificate Role OID

The purpose of this sequence is to make sure that the eUICC refuses any SM-DP+ Certificate for authentication that does not indicate “id-rspRole-dp-auth” in its extension for Certificate Policies.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: <S_TRANSACTION_ID> <EUICC_CHALLENGE> <S_SMDP_CHALLENGE> <S_SMDP_SIGNATURE1> Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3>		

	Choose the #CERT_S_SM_DPpb_SIG (instead of #CERT_S_SM_DPauth_SIG) and the remaining part of the Variant C Certificates chain leading to the same Root CI certificate as the one chosen for signing		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPpb_SIG, #CERT_S_SM_DP_SubCAList_SIG, #CRL_LIST, TRUE))	MTD_CHECK_AUTH_ERROR_RESP(invalidOid) SW = 0x9000

Test Sequence #07 Error: No RSP session on-going

Initial Conditions	
Entity	**Description of the initial state**
eUICC	No RSP session is on-going (i.e. no ES10b.getEUICCChallenge has been sent to the eUICC).

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

2	The following inputs are required for Step 3 as described in the InitiateAuthentication function: <S_TRANSACTION_ID> Set <EUICC_CHALLENGE> to a random value <S_SMDP_CHALLENGE> <S_SMDP_SIGNATURE1> Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> Choose the #CERT_S_SM_DPauth_SIG and the remaining part of the Variant C Certificates chain leading to the same Root CI certificate as the one chosen for signing		

3	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(
		MTD_AUTHENTICATE_SMDP(
		#TEST_DP_ADDRESS1,
		<S_SMDP_CHALLENGE>,
		#CTX_PARAMS1,
		<S_SMDP_SIGNATURE1>,
		#CERT_S_SM_DPauth_SIG,
		#CERT_S_SM_DP_SubCAList_SIG,
		#CRL_LIST,
		TRUE)
		)
			#R_AUTH_SERVER_NO_SESSION
			SW = 0x9000
			The transactionId returned in the response SHALL not be checked (any value SHALL be accepted)

Test Sequence #08 Error: With Incorrect SM-DPauth certificate – (i.e. invalid signature) – CtxParam1 with OperationType RPM

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED_V3> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> • Choose the #CERT_S_SM_DPauth_INV_SIGN and the remaining part of the Variant C Certificates chain leading to the same Root CI certificate as the one chosen for signing		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_RPM_ICCID1, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_INV_SIGN, #CERT_S_SM_DP_SubCAList_SIG, #CRL_LIST, TRUE))	MTD_CHECK_AUTH_ERROR_RESP(`invalidCertificate`) SW = 0x9000

4.2.18.2.13 TC_eUICC_ES10b.AuthenticateServerV3_SM-DP+_Independently_Of_Server_Variant

The purpose of the test sequences in this Test Case is to verify that regardless of the Variant of eUICC Certificate chain configured in the eUICC, the eUICC is able to verify the SM-DP+ Certificates chain, including if the server uses a different Variant.

This Test Case does not aim to test all combinations of options, so a single one is chosen, using MatchingId and no IMEI in CtxParams1.

Test Sequence #01 Nominal: with server Certificate chain Variant O

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000
			Extract the <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION> from response data and verify that it contains at least #IUT_SERVER_CI_PK_ID
			Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> and < EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> from response data and verify that at least one of them contains #IUT_EUICC_CERT_CHAIN_VARIANT_CI_PK_ID

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED> and <EUICC_CI_PK_ID_TO_BE_USED_V3> to #IUT_EUICC_CERT_CHAIN_VARIANT_CI_PK_ID • Choose the #CERT_S_SM_DPauth_SIG leading to the Root CI identified by #IUT_SERVER_CI_PK_ID		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP_USING_IUT_VARIANT (#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_MATCH_ID, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_SIG, NO_PARAM, #CRL_LIST))	MTD_CHECK_AUTH_SERVER_RESP (#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_MATCH_ID) SW = 0x9000

Test Sequence #02 Nominal: with server Certificate chain Variant A

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000

			Extract the <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION> from response data and verify that it contains at least #IUT_SERVER_CI_PK_ID Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> and <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> from response data and verify that at least one of them contains #IUT_EUICC_CERT_CHAIN_VARIANT_CI_PK_ID

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED> and <EUICC_CI_PK_ID_TO_BE_USED_V3> to #IUT_EUICC_CERT_CHAIN_VARIANT_CI_PK_ID • Choose the #CERT_S_SM_DPauth_SIG and the remaining part of the Variant A Certificates chain leading to the Root CI identified by #IUT_SERVER_CI_PK_ID		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP_USING_IUT_VARIANT (#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_MATCH_ID, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_SIG, #CERT_S_SM_DP_SubCA_SIG, #CRL_LIST))	MTD_CHECK_AUTH_SERVER_RESP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_MATCH_ID) SW = 0x9000

Test Sequence #03 Nominal: with Server Certificate chain Variant B

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000
			Extract the <EUICC_CI_PK_ID_LIST_FOR_VERIFIC

			ATION> from response data and verify that it contains at least #IUT_SERVER_CI_PK_ID Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> and <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> from response data and verify that at least one of them contains #IUT_EUICC_CERT_CHAIN_VARIANT_CI_PK_ID

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED> and <EUICC_CI_PK_ID_TO_BE_USED_V3> to #IUT_EUICC_CERT_CHAIN_VARIANT_CI_PK_ID • Choose the #CERT_S_SM_DPauth_SIG and the remaining part of the Variant B Certificates chain leading to the Root CI identified by #IUT_SERVER_CI_PK_ID		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP_USING_IUT_VARIANT (#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_MATCH_ID, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_SIG, #CERT_CI_SubCA_SIG, #CRL_LIST))	MTD_CHECK_AUTH_SERVER_RESP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_MATCH_ID) SW = 0x9000

Test Sequence #04 Nominal: with Server Certificate chain Variant C

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000
			Extract the <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION> from response data and verify that

			it contains at least #IUT_SERVER_CI_PK_ID Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> and <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> from response data and verify that at least one of them contains #IUT_EUICC_CERT_CHAIN_VARIANT _CI_PK_ID

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE> • <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED> and <EUICC_CI_PK_ID_TO_BE_USED_V3> to #IUT_EUICC_CERT_CHAIN_VARIANT _CI_PK_ID • Choose the #CERT_S_SM_DPauth_SIG and the remaining part of the Variant C Certificates chain leading to the Root CI identified by #IUT_SERVER_CI_PK_ID		
4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_AUTHENTICATE_SMDP_USING _IUT_VARIANT (#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_MATCH_ID, <S_SMDP_SIGNATURE1>, #CERT_S_SM_DPauth_SIG, #CERT_S_SM_DP_SubCAList_SIG, #CRL_LIST))	MTD_CHECK_AUTH_SERVER_RESP(#TEST_DP_ADDRESS1, <S_SMDP_CHALLENGE>, #CTX_PARAMS1_MATCH_ID) SW = 0x9000

4.2.18.2.14 TC_eUICC_ES10b.AuthenticateServerV3_SM-DS_Independently_Of_Server_Variant

The purpose of the test sequences in this Test Case is to verify that regardless of the Variant of eUICC Certificate chain configured in the eUICC, the eUICC is able to verify the SM-DS Certificates chain, including if the server uses a different Variant.

This Test Case does not aim to test all combinations of options, so a single one is chosen, using EventId and no IMEI in CtxParams1.

Test Sequence #01 Nominal: with server Certificate Chain Variant O

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_INFO1)	#R_EUICC_INFO1
			SW = 0x9000
			Extract the <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION> from response data and verify that it contains at least #IUT_SERVER_CI_PK_ID
			Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> and < EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3> from response data and verify that at least one of them contains #IUT_EUICC_CERT_CHAIN_VARIANT _CI_PK_ID

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_EUICC_CHALLENGE)	#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

3	The following inputs are required for Step 4 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDS_CHALLENGE> • <S_SMDS_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED> and <EUICC_CI_PK_ID_TO_BE_USED_V3> to #IUT_EUICC_CERT_CHAIN_VARIANT _CI_PK_ID • Choose the #CERT_S_SM_DPauth_SIG leading to the Root CI identified by #IUT_SERVER_CI_PK_ID		

4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(
		
		MTD_AUTHENTICATE_SMDS_USING_IUT_VARIANT (
		#TEST_ROOT_DS_ADDRESS,
		<S_SMDS_CHALLENGE>,
			MTD_CHECK_AUTH_SERVER_RESP(
			#TEST_ROOT_DS_ADDRESS,
			<S_SMDS_CHALLENGE>,
			#CTX_PARAMS1_EVENT_ID)
			SW = 0x9000

		#CTX_PARAMS1_EVENT_ID, <S_SMDS_SIGNATURE1>, #CERT_S_SM_DSauth_SIG, NO_PARAM, #CRL_LIST))	

Test Sequence #02 Nominal: with server Certificate Chain Variant A

This test sequence is FFS.

Test Sequence #03 Nominal: with server Certificate Chain Variant B

This test sequence is FFS.

Test Sequence #04 Nominal: with server Certificate Chain Variant C

This test sequence is FFS.

4.2.19 ES10b (LPA -- eUICC): CancelSession

4.2.19.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 2.6.7.2
- Section 3.0.2, 3.1.3, 3.1.3.2, 3.1.3.3
- Section 5.7.14

4.2.19.2 Test Cases

4.2.19.2.1 TC_eUICC_ES10b.CancelSessionNIST

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 is not loaded on the eUICC.
eUICC	The communication between the S_Device and the eUICC has been initialized and the S_LPAd has selected the ISD-R. Common Mutual Authentication procedure has been successfully executed between the eUICC and the S_SM-DP+ (i.e. the response has been sent by the eUICC for ES10b.AuthenticateServer)

	• the same GSMA CI based on NIST P-256 curve has been chosen for signing and for verification

Test Sequence #01 Nominal: End User Rejection

Step	Direction	Sequence / Description	Expected result

1	S_LPAd → eUICC	MTD_STORE_DATA(#CANCEL_SESSION_REJECT)	#R_CANCEL_SESSION_REJ
			SW = 0x9000
			The <EUICC_CS_SIGNATURE> SHALL be verified with the #PK_EUICC_SIG.
			Verify that the <S_TRANSACTION_ID> present in the response is the same as in #CANCEL_SESSION_REJECT

2	PROC_VERIFY_SESSION_IS_CANCELLED		

Test Sequence #02 Nominal: End User Postponed

Step	Direction	Sequence / Description	Expected result

1	S_LPAd → eUICC	MTD_STORE_DATA(#CANCEL_SESSION_POSTPONED)	#R_CANCEL_SESSION_POSTPONED
			SW = 0x9000
			The <EUICC_CS_SIGNATURE> SHALL be verified with the #PK_EUICC_SIG.
			Verify that the <S_TRANSACTION_ID> present in the response is the same as in #CANCEL_SESSION_POSTPONED

2	PROC_VERIFY_SESSION_IS_CANCELLED		

Test Sequence #03 Nominal: Timeout

The RSP session is delayed because the End User does not confirm the download of the Profile within the timeout interval defined by the LPAd.

Step	Direction	Sequence / Description	Expected result

1	S_LPAd → eUICC	MTD_STORE_DATA(#CANCEL_SESSION_TIMEOUT)	#R_CANCEL_SESSION_TIMEOUT
			SW = 0x9000
			The <EUICC_CS_SIGNATURE> SHALL be verified with the #PK_EUICC_SIG.
			Verify that the <S_TRANSACTION_ID> present in the response is the same as in #CANCEL_SESSION_TIMEOUT

2	PROC_VERIFY_SESSION_IS_CANCELLED		

Test Sequence #04 Nominal: PPR not allowed

The RSP session is terminated because the LPAd detected that PPR(s) set in the Profile Metadata is/are not allowed.

Step	Direction	Sequence / Description	Expected result

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#CANCEL_SESSION_PPR)
			#R_CANCEL_SESSION_PPR
			SW = 0x9000
			The <EUICC_CS_SIGNATURE> SHALL be verified with the #PK_EUICC_SIG.
			Verify that the <S_TRANSACTION_ID> present in the response is the same as in #CANCEL_SESSION_PPR

2	PROC_VERIFY_SESSION_IS_CANCELLED		

Test Sequence #05 Nominal: Metadata Mismatch

The RSP session is terminated because the LPAd detected that the Profile Metadata in the response to "ES9+.AuthenticateClient" does not match the Profile Metadata in the Bound Profile Package.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	Sub-procedure Profile Download and Installation – End User Confirmation has been successfully executed between the eUICC and the S_SM-DP+ (i.e. the response has been sent by the eUICC for ES10b.PrepareDownload) • #PREP_DOWNLOAD_NO_CC has been sent to the eUICC

Step	Direction	Sequence / Description	Expected result

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#CANCEL_SESSION_METADATA)
			#R_CANCEL_SESSION_METADATA
			SW = 0x9000
			The <EUICC_CS_SIGNATURE> SHALL be verified with the #PK_EUICC_SIG.
			Verify that the <S_TRANSACTION_ID> present in the response is the same as in #CANCEL_SESSION_METADATA

2	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
3	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_OP_PROF1, NO_PARAM, #UPP_OP_PROF1)		

4	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT> • <BPP_SEG_A0> • <BPP_SEG_A1> • <BPP_SEG_A3>		
5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x6985 or 0x6A88
6	PROC_VERIFY_SESSION_IS_CANCELLED		

Test Sequence #06 Nominal: BPP Parsing Error

The RSP session is terminated because the LPAd has encountered an error while parsing the Bound Profile Package received from the SM-DP+.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	Sub-procedure Profile Download and Installation – End User Confirmation has been successfully executed between the eUICC and the S_SM-DP+ (i.e. the response has been sent by the eUICC for ES10b.PrepareDownload) • #PREP_DOWNLOAD_NO_CC has been sent to the eUICC

Step	Direction	Sequence / Description	Expected result

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#CANCEL_SESSION_LOAD_BPP)
			#R_CANCEL_SESSION_LOAD_BPP
			SW = 0x9000
			The <EUICC_CS_SIGNATURE> SHALL be verified with the #PK_EUICC_SIG.
			Verify that the <S_TRANSACTION_ID> present in the response is the same as in #CANCEL_SESSION_LOAD_BPP

2	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
3	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_OP_PROF1, NO_PARAM, #UPP_OP_PROF1)		
4	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT> • <BPP_SEG_A0> • <BPP_SEG_A1> • <BPP_SEG_A3>		
5	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x6985 or 0x6A88

6	PROC_VERIFY_SESSION_IS_CANCELLED

Test Sequence #07 Nominal: Load BPP Execution Error

The RSP session is terminated because the LPAd has encountered an error while installing the Bound Profile Package received from the SM-DP+.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	Sub-procedure Profile Download and Installation – End User Confirmation has been successfully executed between the eUICC and the S_SM-DP+ (i.e. the response has been sent by the eUICC for ES10b.PrepareDownload) • #PREP_DOWNLOAD_NO_CC has been sent to the eUICC

Step	Direction	Sequence / Description	Expected result
IC1	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
IC2	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_PROF1, #METADATA_OP_PROF1, NO_PARAM, #UPP_OP_PROF1)		
IC3	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT> • <BPP_SEG_A0> • <BPP_SEG_A1> • <BPP_SEG_A3>		
IC4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#CANCEL_SESSION_LOAD_BPP)
			#R_CANCEL_SESSION_LOAD_BPP
			SW = 0x9000
			The <EUICC_CS_SIGNATURE> SHALL be verified with the #PK_EUICC_SIG.
			Verify that the <S_TRANSACTION_ID> present in the response is the same as in #CANCEL_SESSION_LOAD_BPP

2	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0)	SW=0x6985 or 0x6A88
3	PROC_VERIFY_SESSION_IS_CANCELLED		

Test Sequence #08 Nominal: Undefined Reason

Step	Direction	Sequence / Description	Expected result

1	S_LPAd → eUICC	MTD_STORE_DATA(#CANCEL_SESSION_UNDEF)	#R_CANCEL_SESSION_UNDEF
			SW = 0x9000
			The <EUICC_CS_SIGNATURE> SHALL be verified with the #PK_EUICC_SIG.
			Verify that the <S_TRANSACTION_ID> present in the response is the same as in #CANCEL_SESSION_UNDEF

2	PROC_VERIFY_SESSION_IS_CANCELLED		

4.2.19.2.2 TC_eUICC_ES10b.CancelSessionBRP

In these test sequences, once the RSP session has been cancelled, verifications are performed in order to check that it is neither possible to execute the Download Confirmation procedure nor to execute the Common Mutual Authentication procedure by referring to the cancelled TransactionID.

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 is not loaded on the eUICC.
eUICC	The communication between the S_Device and the eUICC has been initialized and the S_LPAd has selected the ISD-R. Common Mutual Authentication procedure has been successfully executed between the eUICC and the S_SM-DP+ • the same GSMA CI based on BrainpoolP256r1 curve has been chosen for signing and for verification

Test Sequence #01 Nominal: End User Rejection

This test sequence SHALL be the same as the Test Sequence #01 defined in section 4.2.19.2.1 – TC_eUICC_ES10b.CancelSessionNIST except that all keys and certificates SHALL be based on BrainpoolP256r1.

Test Sequence #02 Nominal: End User Postponed

This test sequence SHALL be the same as the Test Sequence #02 defined in section 4.2.19.2.1 – TC_eUICC_ES10b.CancelSessionNIST except that all keys and certificates SHALL be based on BrainpoolP256r1.

Test Sequence #03 Nominal: Timeout

This test sequence SHALL be the same as the Test Sequence #03 defined in section 4.2.19.2.1 – TC_eUICC_ES10b.CancelSessionNIST except that all keys and certificates SHALL be based on BrainpoolP256r1.

Test Sequence #04 Nominal: PPR not allowed

This test sequence SHALL be the same as the Test Sequence #04 defined in section 4.2.19.2.1 – TC_eUICC_ES10b.CancelSessionNIST except that all keys and certificates SHALL be based on BrainpoolP256r1.

Test Sequence #05 Nominal: Metadata Mismatch

Initial Conditions	
Entity	**Description of the initial state**
eUICC	Sub-procedure Profile Download and Installation – End User Confirmation has been successfully executed between the eUICC and the S_SM-DP+ • #PREP_DOWNLOAD_NO_CC has been sent to the eUICC

This test sequence SHALL be the same as the Test Sequence #05 defined in section 4.2.19.2.1 – TC_eUICC_ES10b.CancelSessionNIST except that all keys and certificates SHALL be based on BrainpoolP256r1.

Test Sequence #06 Nominal: BPP Parsing Error

Initial Conditions	
Entity	**Description of the initial state**
eUICC	Sub-procedure Profile Download and Installation – End User Confirmation has been successfully executed between the eUICC and the S_SM-DP+ • #PREP_DOWNLOAD_NO_CC has been sent to the eUICC

This test sequence SHALL be the same as the Test Sequence #06 defined in section 4.2.19.2.1 – TC_eUICC_ES10b.CancelSessionNIST except that all keys and certificates SHALL be based on BrainpoolP256r1.

Test Sequence #07 Nominal: Load BPP Execution Error

Initial Conditions	
Entity	**Description of the initial state**
eUICC	Sub-procedure Profile Download and Installation – End User Confirmation has been successfully executed between the eUICC and the S_SM-DP+ • #PREP_DOWNLOAD_NO_CC has been sent to the eUICC

This test sequence SHALL be the same as the Test Sequence #07 defined in section 4.2.19.2.1 – TC_eUICC_ES10b.CancelSessionNIST except that all keys and certificates SHALL be based on BrainpoolP256r1.

Test Sequence #08 Nominal: Undefined Reason

This test sequence SHALL be the same as the Test Sequence #08 defined in section 4.2.19.2.1 – TC_eUICC_ES10b.CancelSessionNIST except that all keys and certificates SHALL be based on BrainpoolP256r1.

4.2.19.2.3 VOID

4.2.19.2.4 TC_eUICC_ES10b.CancelSession_ErrorCase

Test Sequence #01 Error: No on-going RSP session

On receiving a CancelSession request whereas there is no on-going RSP session, the eUICC SHALL return an error code.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	No RSP session is on-going (i.e. no Common Mutual Authentication procedure has been executed).

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(#CANCEL_SESSION_INV_TRANS_ID)	#R_CANCEL_SESSION_INV_TRANS_ID SW = 0x9000

Test Sequence #02 Error: Invalid Transaction ID

On receiving a CancelSession request with a TransactionID different from the on-going one, the eUICC SHALL not discard the current RSP session and return an error code.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The communication between the S_Device and the eUICC has been initialized and the S_LPAd has selected the ISD-R. • #GET_EUICC_INFO1, #GET_EUICC_CHALLENGE and MTD_AUTHENTICATE_SMDP have been sent to the eUICC as defined in section 2.2.3.5

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA(#CANCEL_SESSION_INV_TRANS_ID)	#R_CANCEL_SESSION_INV_TRANS_ID SW = 0x9000
2	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(#PREP_DOWNLOAD_NO_CC)	#R_PREP_DOWNLOAD_NO_CC SW=0x9000

4.2.20 ES10c (LPA -- eUICC): GetProfilesInfo

4.2.20.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 2.4.5.2
- Section 3.1.4
- Section 3.2.4
- Section 5.7.15

4.2.20.2 Test Cases

4.2.20.2.1 TC_eUICC_ES10c.GetProfilesInfo

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 is Enabled.
eUICC	The Nickname of PROFILE_OPERATIONAL1 and PROFILE_OPERATIONAL2 is empty.
eUICC	The Nickname of the PROFILE_OPERATIONAL3 is equal to #NICKNAME3.

Test Sequence #01 Nominal: Get All Profiles

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_PROFILES_INFO_ALL)	response ProfileInfoListResponse::= profileInfoListOk: {
			#PROFILE_INFO1,
			#PROFILE_INFO2,
			#PROFILE_INFO3
			}
			SW = 0x9000

Test Sequence #02 Nominal: Get Profile by ICCID

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		

IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW = 0x9000

Test Sequence #03 Nominal: Get Profile by AID

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW = 0x9000

Test Sequence #04 Nominal: Get All Operational Profiles

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA
		(#GET_PROFILES_INFO_PROFCLASS)
			response ProfileInfoListResponse::=
			profileInfoListOk: {
			#PROFILE_INFO1,
			#PROFILE_INFO2,
			#PROFILE_INFO3
			}
			SW = 0x9000

Test Sequence #05 Nominal: Get Profile ICCID list

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_PROFILES_INFO_TAGLIST_ICCID)	response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILES_INFO_TAGLIST_ICCID
			}
			SW = 0x9000

Test Sequence #06 Nominal: Get Profile AID list

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_PROFILES_INFO_TAGLIST_ISDPAID)	response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILES_INFO_TAGLIST_ISDPAID
			}
			SW = 0x9000

Test Sequence #07 Nominal: Get Profile Nickname list

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_PROFILES_INFO_TAGLIST_PROFILE_NICKNAME)	response ProfileInfoListResponse::= profileInfoListOk : {
			…
			#PROFILES_INFO_TAGLIST_PROFILE_NICKNAME
			...
			}
			SW = 0x9000

Test Sequence #08 Nominal: Get Profile SP Name list

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_PROFILES_INFO_TAGLIST_SP_NAME)	response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILES_INFO_TAGLIST_SP_NAME
			}
			SW = 0x9000

Test Sequence #09 Nominal: Get Profile Name list

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_PROFILES_INFO_TAGLIST_PROFILE_NAME)	response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILES_INFO_TAGLIST_PROFILE_NAME
			}
			SW = 0x9000

Test Sequence #10 Nominal: Get Profile Icon list

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_PROFILES_INFO_TAGLIST_ICON)	response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILES_INFO_TAGLIST_ICON
			}
			SW = 0x9000

Test Sequence #11 Nominal: Get Profile Owner list

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_PROFILES_INFO_TAGLIST_PROFILE_OWNER)	response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILES_INFO_TAGLIST_PROFILE_OWNER
			}
			SW = 0x9000

Test Sequence #12 Nominal: Get Profile SM-DP+ proprietary data list

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC with dpProprietaryData #SMDP_PROP_DATA1 (i.e. #CONF_ISDP_PROF1 is used during the Profile downloading).

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_PROFILES_INFO_TAGLIST_SMDP_PROP_DATA)	response ProfileInfoListResponse::= profileInfoListOk : {
			…
			#PROFILES_INFO_TAGLIST_SMDP_PROP_DATA
			…
			}
			SW = 0x9000

Test Sequence #13 Nominal: Get Profile ICCID and State list

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_PROFILES_INFO_TAGLIST1)	response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILES_INFO_TAGLIST1
			}
			SW = 0x9000

Test Sequence #14 Nominal: Get Profile Nickname and State list

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_PROFILES_INFO_TAGLIST2)	response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILES_INFO_TAGLIST2
			}
			SW = 0x9000

Test Sequence #15 Nominal: Get Profile Icon and Icon Type list

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_PROFILES_INFO_TAGLIST3)	response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILES_INFO_TAGLIST3
			}

			SW = 0x9000

Test Sequence #16 Nominal: Get Profile Icon and State list

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_PROFILES_INFO_TAGLIST4)	response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILES_INFO_TAGLIST4
			}
			SW = 0x9000

Test Sequence #17 Nominal: Get Operational Profile ICCID and State list

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_PROFILES_INFO_OPTAGLIST1)	response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILES_INFO_TAGLIST1
			}
			SW = 0x9000

Test Sequence #18 Nominal: Get Operational Profile Nickname and State list

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_PROFILES_INFO_OPTAGLIST2)	response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILES_INFO_TAGLIST2
			}
			SW = 0x9000

Test Sequence #19 Nominal: Get Operational Profile Icon and Icon type list

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_PROFILES_INFO_OPTAGLIST3)	response ProfileInfoListResponse::= profileInfoListOk : { #PROFILES_INFO_TAGLIST3

			} SW = 0x9000

Test Sequence #20 Nominal: Get Operational Profile Icon and State list

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_PROFILES_INFO_OPTAGLIST4)	response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILES_INFO_TAGLIST4
			}
			SW = 0x9000

Test Sequence #21 Nominal: Get Profile State of the defined Profile

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_PROFILES_INFO_ICCID_TAGLIST1)	response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILES_INFO_ICCID_TAGLIST1
			}
			SW = 0x9000

Test Sequence #22 Nominal: Get Profile Icon Type of the defined Profile

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_PROFILES_INFO_ICCID_TAGLIST2)	response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILES_INFO_ICCID_TAGLIST2
			}
			SW = 0x9000

Test Sequence #23 Nominal: Get Profile Class of the defined Profile

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_PROFILES_INFO_ICCID_TAGLIST3)	response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILES_INFO_ICCID_TAGLIST 3
			}
			SW = 0x9000

Test Sequence #24 Nominal: Get Notification Configuration of the defined Profile

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_PROFILES_INFO_ICCID_TAGLIST4)	response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILES_INFO_ICCID_TAGLIST 4
			}
			SW = 0x9000

Test Sequence #25 Nominal: Get Profile Policy Rules of the defined Profile

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_PROFILES_INFO_ICCID_TAGLIST5)	response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILES_INFO_ICCID_TAGLIST 5
			}
			SW = 0x9000

Test Sequence #26 Nominal: Get empty Profile list. No Profile installed

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	No Profile is loaded on the eUICC (this condition overrides the general initial condition defined in this test case).

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA (#GET_PROFILES_INFO_ALL)	response ProfileInfoListResponse::= profileInfoListOk: {

			} SW = 0x9000

4.2.20.2.2 TC_eUICC_ES10c.GetProfilesInfo_ErrorCases

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.
eUICC	The Nickname of PROFILE_OPERATIONAL1 is empty.

Test Sequence #01 Error: Get Profiles during a Profile Enabling

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_ENABLE_PROFILE(#ICCID_OP_PROF1, NO_PARAM, TRUE))	No response data is returned SW=0x91XX
2	S_LPAd → eUICC	MTD_STORE_DATA(#GET_PROFILES_INFO_ALL)	SW = 0x6985
3	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“UICC Reset”)
4	Repeat IC1 and IC2		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::= profileInfoListOk: {
			#PROFILE_INFO1
			}
			SW = 0x9000

6	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

7	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #02 Error: Get Profiles during a Profile Disabling

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_DISABLE_PROFILE(#ICCID_OP_PROF1, NO_PARAM, TRUE))	No response data is returned SW=0x91XX
2	S_LPAd → eUICC	MTD_STORE_DATA(#GET_PROFILES_INFO_ALL)	SW = 0x6985
3	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“UICC Reset”)
4	Repeat IC1 and IC2		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::=
			profileInfoListOk: {
			#PROFILE_INFO1_DISABLED
			}
			
			SW = 0x9000

6	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #03 Error: Get Profiles during an eUICC Memory Reset

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(#EUICC_MEMORY_RESET_OP_PRO)	#R_EUICC_MEMORY_RESET_OK SW=0x91XX
2	S_LPAd → eUICC	MTD_STORE_DATA(#GET_PROFILES_INFO_ALL)	SW = 0x6985
3	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“UICC Reset”)
4	Repeat IC1 and IC2		
5	S_LPAd → eUICC	MTD_STORE_DATA(#GET_PROFILES_INFO_ALL)	response ProfileInfoListResponse::= profileInfoListOk: { } SW = 0x9000

4.2.20.2.3 TC_eUICC_ES10c.GetProfilesInfo_MEPA1

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 is Enabled.
eUICC	The Nickname of PROFILE_OPERATIONAL1 and PROFILE_OPERATIONAL2 is empty.
eUICC	The Nickname of the PROFILE_OPERATIONAL3 is equal to #NICKNAME3.

Test Sequence #01 Nominal: Get Profile ICCID, State and enable on eSIM Port list

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	eUICC is MEP capable
eUICC	The PROFILE_OPERATIONAL1 is Enabled on Port 1
eUICC	The PROFILE_OPERATIONAL2 is Enabled on Port 2

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify ‘LSI Support’ is present in <ATR>

IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, “010203”, 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO_TAGLIST (
		‘5A9F709F24’))
			response ProfileInfoListResponse::= profileInfoListOk : {
			
			#PROFILES_INFO_TAGLIST5_MEPA1;
			}
			SW=0x9000

4.2.20.2.4 TC_eUICC_ES10c.GetProfilesInfo_MEPA2

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 is Enabled.
eUICC	The Nickname of PROFILE_OPERATIONAL1 and PROFILE_OPERATIONAL2 is empty.
eUICC	The Nickname of the PROFILE_OPERATIONAL3 is equal to #NICKNAME3.

Test Sequence #01 Nominal: Get Profile ICCID, State and enable on eSIM Port list

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	eUICC is MEP capable
eUICC	The PROFILE_OPERATIONAL1 is Enabled
eUICC	The PROFILE_OPERATIONAL2 is Enabled

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO_TAGLIST (
		'5A9F709F24'))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILES_INFO_TAGLIST5_MEPA2_12
			OR
			PROFILES_INFO_TAGLIST5_MEPA2_21;
			}
			SW=0x9000

4.2.20.2.5 TC_eUICC_ES10c.GetProfilesInfo_MEPB

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 is Enabled.
eUICC	The Nickname of PROFILE_OPERATIONAL1 and PROFILE_OPERATIONAL2 is empty.
eUICC	The Nickname of the PROFILE_OPERATIONAL3 is equal to #NICKNAME3.

Test Sequence #01 Nominal: Get Profile ICCID, State and enable on eSIM Port list

The purpose of this test sequence is to verify that the fields profileState and enabledOnEsimPort properly reflect the Profile's state depending on the Command Port.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	eUICC is MEP capable
eUICC	The PROFILE_OPERATIONAL1 is Enabled on Port 0
eUICC	The PROFILE_OPERATIONAL2 is Enabled on Port 1

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS Verify <MEP_MAX_LSIS> = #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO_TAGLIST_MEP (
		'5A9F709F24'))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILES_INFO_TAGLIST5_MEP B
			}
			SW=0x9000
			Verify that:
			- the ProfileInfo object with `#ICCID_OP_PROF1` has its `profileState` equal to `enabled` and its `enabledOnEsimPort` equal to 0
			- the ProfileInfo object with `#ICCID_OP_PROF2` has its `enabledOnEsimPort` equal to 1

2	PROC_MEP_LSI_MULTIPLEXING(1)		
3	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
4	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO_TAGLIST_MEP (	Verify that: - the ProfileInfo object with #ICCID_OP_PROF2 has its

		'5A9F709F24'))	`profileState` equal to `enabled` and its `enabledOnEsimPort` equal to 1 - the ProfileInfo object with `#ICCID_OP_PROF1` has its `enabledOnEsimPort` equal to 0

4.2.21 ES10c (LPA -- eUICC): EnableProfile

4.2.21.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 2.4.3
- Section 2.9.1, 2.9.3.3
- Section 3.2.1
- Section 3.4.3
- Section 5.7.16

4.2.21.2 Test Cases

4.2.21.2.1 TC_eUICC_ES10c.EnableProfile_Case3

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.

Test Sequence #01 Nominal: Enable Profile by ISD-P AID and "refreshFlag" set when Device supports "UICC Reset"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_ENABLE_PROFILE(NO_PARAM, <ISD_P_AID1>,	No response data is returned SW=0x91XX

		TRUE))	
2	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“UICC Reset”)
3	Repeat IC1 and IC2		

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response
			ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

5	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

6	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #02 Nominal: Enable Profile by ICCID and “refreshFlag” set when Device supports “UICC Reset”

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_ENABLE_PROFILE(#ICCID_OP_PROF1, NO_PARAM, TRUE))	No response data is returned SW=0x91XX
2	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“UICC Reset”)
3	Repeat IC1 and IC2		

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response
			ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

5	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

6	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #03 Nominal: Enable Profile by ISD-P AID and "refreshFlag" set when Device supports "eUICC Profile State Change"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_ENABLE_PROFILE(NO_PARAM, <ISD_P_AID1>, TRUE))	No response data is returned SW=0x91XX
2	S_Device → eUICC	FETCH 'XX'	REFRESH Command ("eUICC Profile State change")
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
4	Execute IC1 from step 2 to step 4		
5	Repeat IC2		

6	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response
			ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

7	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

8	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #04 Nominal: Enable Profile by ICCID and "refreshFlag" set when Device supports "eUICC Profile State Change"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_ENABLE_PROFILE(#ICCID_OP_PROF1, NO_PARAM, TRUE))	No response data is returned SW=0x91XX
2	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“eUICC Profile State change”)
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
4	Execute IC1 from step 2 to step 4		
5	Repeat IC2		

6	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response
			ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

7	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

8	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #05 Nominal: Enable Profile by ISD-P AID and “refreshFlag” not set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_ENABLE_PROFILE(NO_PARAM, <ISD_P_AID1>, FALSE))	No response data is returned SW=0x9000
2	S_Device → eUICC	[TERMINAL_PROFILE]	Toolkit initialization THEN SW=0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response
			ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

4	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

5	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #06 Nominal: Enable Profile by ICCID and "refreshFlag" not set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_ENABLE_PROFILE(#ICCID_OP_PROF1, NO_PARAM, FALSE))	No response data is returned SW=0x9000
2	S_Device → eUICC	[TERMINAL_PROFILE]	Toolkit initialization THEN SW=0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response
			ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

4	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

5	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #07 Nominal: Enable Profile by ICCID with refreshFLag set while proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x9100
IC4	Do not send FETCH command		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_ENABLE_PROFILE(#ICCID_OP_PROF2, NO_PARAM, TRUE))	No response data is returned SW=0x91YY
2	S_Device →eUICC	FETCH 'YY'	REFRESH Command (“UICC Reset”)
3	Repeat IC1 and IC2		
4	S_LPAd → eUICC	MTD_STORE_DATA(#GET_PROFILES_INFO_ALL)	response ProfileInfoListResponse::= profileInfoListOk : { #PROFILE_INFO1_DISABLED, #PROFILE_INFO2_DISABLED } SW=0x9000
5	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000
6	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2 SW=0x9000

Test Sequence #08 Nominal: Enable Profile by ICCID with refreshFLag not set while proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x9100

IC4	Do not send FETCH command		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_ENABLE_PROFILE(#ICCID_OP_PROF2, NO_PARAM, FALSE))	No response data is returned SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED,
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

3	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

4	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

Test Sequence #09 Nominal: Enable Profile by ICCID with refreshFLag set while proactive session is ongoing with Terminal Response outstanding – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
1	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
2	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_ENABLE_PROFILE(#ICCID_OP_PROF2, NO_PARAM, TRUE))	No response data returned SW=0x9000

3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x91YY
4	S_Device →eUICC	FETCH 'YY'	REFRESH Command (“UICC Reset”)
5	Repeat IC1 and IC2		

6	S_LPAd → eUICC	MTD_STORE_DATA(#GET_PROFILES_INFO_ALL)	response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED, #PROFILE_INFO2_ENABLED
			}
			SW=0x9000

7	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

8	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

Test Sequence #10 Nominal: Enable Profile by ICCID with refreshFLag not set while proactive session is ongoing with Terminal Response outstanding – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
1	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
2	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_ENABLE_PROFILE(#ICCID_OP_PROF2, NO_PARAM, FALSE))	No response data is returned SW=0x9000

3	S_Device → eUICC	TERMINAL RESPONSE	SW=any value except 91XX

4	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED,
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

5	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

6	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

4.2.21.2.2 TC_eUICC_ES10c.EnableProfile_ErrorCases_Case3

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.

Test Sequence #01 Error: Enable Profile by an unknown ISD-P AID

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The Operational Profile identified by the ISD-P AID <ISD_P_AIDX> is not loaded.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_ENABLE_PROFILE(NO_PARAM, <ISD_P_AIDX>, TRUE))	SW=0x6A82
2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(NO_PARAM, <ISD_P_AID1>))	response ProfileInfoListResponse::= profileInfoListOk : { #PROFILE_INFO1_DISABLED

			} SW=0x9000

Test Sequence #02 Error: Enable Profile by an unknown ICCID

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The Operational Profile identified by the ICCID #ICCID_OP_PROFX is not loaded.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_ENABLE_PROFILE(#ICCID_OP_PROFX, NO_PARAM, TRUE))	SW=0x6A82

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response
			ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

Test Sequence #03 Error: Enable Profile (by ISD-P AID) is not possible when this Operational Profile is in Enabled state

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_ENABLE_PROFILE(NO_PARAM,	SW=0x6985

		<ISD_P_AID1>, TRUE))	

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #04 Error: Enable Profile (by ICCID) is not possible when this Operational Profile is in Enabled state

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_ENABLE_PROFILE(#ICCID_OP_PROF1, NO_PARAM, TRUE))	SW=0x6985

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #05 Error: Enable Profile (by ISD-P AID) not possible when an Operational Profile with a PPR1 is loaded

The purpose of this test is to ensure that it is NOT possible to enable an Operational Profile when there is another Operational Profile Enabled with the Policy Rule “Disabling of this Profile is not allowed”.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	No Profile is installed on the eUICC. (this condition overrides the general initial condition defined in this test case)

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL4 NOTE: The PROFILE_OPERATIONAL4 corresponds to <ISD_P_AID4>.		
IC4	Install PROFILE_OPERATIONAL1 NOTE: The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.		
IC5	Enable PROFILE_OPERATIONAL4		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_ENABLE_PROFILE(NO_PARAM, <ISD_P_AID1>, TRUE))	SW=0x6985

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response
			ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			#PROFILE_INFO4_ENABLED
			}
			
			SW=0x9000

Test Sequence #06 Error: Enable Profile (by ICCID) not possible with an Operational Profile with PPR1 is loaded

The purpose of this test is to ensure that it is NOT possible to enable an Operational Profile when there is another Operational Profile Enabled with the Policy Rule “Disabling of this Profile is not allowed”.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	No Profile is installed on the eUICC. (this condition overrides the general initial condition defined in this test case)

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL4		

IC4	Install PROFILE_OPERATIONAL1		
IC5	Enable PROFILE_OPERATIONAL4		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_ENABLE_PROFILE(#ICCID_OP_PROF1, NO_PARAM, TRUE))	SW=0x6985

2	S_LPAd → eUICC	MTD_STORE_DATA (#GET_PROFILES_INFO_ALL)	response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			#PROFILE_INFO4_ENABLED
			}
			
			SW=0x9000

Test Sequence #07 Error: Enable Profile by ISD-P AID without refreshFlag while proactive session is ongoing – catBusy supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC4	Do not send FETCH command		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_ENABLE_PROFILE(NO_PARAM, <ISD_P_AID2>, FALSE))	SW=0x9300
2	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK

3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1,
			#PROFILE_INFO2
			}
			SW=0x9000

Test Sequence #08 Error: Enable Profile by ICCID with refreshFLag set while proactive session is ongoing – catBusy supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC4	Do not send FETCH command		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_ENABLE_PROFILE(#ICCID_OP_PROF2, NO_PARAM, TRUE))	SW=0x9300
2	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1,
			#PROFILE_INFO2
			}
			SW=0x9000

4.2.21.2.3 TC_eUICC_ES10c.EnableProfile_Case4

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.

Test Sequence #01 Nominal: Enable Profile by ISD-P AID and "refreshFlag" set when Device supports "UICC Reset"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		TRUE))
			#R_ENABLE_PROFILE_OK
			SW=0x91XX

2	S_Device → eUICC	FETCH 'XX'	REFRESH Command ("UICC Reset")
3	Repeat IC1 and IC2		

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

5	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

6	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #02 Nominal: Enable Profile by ICCID and "refreshFlag" set when Device supports "UICC Reset"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		#ICCID_OP_PROF1,
		NO_PARAM,
		TRUE))
			#R_ENABLE_PROFILE_OK
			SW=0x91XX

2	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“UICC Reset”)
3	Repeat IC1 and IC2		

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

5	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

6	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #03 Nominal: Enable Profile by ISD-P AID and “refreshFlag” set when Device supports “eUICC Profile State Change”

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		TRUE))
			#R_ENABLE_PROFILE_OK
			SW=0x91XX

2	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“eUICC Profile State change”)
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
4	Execute IC1 from step 2 to step 4		
5	Repeat IC2		

6	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

7	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

8	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #04 Nominal: Enable Profile by ICCID and “refreshFlag” set when Device supports “eUICC Profile State Change”

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		#ICCID_OP_PROF1,
		NO_PARAM,
		TRUE))
			#R_ENABLE_PROFILE_OK
			SW=0x91XX

2	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“eUICC Profile State change”)
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
4	Execute IC1 from step 2 to step 4		
5	Repeat IC2		
6	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(	response ProfileInfoListResponse::= profileInfoListOk : {

		#ICCID_OP_PROF1, NO_PARAM))	#PROFILE_INFO1 } SW=0x9000
7	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

8	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #05 Nominal: Enable Profile by ISD-P AID and "refreshFlag" not set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		FALSE))
			#R_ENABLE_PROFILE_OK
			SW=0x9000

2	S_Device → eUICC	[TERMINAL_PROFILE]	Toolkit initialization THEN SW=0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response
			ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

4	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

5	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #06 Nominal: Enable Profile by ICCID and "refreshFlag" not set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		#ICCID_OP_PROF1,
		NO_PARAM,
		FALSE))
			#R_ENABLE_PROFILE_OK
			SW=0x9000

2	S_Device → eUICC	[TERMINAL_PROFILE]	Toolkit initialization THEN SW=0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

4	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

5	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #07 Nominal: Enable Profile by ISD-P AID and "refreshFlag" set while proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC4	Do not send FETCH command		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID2>,
		TRUE))
			resp EnableProfileResponse ::=
			{
			enableResult ok
			}
			SW=0x91YY

2	S_Device →eUICC	FETCH 'YY'	REFRESH Command (“UICC Reset”)
3	Repeat IC1 and IC2		

4	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			
			#PROFILE_INFO1_DISABLED,
			
			#PROFILE_INFO2_DISABLED
			}
			SW=0x9000

5	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

6	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

Test Sequence #08 Nominal: Enable Profile by ISD-P AID and “refreshFlag” not set while proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC4	Do not send FETCH command		
1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_ENABLE_PROFILE(NO_PARAM,	resp EnableProfileResponse ::= { enableResult ok

		<ISD_P_AID2>, FALSE))	} SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::= profileInfoListOk : {
			
			#PROFILE_INFO1_DISABLED,
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

3	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

4	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

Test Sequence #09 Nominal: Enable Profile by ISD-P AID and "refreshFlag" set while proactive session is ongoing with Terminal Response outstanding – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
1	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID2>,
		TRUE))
			resp EnableProfileResponse ::= {
			enableResult ok
			}
			SW=0x9000

3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x91YY

4	S_Device →eUICC	FETCH 'YY'	REFRESH Command (“UICC Reset”)
5	Repeat IC1 and IC2		

6	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED,
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

7	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

8	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

Test Sequence #10 Nominal: Enable Profile by ISD-P AID and “refreshFlag” not set while proactive session is ongoing with Terminal Response outstanding – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
1	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID2>,
		FALSE))
			resp EnableProfileResponse ::= {
			enableResult ok
			}
			SW=0x9000

3	S_Device → eUICC	TERMINAL RESPONSE	SW=any value except 91XX

4	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED,
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

5	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

6	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

4.2.21.2.4 TC_eUICC_ES10c.EnableProfile_ErrorCases_Case4

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.

Test Sequence #01 Error: Enable Profile by an unknown ISD-P AID

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The Operational Profile identified by the ISD-P AID <ISD_P_AIDX> is not loaded.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AIDX>,
		TRUE))
			#R_ENABLE_PROFILE_ICCID_ISDP_NOTFOUND
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(NO_PARAM,	response ProfileInfoListResponse::= profileInfoListOk : {

		<ISD_P_AID1>))	#PROFILE_INFO1_DISABLED } SW=0x9000

Test Sequence #02 Error: Enable Profile by an unknown ICCID

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The Operational Profile identified by the ICCID #ICCID_OP_PROFX is not loaded.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		#ICCID_OP_PROFX,
		NO_PARAM,
		TRUE))
			#R_ENABLE_PROFILE_ICCID_ISDP_NOTFOUND
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

Test Sequence #03 Error: Enable Profile (by ISD-P AID) is not possible when this Operational Profile is in Enable state

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		TRUE))
			#R_ENABLE_PROFILE_NOT_DISABLE_STATE
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #04 Error: Enable Profile (by ICCID) is not possible when this Operational Profile is in Enabled state

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		#ICCID_OP_PROF1,
		NO_PARAM,
		TRUE))
			#R_ENABLE_PROFILE_NOT_DISABLE_STATE
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #05 Error: Enable Profile (by ISD-P AID) not possible when an Operational Profile with PPR1 is loaded

The purpose of this test is to ensure that it is NOT possible to enable an Operational Profile when there is another Operational Profile Enabled with the Policy Rule “Disabling of this Profile is not allowed”.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	No Profile is installed on the eUICC. (this condition overrides the general initial condition defined in this test case)

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL4 NOTE: The PROFILE_OPERATIONAL4 corresponds to <ISD_P_AID4>.		
IC4	Install PROFILE_OPERATIONAL1 NOTE: The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.		
IC5	Enable PROFILE_OPERATIONAL4		

1	S_LPAd → eUICC	MTD_STORE_DATA (
		MTD_ENABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		TRUE))
			#R_ENABLE_PROFILE_DISALLOWEDbyPOLICY
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			#PROFILE_INFO4_ENABLED
			}
			SW=0x9000

Test Sequence #06 Error: Enable Profile (by ICCID) not possible when an Operational Profile with PPR1 is loaded

The purpose of this test is to ensure that it is NOT possible to enable an Operational Profile when there is another Operational Profile Enabled with the Policy Rule “Disabling of this Profile is not allowed”.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	No Profile is installed on the eUICC. (this condition overrides the general initial condition defined in this test case)

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL4		
IC4	Install PROFILE_OPERATIONAL1		
IC5	Enable PROFILE_OPERATIONAL4		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		#ICCID_OP_PROF1,
		NO_PARAM,
		TRUE))
			#R_ENABLE_PROFILE_DISALLOWEDbyPOLICY
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			#PROFILE_INFO4_ENABLED
			}
			
			SW=0x9000

Test Sequence #07 Error: Enable Profile by ISD-P AID without refreshFlag while proactive session is ongoing – catBusy supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC4	Do not send FETCH command		
1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_ENABLE_PROFILE(NO_PARAM,	resp EnableProfileResponse ::= { enableResult catBusy

		<ISD_P_AID2>, FALSE))	} SW=0x9000 or 0x91XX
2	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1,
			#PROFILE_INFO2
			}
			SW=0x9000

Test Sequence #08 Error: Enable Profile by ICCID with refreshFlag set while proactive session is ongoing – catBusy supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC4	Do not send FETCH command		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		#ICCID_OP_PROF2,
		NO_PARAM,
		TRUE))
			resp EnableProfileResponse ::=
			{
			enableResult catBusy
			}
			SW=0x9000 or 0x91XX

2	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
4	S_LPAd → eUICC	MTD_STORE_DATA(#GET_PROFILES_INFO_ALL)	response ProfileInfoListResponse::= profileInfoListOk : {

			#PROFILE_INFO1, #PROFILE_INFO2 } SW=0x9000

4.2.21.2.5 TC_eUICC_ES10c.EnableProfile_ErrorCases_Case4_catBusySupported

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.

Test Sequence #01 Error: Enable Profile by ISD-P AID without refreshFlag while proactive session is ongoing

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC4	Do not send FETCH command		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID2>,
		FALSE))
			resp EnableProfileResponse ::=
			{
			enableResult catBusy
			}
			SW= 0x91XX

2	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1,
			#PROFILE_INFO2
			}

			SW=0x9000

Test Sequence #02 Error: Enable Profile by ICCID with refreshFlag set while proactive session is ongoing

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC4	Do not send FETCH command		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		#ICCID_OP_PROF2,
		NO_PARAM,
		TRUE))
			resp EnableProfileResponse ::=
			{
			enableResult catBusy
			}
			SW= 0x91XX

2	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1,
			#PROFILE_INFO2
			}
			SW=0x9000

4.2.21.2.6 TC_eUICC_ES10c.EnableProfile_Case4_catBusyNotSupported

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.

Test Sequence #01: Enable Profile by ISD-P AID without refreshFlag while proactive session is ongoing

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC4	Do not send FETCH command		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID2>,
		FALSE))
			resp EnableProfileResponse ::=
			{
			#R_ENABLE_PROFILE_OK
			}
			SW=0x9000

2	S_Device → eUICC	[TERMINAL_PROFILE]	Toolkit initialization THEN SW=0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response
			ProfileInfoListResponse::=
			profileInfoListOk : {
			
			#PROFILE_INFO1_DISABLED,
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

4	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

5	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

Test Sequence #02: Enable Profile by ICCID with refreshFlag set while proactive session is ongoing

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.

eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC4	Do not send FETCH command		

1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_ENABLE_PROFILE(#ICCID_OP_PROF2, NO_PARAM, TRUE))	#R_ENABLE_PROFILE_OK
			SW= 0x91YY
			
			Note: 91YY MAY be different from 91XX in IC3

2	S_Device → eUICC	FETCH 'YY'	REFRESH Command (“UICC Reset”)
3	Repeat IC1 and IC2		

4	S_LPAd → eUICC	MTD_STORE_DATA(#GET_PROFILES_INFO_ALL)	response ProfileInfoListResponse::= profileInfoListOk : {
			
			#PROFILE_INFO1_DISABLED,
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

5	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

6	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

4.2.21.2.7 TC_eUICC_ES10c.EnableProfile_ErrorCases_Case3_catBusySupported

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.

Test Sequence #01 Error: Enable Profile by ISD-P AID without refreshFlag while proactive session is ongoing

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.

eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC4	Do not send FETCH command		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_ENABLE_PROFILE(NO_PARAM, <ISD_P_AID2>, FALSE))	SW=0x9300
2	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1,
			#PROFILE_INFO2
			}
			SW=0x9000

Test Sequence #02 Error: Enable Profile by ICCID with refreshFLag set while proactive session is ongoing

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC4	Do not send FETCH command		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_ENABLE_PROFILE(#ICCID_OP_PROF2, NO_PARAM, TRUE))	SW=0x9300
2	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1,
			#PROFILE_INFO2
			}
			SW=0x9000

4.2.21.2.8 TC_eUICC_ES10c.EnableProfile_Case4_MEPA1

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	eUICC in MEP mode
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.

Test Sequence #01 Nominal: Enable 1st Profile by ISD-P AID and "refreshFlag" set when Device supports "UICC Reset"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2,	Verify <MEP_MODE> = '01', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS,

		#IUT_MEP_LSI_OPTIONS, “010203”, 2)	Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		NO_PARAM,
		<ISD_P_AID1>,
		TRUE, 1))
			#R_ENABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(1, “UICC Reset”)		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_FIRST_PROFILE		
4	PROC_MEP_LSI_MULTIPLEXING(0)		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response
			ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

6	PROC_MEP_LSI_MULTIPLEXING(1)		
7	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

8	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #02 Nominal: Enable Profile by ICCID and “refreshFlag” set when Device supports “UICC Reset”

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify ‘LSI Support’ is present in <ATR>

IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, “010203”, 2)	Verify <MEP_MODE> = ’01’, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		#ICCID_OP_PROF1,
		NO_PARAM,
		TRUE,
		1))
			#R_ENABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(1, “UICC Reset”)		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_FIRST_PROFILE		
4	PROC_MEP_LSI_MULTIPLEXING(0)		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

6	PROC_MEP_LSI_MULTIPLEXING(1)		
7	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

8	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #03 Nominal: Enable Profile by ISD-P AID and “refreshFlag” set when Device supports “eUICC Profile State Change”

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = '01', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		NO_PARAM,
		<ISD_P_AID1>,
		TRUE,
		1))
			#R_ENABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(1, "eUICC Profile State Change")		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP_EN_DS_FIRST_PROFILE		
4	PROC_MEP_LSI_MULTIPLEXING(0)		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

6	PROC_MEP_LSI_MULTIPLEXING(1)		
7	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

8	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #04 Nominal: Enable Profile by ICCID and "refreshFlag" set when Device supports "eUICC Profile State Change"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = '01', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		#ICCID_OP_PROF1,
		NO_PARAM,
		TRUE,
		1))
			#R_ENABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(1, "eUICC Profile State Change")		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP_EN_DS_FIRST_PROFILE		
4	PROC_MEP_LSI_MULTIPLEXING(0)		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

6	PROC_MEP_LSI_MULTIPLEXING(1)		
7	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

8	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #05 Nominal: Enable Profile by ISD-P AID and "refreshFlag" not set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> ='01', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		NO_PARAM,
		<ISD_P_AID1>,
		FALSE,
		1))
			#R_ENABLE_PROFILE_OK
			SW=0x9000

2	PROC_MEP_LSI_MULTIPLEXING(1)		
3	S_Device → eUICC	[TERMINAL_PROFILE_LSI_COMMAND]	Toolkit initialization THEN SW=0x9000
4	PROC_MEP_LSI_MULTIPLEXING(0)		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

6	PROC_MEP_LSI_MULTIPLEXING(1)		

7	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

8	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #06 Nominal: Enable Profile by ICCID and "refreshFlag" not set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = '01', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		#ICCID_OP_PROF1,
		NO_PARAM,
		FALSE,
		1))
			#R_ENABLE_PROFILE_OK
			SW=0x9000

2	PROC_MEP_LSI_MULTIPLEXING(1)		
3	S_Device → eUICC	[TERMINAL_PROFILE_LSI_COMMAND]	Toolkit initialization THEN SW=0x9000
4	PROC_MEP_LSI_MULTIPLEXING(0)		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

6	PROC_MEP_LSI_MULTIPLEXING(1)		
7	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

8	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #07 Nominal: Enable Profile by ISD-P AID and "refreshFlag" set while proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = '01', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(0)		
IC8	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		NO_PARAM,
		<ISD_P_AID2>,
			resp EnableProfileResponse ::=
			{
			enableResult ok
			}

		TRUE, 1))	SW=0x91YY
2	PROC_MEP_REFRESH_EN_DS(1, “UICC Reset”)		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_FIRST_PROFILE		
4	PROC_MEP_LSI_MULTIPLEXING(0)		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		<NO_PARAM>,
		<NO_PARAM>)
			response ProfileInfoListResponse ::= profileInfoListOk : {
			
			#PROFILE_INFO1_DISABLED, #PROFILE_INFO2_ENABLED
			}
			SW=0x9000

6	PROC_MEP_LSI_MULTIPLEXING(1)		
7	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

8	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

Test Sequence #08 Nominal: Enable Profile by ISD-P AID and “refreshFlag” not set while proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify ‘LSI Support’ is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP(2, #IUT_MEP_LSI_OPTIONS, “010203”,	Verify <MEP_MODE> =’01’, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify

		2)	<MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(0)		
IC8	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		NO_PARAM,
		<ISD_P_AID2>,
		FALSE,
		1))
			resp EnableProfileResponse ::= {
			enableResult ok
			}
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED,
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

3	PROC_MEP_LSI_MULTIPLEXING(1)		
4	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

5	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

Test Sequence #09 Nominal: Enable 2nd Profile by ISD-P AID and "refreshFlag" set when Device supports "UICC Reset"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		NO_PARAM,
		<ISD_P_AID2>,
		TRUE,
		2))
			#R_ENABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(<MEP_MODE>, 2, "UICC Reset")		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_SECOND_PROFILE		
4	PROC_MEP_LSI_MULTIPLEXING(0)		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

6	PROC_MEP_LSI_MULTIPLEXING(2)		
7	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

8	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

Test Sequence #10 Nominal: Enable 2nd Profile by ICCID and "refreshFlag" set when Device supports "UICC Reset"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1 .

eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to #ICCID_OP_PROF1
eUICC	The PROFILE_OPERATIONAL2 corresponds to #ICCID_OP_PROF2

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		#ICCID_OP_PROF2,
		NO_PARAM,
		TRUE,
		2))
			#R_ENABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(<MEP_MODE>, 2, "UICC Reset")		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_SECOND_PROFILE		
4	PROC_MEP_LSI_MULTIPLEXING(0)		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

6	PROC_MEP_LSI_MULTIPLEXING(2)		
7	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

8	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

Test Sequence #11 Nominal: Enable 2nd Profile by ISD-P AID and "refreshFlag" set when Device supports "eUICC Profile State Change"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		NO_PARAM,
		<ISD_P_AID2>,
		TRUE,
		2))
			#R_ENABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(<MEP_MODE>, 0, "eUICC Profile State Change")		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP _EN_DS_SECOND_PROFILE		
4	PROC_MEP_LSI_MULTIPLEXING(0)		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

6	PROC_MEP_LSI_MULTIPLEXING(2)		
7	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

8	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

Test Sequence #12 Nominal: Enable 2nd Profile by ICCID and "refreshFlag" set when Device supports "eUICC Profile State Change"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to #ICCID_OP_PROF1
eUICC	The PROFILE_OPERATIONAL2 corresponds to #ICCID_OP_PROF2

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		#ICCID_OP_PROF2,
		NO_PARAM,
		TRUE,
		2))
			#R_ENABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(<MEP_MODE>, 2, "eUICCProfileStateChange")		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP_EN_DS_SECOND_PROFILE		
4	PROC_MEP_LSI_MULTIPLEXING(0)		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

6	PROC_MEP_LSI_MULTIPLEXING(2)		
7	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

8	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

Test Sequence #13 Nominal: Enable 2nd Profile by ISD-P AID and "refreshFlag" not set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		NO_PARAM,
		<ISD_P_AID2>,
		FALSE,
		2))
			#R_ENABLE_PROFILE_OK
			SW=0x9000

2	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_SECOND_PROFILE		
3	PROC_MEP_LSI_MULTIPLEXING(0)		

4	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO (
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1;
			
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

5	PROC_MEP_LSI_MULTIPLEXING(2)		
6	S_Device → eUICC	[TERMINAL_PROFILE_LSI_COMMAND]	Toolkit initialization THEN SW=0x9000
7	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

8	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

Test Sequence #14 Nominal: Enable 2nd Profile by ICCID and "refreshFlag" not set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to #ICCID_OP_PROF1
eUICC	The PROFILE_OPERATIONAL2 corresponds to #ICCID_OP_PROF2

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		#ICCID_OP_PROF2,
		NO_PARAM,
		FALSE,
		2))
			#R_ENABLE_PROFILE_OK
			SW=0x9000

2	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_SECOND_PROFILE		
3	PROC_MEP_LSI_MULTIPLEXING(0)		

4	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

5	PROC_MEP_LSI_MULTIPLEXING(2)		
6	S_Device → eUICC	[TERMINAL_PROFILE_LSI_COMMAND]	Toolkit initialization THEN SW=0x9000
7	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

8	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

Test Sequence #15 Nominal: Enable 3rd Profile by ISD-P AID and "refreshFlag" set while proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 2.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 corresponds to <ISD_P_AID3>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR>

			Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(2)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(0)		
IC8	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		NO_PARAM,
		<ISD_P_AID3>,
		TRUE,
		2))
			resp EnableProfileResponse ::= {
			enableResult ok
			}
			SW=0x91YY

2	PROC_MEP_REFRESH_EN_DS(2, "UICC Reset")		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_SECOND_PROFILE		
4	PROC_MEP_LSI_MULTIPLEXING(0)		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO3_ENABLED;
			}
			SW=0x9000

6	PROC_MEP_LSI_MULTIPLEXING(2)		
7	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

8	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF3
			SW=0x9000

Test Sequence #16 Nominal: Enable 3rd Profile by ISD-P AID and "refreshFlag" set while proactive session is ongoing with Terminal Response outstanding – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 2.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 corresponds to <ISD_P_AID3>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK and waiting for Terminal Response
IC7	PROC_MEP_LSI_MULTIPLEXING(0)		
IC8	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		NO_PARAM,
		<ISD_P_AID3>,
		TRUE,
			resp EnableProfileResponse ::= {
			enableResult ok
			}
			SW=0x91YY

		1))	
2	S_Device →eUICC	FETCH 'YY'	LSI COMMAND (“Proactive Session Request”) Verify <LSI_COMMAND_ACTION> = “Proactive session request” and <LSI_NUMBER> = 1
3	PROC_MEP_LSI_MULTIPLEXING(<LSI_NUMBER>)		
4	S_Device →eUICC	STATUS command	SW=0x9000
5	S_Device →eUICC	TERMINAL RESPONSE	SW=0x91ZZ
6	S_Device →eUICC	FETCH 'ZZ'	REFRESH Command (“UICC Reset”)
7	S_Device → eUICC	[SELECT_MF]	FCP Template present SW=0x9000
8	S_Device → eUICC	[TERMINAL_CAPABILITY_LPAd]	SW=0x9000
9	S_Device → eUICC	[TERMINAL_PROFILE_LSI_COMMAND]	Toolkit initialization THEN SW=0x9000
10	PROC_MEP_LSI_MULTIPLEXING(0)		

11	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO3_ENABLED;
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

12	PROC_MEP_LSI_MULTIPLEXING(1)		
13	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

14	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF3
			SW=0x9000

Test Sequence #17 Nominal: Enable 3rd Profile by ISD-P AID and “refreshFlag” set while 2 proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 2.

eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 corresponds to <ISD_P_AID3>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(2)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(1)		
IC8	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91YY
IC9	Do not send FETCH command		
IC10	PROC_MEP_LSI_MULTIPLEXING(0)		
IC11	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → Euicc	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		NO_PARAM,
		<ISD_P_AID3>,
		TRUE,
		1))
			resp EnableProfileResponse ::= {
			enableResult ok
			}
			SW=0x91ZZ

2	PROC_MEP_REFRESH_EN_DS(1, "UICC Reset")		
3	PROC_MEP_LSI_MULTIPLEXING(2)		
4	S_Device →eUICC	FETCH 'XX'	SMS POR received

			SCP80 response status code equal to 0x00 – POR OK
5	S_Device →eUICC	TERMINAL RESPONSE	
6	PROC_MEP_LSI_MULTIPLEXING(0)		
7	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(<NO_PARAM>, <NO_PARAM>))	response ProfileInfoListResponse::= profileInfoListOk : { #PROFILE_INFO3_ENABLED; #PROFILE_INFO2_ENABLED; } SW=0x9000
8	PROC_MEP_LSI_MULTIPLEXING(1)		
9	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000
10	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF3 SW=0x9000

Test Sequence #18 Nominal: Enable 3rd Profile by ISD-P AID and "refreshFlag" set while 2 proactive session is ongoing with Terminal Response outstanding – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 2.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 corresponds to <ISD_P_AID3>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify

			<MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(2)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK and waiting for Terminal Response
IC7	PROC_MEP_LSI_MULTIPLEXING(1)		
IC8	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91YY
IC9	S_Device → eUICC	FETCH 'YY'	SMS POR received SCP80 response status code equal to 0x00 – POR OK and waiting for Terminal Response
IC10	PROC_MEP_LSI_MULTIPLEXING(0)		
IC11	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		NO_PARAM,
		<ISD_P_AID3>,
		TRUE,
		1))
			resp EnableProfileResponse ::= {
			enableResult ok
			}
			SW=0x91ZZ

2	S_Device →eUICC	FETCH 'ZZ'	LSI COMMAND ("ProactiveSessionREquest") Verify <LSI_COMMAND_ACTION> = "Proactive session request" and <LSI_NUMBER> = 1
3	PROC_MEP_LSI_MULTIPLEXING(<LSI_NUMBER>)		
4	S_Device →eUICC	STATUS command	SW=0x9000
5	S_Device →eUICC	TERMINAL RESPONSE	SW=0x91KK
6	S_Device →eUICC	FETCH 'KK'	REFRESH Command ("UICC Reset")
7	S_Device → eUICC	[SELECT_MF]	FCP Template present SW=0x9000
8	S_Device → eUICC	[TERMINAL_CAPABILITY_LPAd]	SW=0x9000

9	S_Device → eUICC	[TERMINAL_PROFILE_LSI_COMMAND]	Toolkit initialization THEN SW=0x9000
10	PROC_MEP_LSI_MULTIPLEXING(2)		
11	S_Device →eUICC	TERMINAL RESPONSE	SW=0x9000
12	S_Device	PROC_MEP_LSI_MULTIPLEXING(0)	
13	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(<NO_PARAM>, <NO_PARAM>))	response ProfileInfoListResponse::= profileInfoListOk : { #PROFILE_INFO3_ENABLED; #PROFILE_INFO2_ENABLED; } SW=0x9000
14	PROC_MEP_LSI_MULTIPLEXING(1)		
15	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000
16	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF3 SW=0x9000

4.2.21.2.9 TC_eUICC_ES10c.EnableProfile_ErrorCases_Case4_MEPA1

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	eUICC in MEP mode
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.

Test Sequence #01 Error: Enable Profile by an unknown ISD-P AID

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The Operational Profile identified by the ISD-P AID <ISD_P_AIDX> is not loaded.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (	Verify

		2, #IUT_MEP_LSI_OPTIONS, “010203”, 2)	<MEP_MODE> =’01’, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		NO_PARAM,
		<ISD_P_AIDX>,
		TRUE,
		1))
			#R_ENABLE_PROFILE_ICCID_ISDP_NOTFOUND
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

Test Sequence #02 Error: Enable Profile by an unknown ICCID

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The Operational Profile identified by the ICCID #ICCID_OP_PROFX is not loaded.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify ‘LSI Support’ is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, “010203”, 2)	Verify <MEP_MODE> = ’01’, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS

IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		#ICCID_OP_PROFX,
		NO_PARAM,
		TRUE,
		1))
			#R_ENABLE_PROFILE_ICCID_ISDP_NOTFOUND
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

Test Sequence #03 Error: Enable Profile (by ISD-P AID) is not possible when this Operational Profile is in Enable state

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> ='01', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		NO_PARAM,
			#R_ENABLE_PROFILE_NOT_DISABLE_STATE
			SW=0x9000

		<ISD_P_AID1>, TRUE, 1))	

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #04 Error: Enable Profile (by ICCID) is not possible when this Operational Profile is in Enabled state

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> ='01', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		#ICCID_OP_PROF1,
		NO_PARAM,
		TRUE,
		1))
			#R_ENABLE_PROFILE_NOT_DISABLE_STATE
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(#ICCID_OP_PROF1,	response ProfileInfoListResponse::= profileInfoListOk : { #PROFILE_INFO1

		NO_PARAM))	} SW=0x9000

Test Sequence #05 Error: Enable Profile by ISD-P AID without refreshFlag while proactive session is ongoing – catBusy supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> ='01', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(0)		
IC8	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		NO_PARAM,
		<ISD_P_AID2>,
		FALSE,
		1))
			resp EnableProfileResponse ::= {
			enableResult catBusy
			}
			SW=0x9000 or 0x91XX

2	PROC_MEP_LSI_MULTIPLEXING(1)		

3	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
4	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
5	PROC_MEP_LSI_MULTIPLEXING(0)		

6	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>)
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1,
			#PROFILE_INFO2
			}
			SW=0x9000

Test Sequence #06 Error: Enable Profile by ICCID with refreshFlag set while proactive session is ongoing – catBusy supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = '01', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(0)		

IC8	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		#ICCID_OP_PROF2,
		NO_PARAM,
		TRUE,
		1))
			resp EnableProfileResponse ::= {
			enableResult catBusy
			}
			SW=0x9000 or 0x91XX

2	PROC_MEP_LSI_MULTIPLEXING(1)		
3	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
4	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
5	PROC_MEP_LSI_MULTIPLEXING(0)		

6	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		<NO_PARAM>,
		<NO_PARAM>)
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1,
			#PROFILE_INFO2
			}
			SW=0x9000

Test Sequence #07 Error: Enable 2nd Profile by an unknown ISD-P AID where one profile is already enabled

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The Operational Profile identified by the ISD-P AID <ISD_P_AIDX> is not loaded.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203",	Verify <MEP_MODE> = 01, Verify

		2)	<MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		NO_PARAM,
		<ISD_P_AIDX>,
		TRUE,
		2))
			#R_ENABLE_PROFILE_ICCID_ISDP_NOTFOUND
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1;
			}
			SW=0x9000

Test Sequence #08 Error: Enable 2nd Profile by an unknown ICCID where one profile is already enabled

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to #ICCID_OP_PROF1.
eUICC	The Operational Profile identified by the ICCID #ICCID_OP_PROFX is not loaded.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS

IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		#ICCID_OP_PROFX,
		NO_PARAM,
		TRUE,
		2))
			#R_ENABLE_PROFILE_ICCID_ISDP_NOTFOUND
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1;
			}
			SW=0x9000

Test Sequence #09 Error: Enable 2nd Profile (by ISD-P AID) is not possible when this Operational Profile is in Enable state at other LSI

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 2.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		NO_PARAM,
		<ISD_P_AID1>,
		TRUE,
		2))
			#R_ENABLE_PROFILE_NOT_DISABLE_STATE
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		NO_PARAM,
		<ISD_P_AID2>,
		TRUE,
		1))
			#R_ENABLE_PROFILE_NOT_DISABLE_STATE
			SW=0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

Test Sequence #10 Error: Enable 2nd Profile (by ICCID) is not possible when this Operational Profile is in Enabled state at other LSI

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to #ICCID_OP_PROF1
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 2.
eUICC	The PROFILE_OPERATIONAL2 corresponds to #ICCID_OP_PROF2

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		#ICCID_OP_PROF1,
		NO_PARAM,
		TRUE,
		2))
			#R_ENABLE_PROFILE_NOT_DISABLE_STATE
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		#ICCID_OP_PROF2,
		NO_PARAM,
		TRUE,
		1))
			#R_ENABLE_PROFILE_NOT_DISABLE_STATE
			SW=0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF2,
		NO_PARAM))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

Test Sequence #11 Error: Enable 2nd Profile by ISD-P AID and "refreshFlag" set while 2 proactive session is ongoing – catBusy supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 2.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 corresponds to <ISD_P_AID3>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(2)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(1)		
IC8	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91YY
IC9	Do not send FETCH command		
IC10	PROC_MEP_LSI_MULTIPLEXING(0)		
IC11	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		NO_PARAM,
		<ISD_P_AID3>,
			resp EnableProfileResponse ::=
			{
			enableResult catBusy
			}

		TRUE, 1))	SW= 9000
2	Repeat IC7		
3	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
4	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
5	Repeat IC4		
6	S_Device → eUICC	FETCH 'YY'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
7	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
8	Repeat IC10		

9	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

Test Sequence #12 Error: Enable 2nd Profile (by ISD-P AID) is not possible when CAT is not initialized for targetPort

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS,	Verify <MEP_MODE> = 01, Verify

		“010203”, 2)	<MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_ONE_LSI_FOR_ENABLED_PROFILE		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE_MEP_A1(
		NO_PARAM,
		<ISD_P_AID3>,
		TRUE,
		2))
			resp EnableProfileResponse ::= {
			enableResult commandError
			}
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1,
			#PROFILE_INFO2
			}
			SW=0x9000

3	PROC_MEP_LSI_MULTIPLEXING(1)		
4	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

5	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

4.2.21.2.10 TC_eUICC_ES10c.EnableProfile_Case4_MEPA2

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	eUICC in MEP mode
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.

Test Sequence #01 Nominal: Enable Profile by ISD-P AID and “refreshFlag” not set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify

			'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> ='02', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		FALSE))
			#R_ENABLE_PROFILE_MEP_A2_OK
			SW=0x9000
			Verify < MEP-A2_TARGET_ESIM_PORT> = 1 OR 2

2	PROC_MEP_LSI_MULTIPLEXING(<MEP-A2_TARGET_ESIM_PORT>)		
3	S_Device → eUICC	[TERMINAL_PROFILE_LSI_COMMAND]	Toolkit initialization THEN SW=0x9000
4	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

5	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

6	PROC_MEP_LSI_MULTIPLEXING(0)		

7	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM
		))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #02 Nominal: Enable Profile by ICCID and "refreshFlag" not set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify

			'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = '02', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		#ICCID_OP_PROF1,
		NO_PARAM,
		FALSE))
			#R_ENABLE_PROFILE_MEP_A2_OK
			SW=0x9000
			Verify < MEP-A2_TARGET_ESIM_PORT> = 1 OR 2

2	PROC_MEP_LSI_MULTIPLEXING(<MEP-A2_TARGET_ESIM_PORT>)		
3	S_Device → eUICC	[TERMINAL_PROFILE_LSI_COMMAND]	Toolkit initialization THEN SW=0x9000
4	PROC_MEP_LSI_MULTIPLEXING(0)		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

6	PROC_MEP_LSI_MULTIPLEXING(<MEP-A2_TARGET_ESIM_PORT>)		
7	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

8	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #03 Nominal: Enable 2nd Profile by ISD-P AID and "refreshFlag" not set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.

eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID2>,
		FALSE))
			#R_ENABLE_PROFILE_MEP_A2_OK
			SW=0x9000
			
			Verify <MEP-A2_TARGET_ESIM_PORT> = 1 OR 2

2	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_SECOND_PROFILE		
3	PROC_MEP_LSI_MULTIPLEXING(0)		

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

5	PROC_MEP_LSI_MULTIPLEXING(<MEP-A2_TARGET_ESIM_PORT>)		
6	S_Device → eUICC	[TERMINAL_PROFILE_LSI_COMMAND]	Toolkit initialization THEN SW=0x9000
7	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

8	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

Test Sequence #04 Nominal: Enable 2nd Profile by ICCID and "refreshFlag" not set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to #ICCID_OP_PROF1
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to #ICCID_OP_PROF2

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		#ICCID_OP_PROF2,
		NO_PARAM,
		FALSE))
			#R_ENABLE_PROFILE_MEP_A2_OK
			SW=0x9000
			
			Verify <MEP-A2_TARGET_ESIM_PORT> =1 OR 2

2	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_SECOND_PROFILE		
3	PROC_MEP_LSI_MULTIPLEXING(0)		

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

5	PROC_MEP_LSI_MULTIPLEXING(<MEP-A2_TARGET_ESIM_PORT>)		
6	S_Device → eUICC	[TERMINAL_PROFILE_LSI_COMMAND]	Toolkit initialization THEN SW=0x9000

7	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

8	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

4.2.21.2.11 TC_eUICC_ES10c.EnableProfile_ErrorCases_Case4_MEPA2

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	eUICC in MEP mode
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.

Test Sequence #01 Error: Enable Profile by an unknown ISD-P AID

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The Operational Profile identified by the ISD-P AID <ISD_P_AIDX> is not loaded.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = '02', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE (
		NO_PARAM,
		<ISD_P_AIDX>,
		FALSE))
			#R_ENABLE_PROFILE_ICCID_ISDP_NOTFOUND
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

Test Sequence #02 Error: Enable Profile by an unknown ICCID

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The Operational Profile identified by the ICCID #ICCID_OP_PROFX is not loaded.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = '02', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		#ICCID_OP_PROFX,
		NO_PARAM,
		FALSE))
			#R_ENABLE_PROFILE_ICCID_ISDP_NOTFOUND
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

Test Sequence #03 Error: Enable Profile (by ISD-P AID) is not possible when this Operational Profile is in Enabled state

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC as MEPA2 mode.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = '02', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		FALSE))
			#R_ENABLE_PROFILE_NOT_DISABLE_STATE
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #04 Error: Enable Profile (by ICCID) is not possible when this Operational Profile is in Enabled state

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC as MEPA2 mode.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = '02', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE (
		#ICCID_OP_PROF1,
		NO_PARAM,
		FALSE))
			#R_ENABLE_PROFILE_NOT_DISABLE_STATE
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #05 Error: Enable Profile by ISD-P AID without refreshFlag while proactive session is ongoing – catBusy supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC as MEPA2 mode.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = '02', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(<MEP-A2_TARGET_ESIM_PORT>)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(0)		
IC8	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID2>,
		FALSE))
			resp EnableProfileResponse ::= {
			enableResult catBusy
			}
			SW=0x9000 or 0x91XX

2	PROC_MEP_LSI_MULTIPLEXING(<MEP-A2_TARGET_ESIM_PORT>)		
3	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
4	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
5	PROC_MEP_LSI_MULTIPLEXING(0)		

6	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1,
			#PROFILE_INFO2
			}
			SW=0x9000

Test Sequence #06 Error: Enable 2nd Profile by an unknown ISD-P AID where one profile is already enabled

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC as MEPA2 mode.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The Operational Profile identified by the ISD-P AID <ISD_P_AIDX> is not loaded.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE (
		NO_PARAM,
		<ISD_P_AIDX>,
		FALSE))
			#R_ENABLE_PROFILE_ICCID_ISDP_NOTFOUND
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1;
			}
			SW=0x9000

Test Sequence #07 Error: Enable 2nd Profile by an unknown ICCID where one profile is already enabled

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC as MEPA2 mode.
eUICC	The PROFILE_OPERATIONAL1 corresponds to #ICCID_OP_PROF1.
eUICC	The Operational Profile identified by the ICCID #ICCID_OP_PROFX is not loaded.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE (
		#ICCID_OP_PROFX,
		NO_PARAM,
		FALSE))
			#R_ENABLE_PROFILE_ICCID_ISDP_NOTFOUND
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			}
			SW=0x9000

Test Sequence #08 Error: Enable 2nd Profile (by ISD-P AID) is not possible when this Operational Profile is in Enable state at other LSI

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port X.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port Y.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE (
		NO_PARAM,
		<ISD_P_AID1>,
		FALSE))
			#R_ENABLE_PROFILE_NOT_DISABLE_STATE
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE (
		NO_PARAM,
			#R_ENABLE_PROFILE_NOT_DISABLE_STATE
			SW=0x9000

		<ISD_P_AID2>, FALSE, 1))	

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

Test Sequence #09 Error: Enable 2nd Profile (by ICCID) is not possible when this Operational Profile is in Enabled state at other LSI

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port X.
eUICC	The PROFILE_OPERATIONAL1 corresponds to #ICCID_OP_PROF1
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port Y.
eUICC	The PROFILE_OPERATIONAL2 corresponds to #ICCID_OP_PROF2

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		#ICCID_OP_PROF1,
		NO_PARAM,
			#R_ENABLE_PROFILE_NOT_DISABLE_STATE
			SW=0x9000

		FALSE, 2))	

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE (
		#ICCID_OP_PROF2,
		NO_PARAM,
		FALSE,
		1))
			#R_ENABLE_PROFILE_NOT_DISABLE_STATE
			SW=0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF2,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

Test Sequence #10 Error: Enable 2nd Profile by ISD-P AID and "refreshFlag" not set while 2 proactive session is ongoing – catBusy supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port X.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port Y.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 corresponds to <ISD_P_AID3>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>

IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, “020103”, 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(Y)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(X)		
IC8	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91YY
IC9	Do not send FETCH command		
IC10	PROC_MEP_LSI_MULTIPLEXING(0)		
IC11	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID3>,
		FALSE,
		1))
			resp EnableProfileResponse ::= {
			enableResult catBusy
			}
			SW= 9000

2	Repeat IC7		
3	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
4	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
5	Repeat IC4		
6	S_Device → eUICC	FETCH 'YY'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
7	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
8	Repeat IC10		
9	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO (<NO_PARAM>,	response ProfileInfoListResponse::= profileInfoListOk : { #PROFILE_INFO1;

		<NO_PARAM>))	#PROFILE_INFO2_ENABLED; } SW=0x9000

4.2.21.2.12 TC_eUICC_ES10c.EnableProfile_Case4_MEPB

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	eUICC in MEP mode
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.

Test Sequence #01 Nominal: Enable 1st Profile by ISD-P AID and "refreshFlag" set when Device supports "UICC Reset"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = '03', <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE (
		NO_PARAM,
		<ISD_P_AID1>,
		TRUE)
		)
			#R_ENABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(NO_PARAM, "UICC Reset")		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_FIRST_PROFILE		
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

6	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

7	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #02 Nominal: Enable Profile by ICCID and "refreshFlag" set when Device supports "UICC Reset"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = '03', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE (
		#ICCID_OP_PROF1,
		NO_PARAM,
		TRUE))
			#R_ENABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(NO_PARAM, "UICC Reset")		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_FIRST_PROFILE		

4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

6	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

7	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #03 Nominal: Enable Profile by ISD-P AID and "refreshFlag" set when Device supports "eUICC Profile State Change"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = '03', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		TRUE))
			#R_ENABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(<MEP_MODE>,		

	0, “eUICC Profile State change”)		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP_EN_DS_FIRST_PROFILE		
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

6	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

7	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #04 Nominal: Enable Profile by ICCID and “refreshFlag” set when Device supports “eUICC Profile State Change”

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify ‘LSI Support’ is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, “030201”, 2)	Verify <MEP_MODE> = ‘03’, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE (
		#ICCID_OP_PROF1,
			#R_ENABLE_PROFILE_OK
			SW=0x91XX

		NO_PARAM, TRUE))	
2	PROC_MEP_REFRESH_EN_DS(<MEP_MODE>, 0, “eUICC Profile State Change”)		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP_EN_DS_FIRST_PROFILE		
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

6	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

7	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #05 Nominal: Enable Profile by ISD-P AID and “refreshFlag” not set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify ‘LSI Support’ is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, “030201”, 2)	Verify <MEP_MODE> = ‘03’, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(	#R_ENABLE_PROFILE_OK

		MTD_ENABLE_PROFILE (NO_PARAM, <ISD_P_AID1>, FALSE))	SW=0x9000
2	S_Device → eUICC	[TERMINAL_PROFILE_LSI_COMMAND]	Toolkit initialization THEN SW=0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

4	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

5	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #06 Nominal: Enable Profile by ICCID and "refreshFlag" not set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = '03', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE (
		#ICCID_OP_PROF1,
			#R_ENABLE_PROFILE_OK
			SW=0x9000

		NO_PARAM, FALSE))	
2	S_Device → eUICC	[TERMINAL_PROFILE_LSI_COMMAND]	Toolkit initialization THEN SW=0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

4	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

5	S_Device → Euicc	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #07 Nominal: Enable Profile by ISD-P AID and "refreshFlag" set whileproactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = '03', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC5	Do not send FETCH command		

IC6	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE (
		NO_PARAM,
		<ISD_P_AID2>,
		TRUE))
			resp EnableProfileResponse ::=
			{
			enableResult ok
			}
			SW=0x91YY

2	PROC_MEP_REFRESH_EN_DS(<MEP_MODE>, 0, “UICC Reset”)		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_FIRST_PROFILE		

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response
			ProfileInfoListResponse ::=
			profileInfoListOk : {
			
			#PROFILE_INFO1_DISABLED,
			#PROFILE_INFO2_ENABLED
			}

5	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

6	S_Device → Euicc	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

Test Sequence #08 Nominal: Enable Profile by ISD-P AID and “refreshFlag” not set while proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify ‘LSI Support’ is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2,	Verify <MEP_MODE> = ‘03’, Verify

		#IUT_MEP_LSI_OPTIONS, “030201”, 2)	<MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC5	Do not send FETCH command		
IC6	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE (
		NO_PARAM,
		<ISD_P_AID2>,
		FALSE))
			resp EnableProfileResponse ::= {
			enableResult ok
			}
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED,
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

3	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

4	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

Test Sequence #09 Nominal: Enable 2nd Profile by ISD-P AID and “refreshFlag” set when Device supports “UICC Reset”

The purpose of this sequence is to make sure that the MEP-capable eUICC can have two Profiles Enabled on two different LSIs.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC5	PROC_MEP_LSI_MULTIPLEXING(1)		
IC6	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE (
		NO_PARAM,
		<ISD_P_AID2>,
		TRUE))
			#R_ENABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(<MEP_MODE>, 1, "UICC Reset")		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_SECOND_PROFILE		
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_MEPB;
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

6	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

7	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

Test Sequence #10 Nominal: Enable 2nd Profile by ICCID and "refreshFlag" set when Device supports "UICC Reset"

The purpose of this test sequence is to make sure that the MEP-capable eUICC can have two Profiles Enabled on two different LSIs.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to #ICCID_OP_PROF1
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to #ICCID_OP_PROF2

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE (
		#ICCID_OP_PROF2,
		NO_PARAM,
		TRUE))
			#R_ENABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(1, "UICC Reset")		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_SECOND_PROFILE		
4	PROC_MEP_LSI_MULTIPLEXING(0)		
5	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

6	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			
			#PROFILE_INFO2_ENABLED_MEPB;
			}
			SW=0x9000

7	PROC_MEP_LSI_MULTIPLEXING(1)		

8	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

9	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

Test Sequence #11 Nominal: Enable 2nd Profile by ISD-P AID and "refreshFlag" set when Device supports "eUICC Profile State Change"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE (
		NO_PARAM,
		<ISD_P_AID2>,
		TRUE))
			#R_ENABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(1, "eUICC Profile State Change")		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP _EN_DS_SECOND_PROFILE		
4	PROC_MEP_LSI_MULTIPLEXING(0)		
5	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
6	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO (	response ProfileInfoListResponse::= profileInfoListOk : {

		<NO_PARAM>, <NO_PARAM>))	#PROFILE_INFO1; #PROFILE_INFO2_ENABLED_MEPB; } SW=0x9000
7	PROC_MEP_LSI_MULTIPLEXING(2)		
8	S_Device → eUICC	[SELECT_ICCID]	 SW=0x9000

9	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

Test Sequence #12 Nominal: Enable 2nd Profile by ICCID and "refreshFlag" set when Device supports "eUICC Profile State Change"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to #ICCID_OP_PROF1
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to #ICCID_OP_PROF2

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE (
		#ICCID_OP_PROF2,
		NO_PARAM,
		TRUE,
		<MEP_MODE>,
		1,
		NO_PARAM))
			#R_ENABLE_PROFILE_OK
			SW=0x91XX

2	S_Device → eUICC	PROC_MEP_REFRESH_EN_DS(1, “eUICCProfileStateChange”)	REFRESH Command (“eUICC Profile State Change”)
3	PROC_EUICC_INITIALIZATION_SEQUENCE eUICCProfileStateChanged_MEP _MEP_EN_DS_SECOND_PROFILE		
4	PROC_MEP_LSI_MULTIPLEXING(0)		
5	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

6	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		<NO_PARAM>,
		<NO_PARAM>))
			response
			ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_MEP;
			#PROFILE_INFO2_MEPB;
			}
			SW=0x9000

7	PROC_MEP_LSI_MULTIPLEXING(1)		
8	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

9	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

Test Sequence #13 Nominal: Enable 2nd Profile by ISD-P AID and “refreshFlag” not set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify ‘LSI Support’ is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, “030201”, 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS

IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID2>,
		FALSE))
			#R_ENABLE_PROFILE_OK
			SW=0x9000

2	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_SECOND_PROFILE		
3	PROC_MEP_LSI_MULTIPLEXING(0)		
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response
			ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2_MEPB;
			}
			SW=0x9000

6	PROC_MEP_LSI_MULTIPLEXING(1)		
7	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

8	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

Test Sequence #14 Nominal: Enable 2nd Profile by ICCID and "refreshFlag" not set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to #ICCID_OP_PROF1
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to #ICCID_OP_PROF2

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>

IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, “030201”, 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		#ICCID_OP_PROF2,
		NO_PARAM,
		FALSE))
			#R_ENABLE_PROFILE_OK
			SW=0x9000

2	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_SECOND_PROFILE		
3	PROC_MEP_LSI_MULTIPLEXING(0)		
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2_ENABLED_MEPB;
			}
			SW=0x9000

6	PROC_MEP_LSI_MULTIPLEXING(1)		
7	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

8	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

Test Sequence #15 Nominal: Enable 3rd Profile by ISD-P AID and “refreshFlag” set while proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 corresponds to <ISD_P_AID3>.
eUICC	The Nickname of the PROFILE_OPERATIONAL3 is equal to #NICKNAME3.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID3>,
		TRUE))
			resp EnableProfileResponse ::= {
			enableResult ok
			}
			SW=0x91YY

2	PROC_MEP_REFRESH_EN_DS(1, UICC Reset)		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_SECOND_PROFILE		
4	PROC_MEP_LSI_MULTIPLEXING(0)		
5	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
6	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(<NO_PARAM>,	response ProfileInfoListResponse::= profileInfoListOk : {

		<NO_PARAM>))	#PROFILE_INFO1; #PROFILE_INFO2; #PROFILE_INFO3_MEPB; } SW=0x9000
7	PROC_MEP_LSI_MULTIPLEXING(1)		
8	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

9	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF3
			SW=0x9000

Test Sequence #16 Nominal: Enable 3rd Profile by ISD-P AID and "refreshFlag" set while 2 proactive session is ongoing – catBusy not supported

The purpose of this test sequence is to verify that profile switching of one target port (here Port 1) does not impact a proactive command on another eSIM Port (here Port 0).

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 corresponds to <ISD_P_AID3>.
eUICC	The Nickname of the PROFILE_OPERATIONAL3 is equal to #NICKNAME3.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify

			<MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(0)		
IC8	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91YY
IC9	Do not send FETCH command		
IC10	PROC_MEP_LSI_MULTIPLEXING(1)		
IC11	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_ENABLE_PROFILE(NO_PARAM, <ISD_P_AID3>, TRUE))	resp EnableProfileResponse ::= { enableResult ok } SW=0x91ZZ
2	PROC_MEP_REFRESH_EN_DS(1, “UICC Reset”)		
3	PROC_MEP_LSI_MULTIPLEXING(0)		
4	S_Device →eUICC	FETCH 'YY'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
5	S_Device →eUICC	TERMINAL RESPONSE	
6	PROC_MEP_LSI_MULTIPLEXING(0)		
7	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
8	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(<NO_PARAM>, <NO_PARAM>))	response ProfileInfoListResponse::= profileInfoListOk : { #PROFILE_INFO1; #PROFILE_INFO2; #PROFILE_INFO3_MEPB } SW=0x9000
9	PROC_MEP_LSI_MULTIPLEXING(1)		
10	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

11	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF3
			SW=0x9000

4.2.21.2.13 TC_eUICC_ES10c.EnableProfile_ErrorCases_Case4_MEPB

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	eUICC in MEP mode
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.

Test Sequence #01 Error: Enable Profile by an unknown ISD-P AID

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The Operational Profile identified by the ISD-P AID <ISD_P_AIDX> is not loaded.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = '03', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AIDX>,
		TRUE))
			#R_ENABLE_PROFILE_ICCID_ISDP_NOTFOUND
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

Test Sequence #02 Error: Enable Profile by an unknown ICCID

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The Operational Profile identified by the ICCID #ICCID_OP_PROFX is not loaded.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = '03', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		#ICCID_OP_PROFX,
		NO_PARAM,
		TRUE))
			#R_ENABLE_PROFILE_ICCID_ISDP_NOTFOUND
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

Test Sequence #03 Error: Enable Profile (by ISD-P AID) is not possible when this Operational Profile is in Enabled state

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = '03', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		TRUE))
			#R_ENABLE_PROFILE_NOT_DISABLE_STATE
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #04 Error: Enable Profile (by ICCID) is not possible when this Operational Profile is in Enabled state

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = '03', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		#ICCID_OP_PROF1,
		NO_PARAM,
		TRUE))
			#R_ENABLE_PROFILE_NOT_DISABLE_STATE
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #05 Error: Enable Profile by ISD-P AID with refreshFlag not set while proactive session is ongoing – catBusy supported

The purpose of this test sequence is to verify that if a proactive session is still ongoing on the target Port and catBusy is supported, the Enable command is rejected.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = '03', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC5	Do not send FETCH command		
IC6	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE (
		NO_PARAM,
		<ISD_P_AID2>,
		FALSE))
			resp EnableProfileResponse ::= {
			enableResult catBusy
			}
			SW=0x9000 or 0x91XX

2	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000

4	S_LPAd → Euicc	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1,
			#PROFILE_INFO2
			}
			SW=0x9000

Test Sequence #06 Error: Enable Profile by ICCID with refreshFlag set while proactive session is ongoing – catBusy supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the Euicc on Port 0.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.

eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = '03', Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC5	Do not send FETCH command		
IC6	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		#ICCID_OP_PROF2,
		NO_PARAM,
		TRUE))
			resp EnableProfileResponse ::=
			{
			enableResult catBusy
			}
			SW=0x9000 or 0x91XX

2	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1,
			#PROFILE_INFO2
			}
			SW=0x9000

Test Sequence #07 Error: Enable 2nd Profile by an unknown ISD-P AID where one profile is already enabled

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The Operational Profile identified by the ISD-P AID <ISD_P_AIDX> is not loaded.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC5	PROC_MEP_LSI_MULTIPLEXING(1)		
IC6	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AIDX>,
		TRUE))
			#R_ENABLE_PROFILE_ICCID_ISDP_NOTFOUND
			SW=0x9000

2	PROC_MEP_LSI_MULTIPLEXING(0)		

3	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			}
			SW=0x9000

Test Sequence #08 Error: Enable 2nd Profile by an unknown ICCID where one profile is already enabled

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to #ICCID_OP_PROF1.
eUICC	The Operational Profile identified by the ICCID #ICCID_OP_PROFX is not loaded.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC5	PROC_MEP_LSI_MULTIPLEXING(1)		
IC6	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		#ICCID_OP_PROFX,
		NO_PARAM,
		TRUE))
			#R_ENABLE_PROFILE_ICCID_ISDP_NOTFOUND
			SW=0x9000

2	PROC_MEP_LSI_MULTIPLEXING(0)		

3	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			}
			SW=0x9000

Test Sequence #09 Error: Enable 2nd Profile (by ISD-P AID) is not possible when this Operational Profile is in Enabled state at other LSI

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID2>,
		TRUE))
			#R_ENABLE_PROFILE_NOT_DISABLE_STATE
			SW=0x9000

2	PROC_MEP_LSI_MULTIPLEXING(1)		
3	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_MEPB;
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		TRUE))
			#R_ENABLE_PROFILE_NOT_DISABLE_STATE
			SW=0x9000

6	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_MEPB;
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

Test Sequence #10 Error: Enable 2nd Profile (by ICCID) is not possible when this Operational Profile is in Enabled state at other LSI

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to #ICCID_OP_PROF1
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 corresponds to #ICCID_OP_PROF2

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		#ICCID_OP_PROF2,
		NO_PARAM,
		TRUE))
			#R_ENABLE_PROFILE_NOT_DISABLE_STATE
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

3	PROC_MEP_LSI_MULTIPLEXING(1)		
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		#ICCID_OP_PROF1,
		NO_PARAM,
		TRUE))
			#R_ENABLE_PROFILE_NOT_DISABLE_STATE
			SW=0x9000

6	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF2,
		NO_PARAM))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

Test Sequence #11 Error: Enable 2nd Profile by ISD-P AID and "refreshFlag" set while 2 proactive sessions are ongoing – catBusy supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.
eUICC	The PROFILE_OPERATIONAL3 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 corresponds to <ISD_P_AID3>.

eUICC	The Nickname of the PROFILE_OPERATIONAL3 is equal to #NICKNAME3.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(0)		
IC8	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91YY
IC9	Do not send FETCH command		
IC10	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID3>,
		TRUE))
			resp EnableProfileResponse ::=
			{
			enableResult catBusy
			}
			SW= 9000

2	Repeat IC4		
3	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
4	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
5	Repeat IC7		
6	S_Device → eUICC	FETCH 'YY'	SMS POR received SCP80 response status code equal to 0x00 – POR OK

7	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
8	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(<NO_PARAM>, <NO_PARAM>))	response ProfileInfoListResponse::= profileInfoListOk : { #PROFILE_INFO1; #PROFILE_INFO2_MEPB; #PROFILE_INFO3 } SW=0x9000
9	PROC_MEP_LSI_MULTIPLEXING(1)		
10	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
11	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(<NO_PARAM>, <NO_PARAM>))	response ProfileInfoListResponse::= profileInfoListOk : { #PROFILE_INFO1_MEPB; #PROFILE_INFO2_ENABLED; #PROFILE_INFO3 } SW=0x9000

Test Sequence #12 Error: Enable 2nd Profile (by ISD-P AID) is not possible when CAT is not initialized for targetPort

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP(2, #IUT_MEP_LSI_OPTIONS, "030201",	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS,

		2)	Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_ONE_LSI_FOR_ENABLED_PROFILE		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC5	PROC_MEP_LSI_MULTIPLEXING(1)		
IC6	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID2>,
		TRUE))
			resp EnableProfileResponse ::= {
			enableResult commandError
			}
			SW=0x9000

2	PROC_MEP_LSI_MULTIPLEXING(0)		

3	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse ::= profileInfoListOk : {
			#PROFILE_INFO1,
			#PROFILE_INFO2
			}
			SW=0x9000

4	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

5	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

4.2.22 ES10c (LPA – eUICC): DisableProfile

4.2.22.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 2.4.5
- Section 2.9.1, 2.9.3.3
- Section 3.2.2
- Section 5.7.17

4.2.22.2 Test Cases

4.2.22.2.1 TC_eUICC_ES10c.DisableProfile_Case3

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.

Test Sequence #01 Nominal: Disable Profile by ISD-P AID and "refreshFlag" set when Device supports "UICC Reset"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_DISABLE_PROFILE(NO_PARAM, <ISD_P_AID1>, TRUE))	No response data is returned SW=0x91XX
2	S_Device → eUICC	FETCH 'XX'	REFRESH Command ("UICC Reset")
3	Repeat IC1 and IC2		

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>)**)**
			response
			ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

5	S_Device → eUICC	[SELECT_ICCID]	SW=6A82

Test Sequence #02 Nominal: Disable Profile by ICCID and "refreshFlag" set when Device supports "UICC Reset"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_DISABLE_PROFILE(#ICCID_OP_PROF1, NO_PARAM, TRUE))	No response data is returned SW=0x91XX
2	S_Device → eUICC	FETCH 'XX'	REFRESH Command ("UICC Reset")
3	Repeat IC1 and IC2		

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

5	S_Device → eUICC	[SELECT_ICCID]	SW=6A82

Test Sequence #03 Nominal: Disable Profile by ISD-P AID and "refreshFlag" set when Device supports "eUICC Profile State Change"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_DISABLE_PROFILE(NO_PARAM, <ISD_P_AID1>, TRUE))	No response data is returned SW=0x91XX
2	S_Device → eUICC	FETCH 'XX'	REFRESH Command ("eUICC Profile State changed")

3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
4	Execute IC1 from step 2 to step 4		
5	Repeat IC2		

6	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

7	S_Device → eUICC	[SELECT_ICCID]	SW=6A82

Test Sequence #04 Nominal: Disable Profile by ICCID and "refreshFlag" set when Device supports "eUICC Profile State Change"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_DISABLE_PROFILE(#ICCID_OP_PROF1, NO_PARAM, TRUE))	No response data is returned SW=0x91XX
2	S_Device → eUICC	FETCH 'XX'	REFRESH Command ("eUICC Profile State changed")
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
4	Execute IC1 from step 2 to step 4		
5	Repeat IC2		

6	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

7	S_Device → eUICC	[SELECT_ICCID]	SW=6A82

Test Sequence #05 Nominal: Disable Profile by ISD-P AID and "refreshFlag" no set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_DISABLE_PROFILE(NO_PARAM, <ISD_P_AID1>, FALSE))	No response data is returned SW=0x9000
2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(NO_PARAM, <ISD_P_AID1>))	response ProfileInfoListResponse::= profileInfoListOk : { #PROFILE_INFO1_DISABLED } SW=0x9000
3	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #06 Nominal: Disable Profile by ICCID and "refreshFlag" no set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_DISABLE_PROFILE(	No response data is returned SW=0x9000

		#ICCID_OP_PROF1, NO_PARAM, FALSE))	

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

3	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #07 Nominal: Disable Profile by ICCID with refreshFLag set while proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC4	Do not send FETCH command		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_DISABLE_PROFILE(#ICCID_OP_PROF1, NO_PARAM, TRUE))	No response data is returned SW=0x91YY
2	S_Device →eUICC	FETCH ‘YY’	REFRESH Command (“UICC Reset”)
3	Repeat IC1 and IC2		

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse ::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

5	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #08 Nominal: Disable Profile by ICCID with refreshFLag not set while proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC4	Do not send FETCH command		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_DISABLE_PROFILE(#ICCID_OP_PROF1, NO_PARAM, FALSE))	No response data is returned SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(#ICCID_OP_PROF1, NO_PARAM))	response ProfileInfoListResponse ::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

3	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #09 Nominal: Disable Profile by ICCID with refreshFLag set while proactive session is ongoing with Terminal Response outstanding – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
1	S_Device →eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
2	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_DISABLE_PROFILE(#ICCID_OP_PROF1, NO_PARAM, TRUE))	No response data is returned SW=0x9000
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x91YY
4	S_Device →eUICC	FETCH 'YY'	REFRESH Command ("UICC Reset")
5	Repeat IC1 and IC2		

6	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(#ICCID_OP_PROF1, NO_PARAM))	response ProfileInfoListResponse ::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

7	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #10 Nominal: Disable Profile by ICCID with refreshFLag not set while proactive session is ongoing with Terminal Response outstanding – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX

1	S_Device →eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
2	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_DISABLE_PROFILE(#ICCID_OP_PROF1, NO_PARAM, FALSE))	No response data is returned SW=0x9000
3	S_Device →eUICC	TERMINAL RESPONSE	SW= any value except 91XX

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse ::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

5	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

4.2.22.2.2 TC_eUICC_ES10c.DisableProfile_ErrorCases_Case3

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.

Test Sequence #01 Error: Disable Profile by an unknown ISD-P AID

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The Operational Profile identified by the ISD-P AID <ISD_P_AIDX> is not loaded.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_DISABLE_PROFILE(NO_PARAM,	SW=0x6A82

		<ISD_P_AIDX>, TRUE))	

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #02 Error: Disable Profile by an unknown ICCID

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The Operational Profile identified by the ICCID #ICCID_OP_PROFX is not loaded.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_DISABLE_PROFILE(#ICCID_OP_PROFX, NO_PARAM, TRUE))	SW=0x6A82

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #03 Error: Disable Profile (by ISD-P AID) is not possible when this Operational Profile is in Disabled state

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_DISABLE_PROFILE(NO_PARAM, <ISD_P_AID1>, TRUE))	SW=0x6985

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

Test Sequence #04 Error: Disable Profile (by ICCID) is not possible when this Operational Profile is in Disabled state

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_DISABLE_PROFILE(#ICCID_OP_PROF1, NO_PARAM, TRUE))	SW=0x6985

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

Test Sequence #05 Error: Disable Profile (by ISD-P AID) not possible when PPR1 is set

The purpose of this test is to ensure that it is NOT possible to disable an Operational Profile4 with the Policy Rule “Disabling of this Profile is not allowed”.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is not loaded. (this condition overrides the general initial condition defined in this test case)
eUICC	The PROFILE_OPERATIONAL4 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL4 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL4 corresponds to <ISD_P_AID4>.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_DISABLE_PROFILE(NO_PARAM, <ISD_P_AID4>, TRUE))	SW=0x6985

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID4>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO4_ENABLED
			}
			
			SW=0x9000

Test Sequence #06 Error: Disable Profile (by ICCID) not possible when PPR1 is set

The purpose of this test is to ensure that it is NOT possible to disable an Operational Profile4 with the Policy Rule “Disabling of this Profile is not allowed”.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is not loaded. (this condition overrides the general initial condition defined in this test case)
eUICC	The PROFILE_OPERATIONAL4 has been installed on the eUICC.

eUICC	The PROFILE_OPERATIONAL4 is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_DISABLE_PROFILE(#ICCID_OP_PROF4, NO_PARAM, TRUE))	SW=0x6985

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF4,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO4_ENABLED
			}
			
			SW=0x9000

Test Sequence #07 Error: Disable Profile by ISDP-AID without refreshFlag while proactive session is ongoing – catBusy supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC4	Do not send FETCH command		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_DISABLE_PROFILE(NO_PARAM, <ISD_P_AID1>, FALSE))	SW=0x9300
2	S_Device → eUICC	FETCH ‘XX’	SMS POR received SCP80 response status code equal to 0x00 – POR OK

3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #08 Error: Disable Profile by ICCID with refreshFlag set while proactive session is ongoing – catBusy supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC4	Do not send FETCH command		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_DISABLE_PROFILE(#ICCID_OP_PROF1, NO_PARAM, TRUE))	SW=0x9300
2	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

4.2.22.2.3 TC_eUICC_ES10c.DisableProfile_Case4

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.

Test Sequence #01 Nominal: Disable Profile by ISD-P AID and "refreshFlag" set when Device supports "UICC Reset"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		TRUE))
			#R_DISABLE_PROFILE_OK
			SW=0x91XX

2	S_Device → eUICC	FETCH 'XX'	REFRESH Command ("UICC Reset")
3	Repeat IC1 and IC2		

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

5	S_Device → eUICC	[SELECT_ICCID]	SW=6A82

Test Sequence #02 Nominal: Disable Profile by ICCID and "refreshFlag" set when Device supports "UICC Reset"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
			#R_DISABLE_PROFILE_OK
			SW=0x91XX

		#ICCID_OP_PROF1, NO_PARAM, TRUE))	
2	S_Device → eUICC	FETCH 'XX'	REFRESH Command ("UICC Reset")
3	Repeat IC1 and IC2		

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

5	S_Device → eUICC	[SELECT_ICCID]	SW=6A82

Test Sequence #03 Nominal: Disable Profile by ISD-P AID and "refreshFlag" set when Device supports "eUICC Profile State Change"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		TRUE))
			#R_DISABLE_PROFILE_OK
			SW=0x91XX

2	S_Device → eUICC	FETCH 'XX'	REFRESH Command ("eUICC Profile State changed")
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
4	Execute IC1 from step 2 to step 4		
5	Repeat IC2		
6	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(NO_PARAM, <ISD_P_AID1>))	response ProfileInfoListResponse::= profileInfoListOk : { #PROFILE_INFO1_DISABLED

			} SW=0x9000
7	S_Device → eUICC	[SELECT_ICCID]	SW=6A82

Test Sequence #04 Nominal: Disable Profile by ICCID and "refreshFlag" set when Device supports "eUICC Profile State Change"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		#ICCID_OP_PROF1,
		NO_PARAM,
		TRUE))
			#R_DISABLE_PROFILE_OK
			SW=0x91XX

2	S_Device → eUICC	FETCH 'XX'	REFRESH Command ("eUICC Profile State changed")
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
4	Execute IC1 from step 2 to step 4		
5	Repeat IC2		

6	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

7	S_Device → eUICC	[SELECT_ICCID]	SW=6A82

Test Sequence #05 Nominal: Disable Profile by ISD-P AID and "refreshFlag" no set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		FALSE))
			#R_DISABLE_PROFILE_OK
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

3	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #06 Nominal: Disable Profile by ICCID and "refreshFlag" no set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		#ICCID_OP_PROF1,
		NO_PARAM,
		FALSE))
			#R_DISABLE_PROFILE_OK
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

3	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #07 Nominal: Disable Profile by ISD-P AID and "refreshFlag" set while proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC4	Do not send FETCH command		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		TRUE))
			resp DisableProfileResponse ::= {
			DisableResult ok
			}
			SW=0x91YY

2	S_Device →eUICC	FETCH 'YY'	REFRESH Command ("UICC Reset")
3	Repeat IC1 and IC2		

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse ::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

5	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #08 Nominal: Disable Profile by ISD-P AID and "refreshFlag" not set while proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC4	Do not send FETCH command		
1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_DISABLE_PROFILE(NO_PARAM, <ISD_P_AID1>, FALSE))	resp DisableProfileResponse ::= { DisableResult ok } SW=0x9000
2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(NO_PARAM, <ISD_P_AID1>))	response ProfileInfoListResponse ::= profileInfoListOk : { #PROFILE_INFO1_DISABLED } SW=0x9000
3	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #09 Nominal: Disable Profile by ISD-P AID and "refreshFlag" set while proactive session is ongoing with Terminal Response outstanding – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
1	S_Device →eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_DISABLE_PROFILE(NO_PARAM, <ISD_P_AID1>, TRUE))	resp DisableProfileResponse ::= { DisableResult ok } SW=0x9000

3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x91YY
4	S_Device →eUICC	FETCH 'YY'	REFRESH Command ("UICC Reset")
5	Repeat IC1 and IC2		

6	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse ::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

7	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #10 Nominal: Disable Profile by ISD-P AID and "refreshFlag" not set while proactive session is ongoing with Terminal Response outstanding – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
1	S_Device →eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		FALSE))
			resp DisableProfileResponse ::= {
			DisableResult ok
			}
			SW=0x9000

3	S_Device → eUICC	TERMINAL RESPONSE	SW= any value except 91XX

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse ::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}

			SW=0x9000
5	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

4.2.22.2.4 TC_eUICC_ES10c.DisableProfile_ErrorCases_Case4

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.

Test Sequence #01 Error: Disable Profile by an unknown ISD-P AID

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The Operational Profile identified by the ISD-P AID <ISD_P_AIDX> is not loaded.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AIDX>,
		TRUE))
			#R_DISABLE_PROFILE_ICCID_ISDP_NOTFOUND
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #02 Error: Disable Profile by an unknown ICCID

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.

eUICC	The Operational Profile identified by the ICCID #ICCID_OP_PROFX is not loaded.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		#ICCID_OP_PROFX,
		NO_PARAM,
		TRUE))
			#R_DISABLE_PROFILE_ICCID_ISDP_NOTFOUND
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #03 Error: Disable Profile (by ISD-P AID) is not possible when this Operational Profile is in Disabled state

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		TRUE))
			#R_DISABLE_PROFILE_NOT_ENABLE_STATE
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

Test Sequence #04 Error: Disable Profile (by ICCID) is not possible when this Operational Profile is in Disabled state

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		#ICCID_OP_PROF1,
		NO_PARAM,
		TRUE))
			#R_DISABLE_PROFILE_NOT_ENABLE_STATE
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

Test Sequence #05 Error: Disable Profile (by ISD-P AID) not possible when PPR1 is set

The purpose of this test is to ensure that it is NOT possible to disable an Operational Profile with the Policy Rule “Disabling of this Profile is not allowed”.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is not loaded. (this condition overrides the general initial condition defined in this test case)
eUICC	The PROFILE_OPERATIONAL4 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL4 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL4 corresponds to <ISD_P_AID4>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID4>,
		TRUE))
			#R_DISABLE_PROFILE_DISALLOWEDbyPOLICY
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID4>)**)**
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO4_ENABLED
			}
			
			SW=0x9000

Test Sequence #06 Error: Disable Profile (by ICCID) not possible when PPR1 is set

The purpose of this test is to ensure that it is NOT possible to disable an Operational Profile4 with the Policy Rule “Disabling of this Profile is not allowed”.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is not loaded. (this condition overrides the general initial condition defined in this test case)
eUICC	The PROFILE_OPERATIONAL4 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL4 is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		#ICCID_OP_PROF4,
		NO_PARAM,
		TRUE))
			#R_DISABLE_PROFILE_DISALLOWEDbyPOLICY
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF4,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO4_ENABLED
			}
			
			SW=0x9000

Test Sequence #07 Error: Disable Profile by ISD-P AID without refreshFlag while proactive session is ongoing – catBusy supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC4	Do not send FETCH command		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		FALSE))
			resp DisableProfileResponse ::= {
			disableResult catBusy
			}
			SW=0x9000 or 0x91XX

2	S_Device → eUICC	FETCH ‘XX’	SMS POR received SCP80 response status code equal to 0x00 – POR OK
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #08 Error: DisableProfile by ICCID with refreshFlag set while proactive session is ongoing – catBusy supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC4	Do not send FETCH command		
1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_DISABLE_PROFILE(#ICCID_OP_PROF1, NO_PARAM, TRUE))	resp DisableProfileResponse ::= { disableResult catBusy } SW=0x9000 or 0x91XX
2	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
4	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(#ICCID_OP_PROF1, NO_PARAM))	response ProfileInfoListResponse::= profileInfoListOk : { #PROFILE_INFO1 } SW=0x9000

4.2.22.2.5 TC_eUICC_ES10c.DisableProfile_Case4_MEPA1

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	eUICC in MEP mode
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.

Test Sequence #01 Nominal: Disable Profile by ISD-P AID and "refreshFlag" set when Device supports "UICC Reset"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE (
		NO_PARAM,
		<ISD_P_AID1>,
		TRUE))
			#R_DISABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(1, "UICC Reset")		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_FIRST_PROFILE		

4	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

5	PROC_MEP_LSI_MULTIPLEXING(1)		
6	S_Device → eUICC	[SELECT_ICCID]	SW=6A82

Test Sequence #02 Nominal: Disable Profile by ICCID and "refreshFlag" set when Device supports "UICC Reset"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		#ICCID_OP_PROF1,
		NO_PARAM,
		TRUE))
			#R_DISABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(0, "UICC Reset")		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_FIRST_PROFILE		

4	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

5	PROC_MEP_LSI_MULTIPLEXING(1)		
6	S_Device → eUICC	[SELECT_ICCID]	SW=6A82

Test Sequence #03 Nominal: Disable Profile by ISD-P AID and "refreshFlag" set when Device supports "eUICC Profile State Change"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		TRUE))
			#R_DISABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(1, "eUICC Profile State Change")		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP_EN_DS_FIRST_PROFILE		

4	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

5	proc_MEP_LSI_MULTIPLEXING(1)		
6	S_Device → eUICC	[SELECT_ICCID]	SW=6A82

Test Sequence #04 Nominal: Disable Profile by ICCID and "refreshFlag" set when Device supports "eUICC Profile State Change"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE (
		#ICCID_OP_PROF1,
		NO_PARAM,
		TRUE))
			#R_DISABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(0, "eUICC Profile State Change")		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP_EN_DS_FIRST_PROFILE		

4	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

5	PROC_MEP_LSI_MULTIPLEXING(1)		
6	S_Device → eUICC	[SELECT_ICCID]	SW=6A82

Test Sequence #05 Nominal: Disable Profile by ISD-P AID and "refreshFlag" not set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		FALSE))
			#R_DISABLE_PROFILE_OK
			SW=0x9000

2	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

3	PROC_MEP_LSI_MULTIPLEXING(1)		
4	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #06 Nominal: Disable Profile by ICCID and "refreshFlag" not set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (	Verify

		2, #IUT_MEP_LSI_OPTIONS, “010203”, 2)	<MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE (
		#ICCID_OP_PROF1,
		NO_PARAM,
		FALSE))
			#R_DISABLE_PROFILE_OK
			SW=0x9000

2	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

3	PROC_MEP_LSI_MULTIPLEXING(1)		
4	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #07 Nominal: Disable Profile by ISD-P AID and “refreshFlag” set while proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify ‘LSI Support’ is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, “010203”, 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS

IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(0)		
IC8	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_DISABLE_PROFILE(NO_PARAM, <ISD_P_AID1>, TRUE))	resp DisableProfileResponse ::= { DisableResult ok } SW=0x91YY
2	S_Device → eUICC	PROC_MEP_REFRESH_EN_DS(0, “UICC Reset”)	Verify <LSI_COMMAND_ACTION> = “Proactive session request” and <LSI_NUMBER> = 1 REFRESH Command (“UICC Reset”)
3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_FIRST_PROFILE		
4	PROC_ MEP_LSI_MULTIPLEXING(0)		
5	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(NO_PARAM, <ISD_P_AID1>))	response ProfileInfoListResponse ::= profileInfoListOk : { #PROFILE_INFO1_DISABLED } SW=0x9000
6	PROC_MEP_LSI_MULTIPLEXING(1)		
7	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #08 Nominal: Disable Profile by ISD-P AID and “refreshFlag” not set while proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify ‘LSI Support’ is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (	Verify <MEP_MODE> = #01,

		2, #IUT_MEP_LSI_OPTIONS, “010203”, 2)	Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(0)		
IC8	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		FALSE))
			resp DisableProfileResponse ::= {
			DisableResult ok
			}
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse ::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

3	PROC_MEP_LSI_MULTIPLEXING(1)		
4	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #09 Nominal: Disable 2nd Profile by ISD-P AID and “refreshFlag” set when Device supports “UICC Reset”

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 2.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify ‘LSI Support’ is present in <ATR>

IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, “010203”, 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID2>,
		TRUE))
			#R_DISABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(2, “UICC Reset”)		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_SECOND_PROFILE		

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2;
			}
			SW=0x9000

5	PROC_MEP_LSI_MULTIPLEXING(2)		
6	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #10 Nominal: Disable 2nd Profile by ICCID and “refreshFlag” set when Device supports “UICC Reset”

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to #ICCID_OP_PROF1
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 2.
eUICC	The PROFILE_OPERATIONAL2 corresponds to #ICCID_OP_PROF2

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR>

			Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		#ICCID_OP_PROF2,
		NO_PARAM,
		TRUE))
			#R_DISABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(2, "UICC Reset")		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_SECOND_PROFILE		

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2;
			}
			SW=0x9000

5	PROC_MEP_LSI_MULTIPLEXING(2)		
6	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #11 Nominal: Disable 2nd Profile by ISD-P AID and "refreshFlag" set when Device supports "eUICC Profile State Change"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 2.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID2>,
		TRUE))
			#R_DISABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(0, "eUICC Profile State Change")		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP_EN_DS_SECOND_PROFILE		

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2;
			}
			SW=0x9000

5	PROC_MEP_LSI_MULTIPLEXING(2)		
6	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #12 Nominal: Disable 2nd Profile by ICCID and "refreshFlag" set when Device supports "eUICC Profile State Change"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to #ICCID_OP_PROF1
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 2.
eUICC	The PROFILE_OPERATIONAL2 corresponds to #ICCID_OP_PROF2

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		#ICCID_OP_PROF2,
		NO_PARAM,
		TRUE))
			#R_DISABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(2, "eUICC Profile State Change")		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP_EN_DS_SECOND_PROFILE		

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2;
			}
			SW=0x9000

5	PROC_MEP_LSI_MULTIPLEXING(2)		
6	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #13 Nominal: Disable 2nd Profile by ISD-P AID and "refreshFlag" not set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 2.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID2>,
		FALSE))
			#R_DISABLE_PROFILE_OK
			SW=0x9000

IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_SECOND_PROFILE		

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			#PROFILE_INFO2
			}
			SW=0x9000

3	PROC_MEP_LSI_MULTIPLEXING(2)		
4	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #14 Nominal: Disable 2nd Profile by ICCID and "refreshFlag" not set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to #ICCID_OP_PROF1
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 2.
eUICC	The PROFILE_OPERATIONAL2 corresponds to #ICCID_OP_PROF2

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		#ICCID_OP_PROF2,
		NO_PARAM,
		FALSE))
			#R_DISABLE_PROFILE_OK
			SW=0x9000

IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_SECOND_PROFILE		

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			#PROFILE_INFO2
			}
			SW=0x9000

3	PROC_MEP_LSI_MULTIPLEXING(2)		
4	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #15 Nominal: Disable 2nd Profile by ISD-P AID and "refreshFlag" set while proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 2.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(2)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(0)		
IC8	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID2>,
		TRUE))
			resp disableProfileResponse ::= {
			disableResult ok
			}
			SW=0x91YY

2	PROC_MEP_REFRESH_EN_DS(2, "UICC Reset")		
IC9	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_SECOND_PROFILE		

3	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse ::= profileInfoListOk : {
			#PROFILE_INFO1,
			#PROFILE_INFO2
			}
			SW=0x9000

4	PROC_MEP_LSI_MULTIPLEXING(2)		
5	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #16 Nominal: Disable 2nd Profile by ISD-P AID and "refreshFlag" set while proactive session is ongoing with Terminal Response outstanding – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	eUICC is MEPA1 capable
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 2.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK and waiting for Terminal Response
IC7	PROC_MEP_LSI_MULTIPLEXING(0)		
IC8	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		TRUE))
			resp DisableProfileResponse ::= {
			disableResult ok
			}
			SW=0x91YY

2	S_Device →eUICC	FETCH 'YY'	Verify <LSI_COMMAND_ACTION> = "Proactive session request" and <LSI_NUMBER> = 1 LSI COMMAND ("ProactiveSessionRequest")
3	S_Device →eUICC	MANAGE_LSI(Select LSI) for <LSI_NUMBER>	Switch to targetPort
4	S_Device →eUICC	STATUS command	SW=0x9000
5	S_Device →eUICC	TERMINAL RESPONSE	SW=0x91ZZ
6	S_Device →eUICC	FETCH 'ZZ'	REFRESH Command ("UICC Reset")
7	S_Device → eUICC	[SELECT_MF]	FCP Template present SW=0x9000
8	S_Device → eUICC	[TERMINAL_CAPABILITY_LPAd]	SW=0x9000
9	S_Device → eUICC	[TERMINAL_PROFILE_LSI_COMMAND]	SW=0x9000
10	PROC_MEP_LSI_MULTIPLEXING(0)		

11	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			
			#PROFILE_INFO1_DISABLED;
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

12	PROC_MEP_LSI_MULTIPLEXING(1)		
13	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #17 Nominal: Disable 2nd Profile by ISD-P AID and "refreshFlag" set while 2 proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 2.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(2)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(1)		
IC8	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91YY
IC9	Do not send FETCH command		
IC10	PROC_MEP_LSI_MULTIPLEXING(0)		
IC11	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		TRUE))
			resp DisableProfileResponse ::= {
			disableResult ok
			}
			SW=0x91ZZ

2	PROC_MEP_REFRESH_EN_DS(0, UICC Reset)		
3	PROC_MEP_LSI_MULTIPLEXING(2)		
4	S_Device →eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
5	S_Device →eUICC	TERMINAL RESPONSE	
6	PROC_MEP_LSI_MULTIPLEXING(0)		

7	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO (<NO_PARAM>, <NO_PARAM>))	response ProfileInfoListResponse::= profileInfoListOk : { #PROFILE_INFO1_DISABLED; #PROFILE_INFO2_ENABLED; } SW=0x9000
8	PROC_MEP_LSI_MULTIPLEXING(1)		
9	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #18 Nominal: Disable 2nd Profile by ISD-P AID and "refreshFlag" set while 2 proactive session is ongoing with Terminal Response outstanding – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	eUICC is MEPA1 capable
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 2.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(2)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX

IC6	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK and waiting for Terminal Response
IC7	PROC_MEP_LSI_MULTIPLEXING(1)		
IC8	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91YY
IC9	S_Device → eUICC	FETCH 'YY'	SMS POR received SCP80 response status code equal to 0x00 – POR OK and waiting for Terminal Response
IC10	PROC_MEP_LSI_MULTIPLEXING(0)		
IC11	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		TRUE))
			resp DisableProfileResponse ::= {
			disableResult ok
			}
			SW=0x91ZZ

2	S_Device →eUICC	FETCH 'ZZ'	Verify <LSI_COMMAND_ACTION> = "Proactive session request" and <LSI_NUMBER> = 1 LSI COMMAND ("ProactiveSessionRequest")
3	S_Device →eUICC	MANAGE_LSI(Select LSI) for <LSI_NUMBER>	Switch to targetPort
4	S_Device →eUICC	STATUS command	SW=0x9000
5	S_Device →eUICC	TERMINAL RESPONSE	SW=0x91KK
6	S_Device →eUICC	FETCH 'KK'	REFRESH Command ("UICC Reset")
7	S_Device → eUICC	[SELECT_MF]	FCP Template present SW=0x9000
8	S_Device → eUICC	[TERMINAL_CAPABILITY_LPAd]	SW=0x9000
9	S_Device → eUICC	[TERMINAL_PROFILE_LSI_COMMAND]	SW=0x9000
10	PROC_MEP_LSI_MULTIPLEXING(2)		
11	S_Device →eUICC	TERMINAL RESPONSE	SW=0x9000
12	PROC_MEP_LSI_MULTIPLEXING(0)		

13	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			
			#PROFILE_INFO1_DISABLED;
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

14	PROC_MEP_LSI_MULTIPLEXING(1)		
15	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #19 Nominal: Disable 2nd Profile (by ISD-P AID) with Refresh ON is performed successfully when CAT is not initialized for targetPort

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 2.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_ONE_LSI_FOR_ENABLED_PROFILE		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID2>,
		TRUE))
			#R_DISABLE_PROFILE_OK
			SW=0x9000

2	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_SECOND_PROFILE		

3	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2;
			}
			SW=0x9000

4	PROC_MEP_LSI_MULTIPLEXING(2)		
5	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

4.2.22.2.6 TC_eUICC_ES10c.DisableProfile_ErrorCases_Case4_MEPA1

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	eUICC in MEP mode
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.

Test Sequence #01 Error: Disable Profile by an unknown ISD-P AID

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The Operational Profile identified by the ISD-P AID <ISD_P_AIDX> is not loaded.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		

IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AIDX>,
		TRUE))
			#R_DISABLE_PROFILE_ICCID_ISDP_NOTFOUND
			SW=0x9000

2	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse ::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #02 Error: Disable Profile by an unknown ICCID

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The Operational Profile identified by the ICCID #ICCID_OP_PROFX is not loaded.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE (
		#ICCID_OP_PROFX,
		NO_PARAM,
		TRUE))
			#R_DISABLE_PROFILE_ICCID_ISDP_NOTFOUND
			SW=0x9000

2	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse ::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #03 Error: Disable Profile (by ISD-P AID) is not possible when this Operational Profile is in Disabled state

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		TRUE))
			#R_DISABLE_PROFILE_NOT_ENABLE_STATE
			SW=0x9000

2	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

Test Sequence #04 Error: Disable Profile (by ICCID) is not possible when this Operational Profile is in Disabled state

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		#ICCID_OP_PROF1,
		NO_PARAM,
		TRUE))
			#R_DISABLE_PROFILE_NOT_ENABLE_STATE
			SW=0x9000

2	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

Test Sequence #05 Error: Disable Profile by ISD-P AID without refreshFlag while proactive session is ongoing – catBusy supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(0)		
IC8	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		FALSE))
			resp DisableProfileResponse ::= {
			disableResult catBusy
			}
			SW=0x9000

2	PROC_MEP_LSI_MULTIPLEXING(1)		
3	S_Device →eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
4	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse ::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #06 Error: DisableProfile by ICCID with refreshFlag set while proactive session is ongoing – catBusy supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on Port 1.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(0)		
IC8	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		#ICCID_OP_PROF1,
		NO_PARAM,
		TRUE))
			resp DisableProfileResponse ::= {
			disableResult catBusy
			}
			SW=0x9000

2	PROC_MEP_LSI_MULTIPLEXING(1)		
3	S_Device →eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
4	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse ::= profileInfoListOk : {
			#PROFILE_INFO1
			}

			SW=0x9000

Test Sequence #07 Error: Disable 3rd Profile by an unknown ISD-P AID where two profile is already enabled

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 2.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.
eUICC	The Operational Profile identified by the ISD-P AID <ISD_P_AIDX> is not loaded.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(0)		
IC5	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AIDX>,
		TRUE))
			#R_DISABLE_PROFILE_ICCID_ISDP_NOT FOUND
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2_ENABLED;
			}

			SW=0x9000

Test Sequence #08 Error: Disable 3rd Profile by an unknown ICCID where two profile is already enabled

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to #ICCID_OP_PROF1.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 2.
eUICC	The PROFILE_OPERATIONAL2 corresponds to #ICCID_OP_PROF2.
eUICC	The Operational Profile identified by the ICCID #ICCID_OP_PROFX is not loaded.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP \(2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		#ICCID_OP_PROFX,
		NO_PARAM,
		TRUE))
			#R_DISABLE_PROFILE_ICCID_ISDP_NOT FOUND
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

Test Sequence #09 Error: Disable 3rd Profile (by ISD-P AID) is not possible when this Operational Profile is in Disabled State

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 2.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 corresponds to <ISD_P_AID3>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID3>,
		TRUE))
			#R_DISABLE_PROFILE_NOT_ENABLE_STATE
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

Test Sequence #10 Error: Disable 3rd Profile (by ICCID) is not possible when this Operational Profile is in Disabled state

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to #ICCID_OP_PROF1
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 2.
eUICC	The PROFILE_OPERATIONAL2 corresponds to #ICCID_OP_PROF2
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 corresponds to #ICCID_OP_PROF3

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		#ICCID_OP_PROF3,
		NO_PARAM,
		TRUE,
		<MEP_MODE>,
		1))
			#R_DISABLE_PROFILE_NOT_ENABLE_STATE
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			}
			SW=0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(	response ProfileInfoListResponse::= profileInfoListOk : { #PROFILE_INFO2_ENABLED;

		#ICCID_OP_PROF2, NO_PARAM))	} SW=0x9000

Test Sequence #11 Error: Disable 3rd Profile by ISD-P AID and "refreshFlag" set while 2 proactive session is ongoing – catBusy supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 2.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(2)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(1)		
IC8	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91YY
IC9	Do not send FETCH command		
IC10	PROC_MEP_LSI_MULTIPLEXING(0)		
IC11	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_DISABLE_PROFILE(	resp DisableProfileResponse ::= {

		NO_PARAM, <ISD_P_AID1>, TRUE))	disableResult catBusy } SW=0x9000
2	Repeat IC7		
3	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
4	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
5	Repeat IC4		
6	S_Device → eUICC	FETCH 'YY'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
7	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000

8	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response
			ProfileInfoListResponse ::=
			profileInfoListOk : {
			#PROFILE_INFO1,
			
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

4.2.22.2.7 TC_eUICC_ES10c.DisableProfile_Case4_MEPA2

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	eUICC in MEP mode
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.

Test Sequence #01 Nominal: Disable Profile by ISD-P AID and "refreshFlag" not set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on target port <MEP-A2_TARGET_ESIM_PORT> assigned by the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE (
		NO_PARAM,
		<ISD_P_AID1>,
		FALSE))
			#R_DISABLE_PROFILE_OK
			SW=0x9000

2	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

3	PROC_MEP_LSI_MULTIPLEXING(<MEP-A2_TARGET_ESIM_PORT>)		
4	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #02 Nominal: Disable Profile by ICCID and "refreshFlag" not set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on target port <MEP-A2_TARGET_ESIM_PORT> assigned by the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>

IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, “020103”, 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		#ICCID_OP_PROF1,
		NO_PARAM,
		FALSE))
			#R_DISABLE_PROFILE_OK
			SW=0x9000

2	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

3	PROC_MEP_LSI_MULTIPLEXING(<MEP-A2_TARGET_ESIM_PORT>)		
4	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #03 Nominal: Disable Profile by ISD-P AID and “refreshFlag” not set while proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on target port <MEP-A2_TARGET_ESIM_PORT> assigned by the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify ‘LSI Support’ is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, “020103”, 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify

			<MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(<MEP-A2_TARGET_ESIM_PORT>)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(0)		
IC8	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_DISABLE_PROFILE(NO_PARAM, <ISD_P_AID1>, FALSE))	resp DisableProfileResponse ::= { DisableResult ok } SW=0x9000
2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(NO_PARAM, <ISD_P_AID1>))	response ProfileInfoListResponse ::= profileInfoListOk : { #PROFILE_INFO1_DISABLED } SW=0x9000
3	PROC_MEP_LSI_MULTIPLEXING(<MEP-A2_TARGET_ESIM_PORT>)		
4	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #04 Nominal: Disable 2nd Profile by ISD-P AID and "refreshFlag" not set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port X.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port Y.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (	Verify <MEP_MODE> = 02,

		2, #IUT_MEP_LSI_OPTIONS, “020103”, 2)	Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID2>,
		FALSE))
			#R_DISABLE_PROFILE_OK
			SW=0x9000

IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_SECOND_PROFILE		

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			#PROFILE_INFO2
			}
			SW=0x9000

3	PROC_MEP_LSI_MULTIPLEXING(Y)		
4	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #05 Nominal: Disable 2nd Profile by ICCID and “refreshFlag” not set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port X.
eUICC	The PROFILE_OPERATIONAL1 corresponds to #ICCID_OP_PROF1
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port Y.
eUICC	The PROFILE_OPERATIONAL2 corresponds to #ICCID_OP_PROF2

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify ‘LSI Support’ is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2,	Verify <MEP_MODE> = 02,

		#IUT_MEP_LSI_OPTIONS, “020103”, 2)	Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		#ICCID_OP_PROF2,
		NO_PARAM,
		FALSE))
			#R_DISABLE_PROFILE_OK
			SW=0x9000

2	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_SECOND_PROFILE		

3	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			#PROFILE_INFO2
			}
			SW=0x9000

4	PROC_MEP_LSI_MULTIPLEXING(Y)		
5	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

4.2.22.2.8 TC_eUICC_ES10c.DisableProfile_ErrorCases_Case4_MEPA2

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	eUICC in MEP mode
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.

Test Sequence #01 Error: Disable Profile by an unknown ISD-P AID

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The Operational Profile identified by the ISD-P AID <ISD_P_AIDX> is not loaded.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AIDX>,
		FALSE))
			#R_DISABLE_PROFILE_ICCID_ISDP_NOTFOUND
			SW=0x9000

2	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse ::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #02 Error: Disable Profile by an unknown ICCID

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The Operational Profile identified by the ICCID #ICCID_OP_PROFX is not loaded.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify

			<MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_DISABLE_PROFILE (#ICCID_OP_PROFX, NO_PARAM, FALSE))	#R_DISABLE_PROFILE_ICCID_ISDP_NOTFOUND SW=0x9000
2	S_LPAd →eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(#ICCID_OP_PROF1, NO_PARAM))	response ProfileInfoListResponse ::= profileInfoListOk : { #PROFILE_INFO1 } SW=0x9000

Test Sequence #03 Error: Disable Profile (by ISD-P AID) is not possible when this Operational Profile is in Disabled state

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_DISABLE_PROFILE(NO_PARAM, <ISD_P_AID1>, FALSE))	#R_DISABLE_PROFILE_NOT_ENABLE_STATE SW=0x9000

2	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

Test Sequence #04 Error: Disable Profile (by ICCID) is not possible when this Operational Profile is in Disabled state

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		#ICCID_OP_PROF1,
		NO_PARAM,
		FALSE))
			#R_DISABLE_PROFILE_NOT_ENABLE_STATE
			SW=0x9000

2	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

Test Sequence #05 Error: Disable Profile by ISD-P AID without refreshFlag while proactive session is ongoing – catBusy supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on target port <MEP-A2_TARGET_ESIM_PORT> assigned by the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(<MEP-A2_TARGET_ESIM_PORT>)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(0)		
IC8	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		FALSE))
			resp DisableProfileResponse ::= {
			disableResult catBusy
			}
			SW=0x9000

2	PROC_MEP_LSI_MULTIPLEXING(<MEP-A2_TARGET_ESIM_PORT>)		
3	S_Device →eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
4	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse ::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #06 Error: Disable 3rd Profile by an unknown ISD-P AID where two profile is already enabled

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.
eUICC	The Operational Profile identified by the ISD-P AID <ISD_P_AIDX> is not loaded.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(0)		
IC5	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AIDX>,
		FALSE))
			#R_DISABLE_PROFILE_ICCID_ISDP_NOT FOUND
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

Test Sequence #07 Error: Disable 3rd Profile by an unknown ICCID where two profile is already enabled

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to #ICCID_OP_PROF1.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to #ICCID_OP_PROF2.
eUICC	The Operational Profile identified by the ICCID #ICCID_OP_PROFX is not loaded.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		#ICCID_OP_PROFX,
		NO_PARAM,
		FALSE))
			#R_DISABLE_PROFILE_ICCID_ISDP_NOT FOUND
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(	response ProfileInfoListResponse::= profileInfoListOk : { #PROFILE_INFO1;

		<NO_PARAM>, <NO_PARAM>))	#PROFILE_INFO2_ENABLED; } SW=0x9000

Test Sequence #08 Error: Disable 3rd Profile (by ISD-P AID) is not possible when this Operational Profile is in Disabled State

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 corresponds to <ISD_P_AID3>.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID3>,
		FALSE))
			#R_DISABLE_PROFILE_NOT_ENABLE_STATE
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2_ENABLED;
			}

			SW=0x9000

Test Sequence #09 Error: Disable 3rd Profile (by ICCID) is not possible when this Operational Profile is in Disabled state

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to #ICCID_OP_PROF1
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to #ICCID_OP_PROF2
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 corresponds to #ICCID_OP_PROF3

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_PROFILE(
		#ICCID_OP_PROF3,
		NO_PARAM,
		FALSE,
		<MEP_MODE>,
		1))
			#R_DISABLE_PROFILE_NOT_ENABLE_STATE
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			}
			SW=0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF2,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

Test Sequence #10 Error: Disable 3rd Profile by ISD-P AID and "refreshFlag" not set while 2 proactive session is ongoing – catBusy supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port X.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port Y.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(Y)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(X)		
IC8	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91YY
IC9	Do not send FETCH command		
IC10	PROC_MEP_LSI_MULTIPLEXING(0)		
IC11	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE (
		NO_PARAM,
		<ISD_P_AID1>,
		FALSE))
			resp DisableProfileResponse ::= {
			disableResult catBusy
			}
			SW=0x9000

2	Repeat IC7		
3	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
4	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
5	Repeat IC4		
6	S_Device → eUICC	FETCH 'YY'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
7	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000

8	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse ::= profileInfoListOk : {
			#PROFILE_INFO1,
			
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

4.2.22.2.9 TC_eUICC_ES10c.DisableProfile_Case4_MEPB

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	eUICC in MEP mode
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.

Test Sequence #01 Nominal: Disable Profile by ISD-P AID and "refreshFlag" set when Device supports "UICC Reset"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		TRUE))
			#R_DISABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(0, "UICC Reset")		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_FIRST_PROFILE		
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

5	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

6	S_Device → eUICC	[SELECT_ICCID]	SW=6A82

Test Sequence #02 Nominal: Disable Profile by ICCID and "refreshFlag" set when Device supports "UICC Reset"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE (
		#ICCID_OP_PROF1,
		NO_PARAM,
		TRUE))
			#R_DISABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(0, "UICC Reset")		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_FIRST_PROFILE		
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

5	S_LPAd →Euicc	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

6	S_Device → Euicc	[SELECT_ICCID]	SW=6A82

Test Sequence #03 Nominal: Disable Profile by ISD-P AID and "refreshFlag" set when Device supports "Euicc Profile State Change"

Initial Conditions	
Entity	**Description of the initial condition**
Euicc	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE (
		NO_PARAM,
		<ISD_P_AID1>,
		TRUE))
			#R_DISABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(0, "eUICC Profile State Change")		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP_EN_DS_FIRST_PROFILE		
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

5	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

6	S_Device → eUICC	[SELECT_ICCID]	SW=6A82

Test Sequence #04 Nominal: Disable Profile by ICCID and "refreshFlag" set when Device supports "eUICC Profile State Change"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE (
		#ICCID_OP_PROF1,
		NO_PARAM,
		TRUE))
			#R_DISABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(0, "eUICC Profile State Change")		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP_EN_DS_FIRST_PROFILE		
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

5	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

6	S_Device → eUICC	[SELECT_ICCID]	SW=6A82

Test Sequence #05 Nominal: Disable Profile by ISD-P AID and "refreshFlag" not set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		FALSE))
			#R_DISABLE_PROFILE_OK
			SW=0x9000

2	S_Device → eUICC	[TERMINAL_PROFILE_LSI_COMMAND]	Toolkit initialization THEN SW=0x9000

3	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

4	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #06 Nominal: Disable Profile by ICCID and "refreshFlag" not set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICCon Port 0.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>

IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, “030201”, 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		#ICCID_OP_PROF1,
		NO_PARAM,
		FALSE))
			#R_DISABLE_PROFILE_OK
			SW=0x9000

2	S_Device → eUICC	[TERMINAL_PROFILE_LSI_COMMAND]	Toolkit initialization THEN SW=0x9000

3	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

4	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #07 Nominal: Disable Profile by ISD-P AID and “refreshFlag” set while proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify ‘LSI Support’ is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, “030201”, 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify

			<MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC5	Do not send FETCH command		
IC6	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_DISABLE_PROFILE(NO_PARAM, <ISD_P_AID1>, TRUE))	resp DisableProfileResponse ::= { DisableResult ok } SW=0x91YY
2	PROC_MEP_REFRESH_EN_DS(0, “UICC Reset”)		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_FIRST_PROFILE		
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
5	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(NO_PARAM, <ISD_P_AID1>))	response ProfileInfoListResponse ::= profileInfoListOk : { #PROFILE_INFO1_DISABLED } SW=0x9000
6	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #08 Nominal: Disable Profile by ISD-P AID and “refreshFlag” not set while proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify ‘LSI Support’ is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, “030201”,	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS,

		2)	Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC5	Do not send FETCH command		
IC6	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		FALSE))
			resp DisableProfileResponse ::= {
			DisableResult ok
			}
			SW=0x9000

2	S_Device → eUICC	[TERMINAL_PROFILE_LSI_COMMAND]	Toolkit initialization THEN SW=0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse ::=
			profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

4	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #09 Nominal: Disable 2nd Profile by ISD-P AID and "refreshFlag" set when Device supports "UICC Reset"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC Port 1.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS,	Verify <MEP_MODE> = 03, Verify

		"030201", 2)	<MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC5	PROC_MEP_LSI_MULTIPLEXING(1)		
IC6	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID2>,
		TRUE))
			#R_DISABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(0, "UICC Reset")		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_SECOND_PROFILE		
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_MEPB;
			#PROFILE_INFO2;
			}
			SW=0x9000

6	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #10 Nominal: Disable 2nd Profile by ICCID and "refreshFlag" set when Device supports "UICC Reset"

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to #ICCID_OP_PROF1
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC Port 1.
eUICC	The PROFILE_OPERATIONAL2 corresponds to #ICCID_OP_PROF2

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>

IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, “030201”, 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE (
		#ICCID_OP_PROF2,
		NO_PARAM,
		TRUE))
			#R_DISABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(0, “UICC Reset”)		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_SECOND_PROFILE		
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

5	S_LPAd → Euicc	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		<NO_PARAM>,
		<NO_PARAM>))
			response
			ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_MEPB;
			#PROFILE_INFO2;
			}
			SW=0x9000

6	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #11 Nominal: Disable 2nd Profile by ISD-P AID and “refreshFlag” set when Device supports “eUICC Profile State Change”

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify ‘LSI Support’ is present in <ATR>

IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, “030201”, 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE (
		NO_PARAM,
		<ISD_P_AID2>,
		TRUE))
			#R_DISABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(0, “eUICC Profile State Change”)		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP_EN_DS_SECOND_PROFILE		
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_MEPB;
			#PROFILE_INFO2;
			}
			SW=0x9000

6	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #12 Nominal: Disable 2nd Profile by ICCID and “refreshFlag” set when Device supports “eUICC Profile State Change”

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to #ICCID_OP_PROF1
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 corresponds to #ICCID_OP_PROF2

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE (
		#ICCID_OP_PROF2,
		NO_PARAM,
		TRUE))
			#R_DISABLE_PROFILE_OK
			SW=0x91XX

2	PROC_MEP_REFRESH_EN_DS(0, "eUICC Profile State Change")		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP_EN_DS_SECOND_PROFILE		
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_MEPB;
			#PROFILE_INFO2;
			}
			SW=0x9000

6	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #13 Nominal: Disable 2nd Profile by ISD-P AID and "refreshFlag" not set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE (
		NO_PARAM,
		<ISD_P_AID2>,
		FALSE))
			#R_DISABLE_PROFILE_OK
			SW=0x9000

2	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_SECOND_PROFILE		

3	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_MEPB
			#PROFILE_INFO2
			}
			SW=0x9000

4	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #14 Nominal: Disable 2nd Profile by ICCID and "refreshFlag" not set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to #ICCID_OP_PROF1
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 corresponds to #ICCID_OP_PROF2

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		#ICCID_OP_PROF2,
		NO_PARAM,
		FALSE))
			#R_DISABLE_PROFILE_OK
			SW=0x9000

2	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_SECOND_PROFILE		

3	S_LPAd → Euicc	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_MEPB
			#PROFILE_INFO2
			}
			SW=0x9000

4	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #15 Nominal: Disable 2nd Profile by ISD-P AID and "refreshFlag" set while proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE (
		NO_PARAM,
		<ISD_P_AID2>,
		TRUE))
			resp disableProfileResponse ::= {
			disableResult ok
			}
			SW=0x91YY

2	PROC_MEP_REFRESH_EN_DS(1, "UICC Reset")		
3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_SECOND_PROFILE		
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

5	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse ::= profileInfoListOk : {
			#PROFILE_INFO1_MEPB,
			#PROFILE_INFO2
			}
			SW=0x9000

6	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #16 Nominal: Disable 2nd Profile by ISD-P AID and "refreshFlag" set while proactive session is ongoing with Terminal Response outstanding – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	eUICC is MEPA1 capable
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK and waiting for Terminal Response
IC7	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE (
		NO_PARAM,
		<ISD_P_AID1>,
		TRUE))
			resp DisableProfileResponse ::= {
			disableResult ok
			}
			SW=0x9000

2	S_Device →eUICC	STATUS command	SW=0x9000
3	S_Device →eUICC	TERMINAL RESPONSE	SW=0x91ZZ
4	PROC_MEP_REFRESH_EN_DS(1, “UICC Reset”)		
5	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_SECOND_PROFILE		
6	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

7	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<ISD_P_AID2>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO2
			}
			SW=0x9000

8	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #17 Nominal: Disable 2nd Profile by ISD-P AID and “refreshFlag” set while 2 proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify ‘LSI Support’ is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, “030201”, 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		

IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(0)		
IC8	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91YY
IC9	Do not send FETCH command		
IC10	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>,
		TRUE))
			resp DisableProfileResponse ::= {
			disableResult ok
			}
			SW=0x91ZZ

2	PROC_MEP_REFRESH_EN_DS(0, UICC Reset)		
3	PROC_MEP_LSI_MULTIPLEXING(1)		
4	S_Device →eUICC	FETCH ‘XX’	SMS POR received SCP80 response status code equal to 0x00 – POR OK
5	S_Device →eUICC	TERMINAL RESPONSE	
6	PROC_MEP_LSI_MULTIPLEXING(0)		
7	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_FIRST_PROFILE		
8	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

9	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

10	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #18 Nominal: Disable 2nd Profile by ISD-P AID and "refreshFlag" set while 2 proactive session is ongoing with Terminal Response outstanding – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	eUICC is MEPA1 capable
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK and waiting for Terminal Response
IC7	PROC_MEP_LSI_MULTIPLEXING(0)		
IC8	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91YY
IC9	S_Device → eUICC	FETCH 'YY'	SMS POR received SCP80 response status code equal to 0x00 – POR OK and waiting for Terminal Response
IC10	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE (
		NO_PARAM,
		<ISD_P_AID1>,
		TRUE))
			resp DisableProfileResponse ::= {
			disableResult ok
			}
			SW=0x9000

2	S_Device →eUICC	STATUS command	SW=0x9000
3	S_Device →eUICC	TERMINAL RESPONSE	SW=0x91KK
4	PROC_MEP_REFRESH_EN_DS(1, “UICC Reset”)		
5	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_FIRST_PROFILE		
6	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
7	PROC_MEP_LSI_MULTIPLEXING(1)		
8	S_Device →eUICC	TERMINAL RESPONSE	SW=0x9000
9	PROC_MEP_LSI_MULTIPLEXING(0)		

10	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED;
			#PROFILE_INFO2_MEPB;
			}
			SW=0x9000

11	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

Test Sequence #19 Nominal: Disable 2nd Profile (by ISD-P AID) with “refreshFlag” set is performed successfully when CAT is not initialized for targetPort

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_ONE_LSI_FOR_ENABLED_PROFILE		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE (
		NO_PARAM,
		<ISD_P_AID2>,
		TRUE))
			#R_DISABLE_PROFILE_OK
			SW=0x9000

2	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_SECOND_PROFILE		
3	PROC_MEP_LSI_MULTIPLEXING(0)		

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2;
			}
			SW=0x9000

5	PROC_MEP_LSI_MULTIPLEXING(1)		
6	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

4.2.22.2.10 TC_eUICC_ES10c.DisableProfile_ErrorCases_Case4_MEPB

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	eUICC in MEP mode
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.

Test Sequence #01 Error: Disable Profile by an unknown ISD-P AID

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The Operational Profile identified by the ISD-P AID <ISD_P_AIDX> is not loaded.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE (
		NO_PARAM,
		<ISD_P_AIDX>,
		TRUE))
			#R_DISABLE_PROFILE_ICCID_ISDP_NOTFOUND
			SW=0x9000

2	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse ::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #02 Error: Disable Profile by an unknown ICCID

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The Operational Profile identified by the ICCID #ICCID_OP_PROFX is not loaded.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		#ICCID_OP_PROFX,
		NO_PARAM,
		TRUE))
			#R_DISABLE_PROFILE_ICCID_ISDP_NOTFOUND
			SW=0x9000

2	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #03 Error: Disable Profile (by ISD-P AID) is not possible when this Operational Profile is in Disabled state

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify

			<MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_DISABLE_PROFILE(NO_PARAM, <ISD_P_AID1>, TRUE))	#R_DISABLE_PROFILE_NOT_ENABLE_STATE SW=0x9000
2	S_LPAd →eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO (NO_PARAM, <ISD_P_AID1>))	response ProfileInfoListResponse::= profileInfoListOk : { #PROFILE_INFO1_DISABLED } SW=0x9000

Test Sequence #04 Error: Disable Profile (by ICCID) is not possible when this Operational Profile is in Disabled state

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(MTD_DISABLE_PROFILE (#ICCID_OP_PROF1, NO_PARAM, TRUE))	#R_DISABLE_PROFILE_NOT_ENABLE_STATE SW=0x9000
2	S_LPAd →eUICC	MTD_STORE_DATA(	response ProfileInfoListResponse::= profileInfoListOk : {

		MTD_GET_PROFILE_INFO (#ICCID_OP_PROF1, NO_PARAM))	#PROFILE_INFO1_DISABLED } SW=0x9000

Test Sequence #05 Error: Disable Profile by ISD-P AID without refreshFlag while proactive session is ongoing – catBusy supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC5	Do not send FETCH command		
IC6	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE (
		NO_PARAM,
		<ISD_P_AID1>,
		FALSE))
			resp DisableProfileResponse ::= {
			disableResult catBusy
			}
			SW=0x9000

2	S_Device →eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse ::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #06 Error: DisableProfile by ICCID with refreshFlag set while proactive session is ongoing – catBusy supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on Port 0.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC5	Do not send FETCH command		
IC6	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		#ICCID_OP_PROF1,
		NO_PARAM,
		TRUE))
			resp DisableProfileResponse ::= {
			disableResult catBusy
			}
			SW=0x9000

2	S_Device →eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #07 Error: Disable 3rd Profile by an unknown ISD-P AID where two profiles are already enabled

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.
eUICC	The Operational Profile identified by the ISD-P AID <ISD_P_AIDX> is not loaded.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(0)		
IC5	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AIDX>,
		TRUE))
			#R_DISABLE_PROFILE_ICCID_ISDP_NOT FOUND
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2_MEPB
			}
			SW=0x9000

Test Sequence #08 Error: Disable 3rd Profile by an unknown ICCID where two profiles are already enabled

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to #ICCID_OP_PROF1.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 corresponds to #ICCID_OP_PROF2.
eUICC	The Operational Profile identified by the ICCID #ICCID_OP_PROFX is not loaded.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		#ICCID_OP_PROFX,
		NO_PARAM,
		TRUE))
			#R_DISABLE_PROFILE_ICCID_ISDP_NOT FOUND
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(	response ProfileInfoListResponse::= profileInfoListOk : { #PROFILE_INFO1;

		<NO_PARAM>, <NO_PARAM>))	#PROFILE_INFO2_MEPB } SW=0x9000

Test Sequence #09 Error: Disable 3rd Profile (by ISD-P AID) is not possible when this Operational Profile is in Disabled State

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 corresponds to <ISD_P_AID3>.
eUICC	The Nickname of the PROFILE_OPERATIONAL3 is equal to #NICKNAME3.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID3>,
		TRUE))
			#R_DISABLE_PROFILE_NOT_ENABLE_STATE
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(<NO_PARAM>,	response ProfileInfoListResponse::= profileInfoListOk : { #PROFILE_INFO1;

		<NO_PARAM>))	#PROFILE_INFO2_MEPB; #PROFILE_INFO3 } SW=0x9000

Test Sequence #10 Error: Disable 3rd Profile (by ICCID) is not possible when this Operational Profile is in Disabled state

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to #ICCID_OP_PROF1
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 corresponds to #ICCID_OP_PROF2
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 corresponds to #ICCID_OP_PROF3

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		#ICCID_OP_PROF3,
		NO_PARAM,
		TRUE))
			#R_DISABLE_PROFILE_NOT_ENABLE_STATE
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}

			SW=0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF2,
		NO_PARAM))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO2_MEPB
			}
			SW=0x9000

Test Sequence #11 Error: Disable 3rd Profile by ISD-P AID and "refreshFlag" set while 2 proactive sessions are ongoing – catBusy supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030201", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(0)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(1)		
IC8	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91YY
IC9	Do not send FETCH command		
IC10	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DISABLE_PROFILE(
		NO_PARAM,
		<ISD_P_AID2>,
		TRUE))
			resp DisableProfileResponse ::= {
			disableResult catBusy
			}
			SW=0x9000

2	Repeat IC4		
3	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
4	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
5	Repeat IC7		
6	S_Device → eUICC	FETCH 'YY'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
7	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000

8	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_MEPB,
			
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

4.2.23 ES10c (LPA -- eUICC): DeleteProfile

4.2.23.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 2.4.4
- Section 2.9.1, 2.9.3.3
- Section 3.2.3
- Section 5.7.15, 5.7.18

4.2.23.2 Test Cases

4.2.23.2.1 TC_eUICC_ES10c.DeleteProfile_Case3

General Initial Conditions	
Entity	**Description of the general initial condition**

eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.

Test Sequence #01 Nominal: Delete Profile by ISD-P AID

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_DELETE_PROFILE(NO_PARAM, <ISD_P_AID1>)	No response data is returned SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::= profileInfoListOk: {
			}
			SW=0x9000

Test Sequence #02 Nominal: Delete Profile by ICCID

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_DELETE_PROFILE(#ICCID_OP_PROF1, NO_PARAM)	No response data is returned SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk: {
			}
			SW=0x9000

4.2.23.2.2 TC_eUICC_ES10c.DeleteProfile_ErrorCases_Case3

General Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.

Test Sequence #01 Error: Delete Profile not possible with unknown ISD-P AID

The purpose of this test is to ensure that it is NOT possible to delete an Operational Profile with an unknown ISD-P AID.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The Operational Profile identified by the ISD-P AID <ISD_P_AIDX> is not loaded.
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_DELETE_PROFILE(NO_PARAM, <ISD_P_AIDX>)	SW=0x6A82

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response
			ProfileInfoListResponse::=
			profileInfoListOk: {
			#PROFILE_INFO1_DISABLED,
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

Test Sequence #02 Error: Delete Profile not possible with unknown ICCID

The purpose of this test is to ensure that it is NOT possible to delete an Operational Profile with an unknown ICCID.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The Operational Profile identified by the ICCID #ICCID_OP_PROFX is not loaded.
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_DELETE_PROFILE(#ICCID_OP_PROFX, NO_PARAM**)**	SW=0x6A82

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response
			ProfileInfoListResponse::=
			profileInfoListOk: {
			#PROFILE_INFO1_DISABLED,
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

Test Sequence #03 Error: Delete Profile (by ISD-P AID) not possible when this Operational Profile is in Enabled state

The purpose of this test is to ensure that it is NOT possible to delete an Operational Profile in Enabled state.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(	SW=0x6985

		MTD_DELETE_PROFILE(NO_PARAM, <ISD_P_AID2>)	

2	S_LPAd → eUICC	MTD_STORE_DATA (
		#GET_PROFILES_INFO_ALL)
			profileInfoListOk: {
			#PROFILE_INFO1_DISABLED,
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

Test Sequence #04 Error: Delete Profile (by ICCID) not possible when this Operational Profile is in Enabled state

The purpose of this test is to ensure that it is NOT possible to delete an Operational Profile in Enabled state.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_DELETE_PROFILE(#ICCID_OP_PROF2, NO_PARAM**)**	SW=0x6985

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response
			ProfileInfoListResponse::=
			profileInfoListOk: {
			#PROFILE_INFO1_DISABLED,
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

Test Sequence #05 Error: Delete Profile (by ISD-P AID) not possible when PPR2 is set

The purpose of this test is to ensure that it is NOT possible to delete an Operational Profile with the Policy Rule “Deletion of this Profile is not allowed”.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL3 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 corresponds to <ISD_P_AID3>.
eUICC	The Nickname of the PROFILE_OPERATIONAL3 is equal to #NICKNAME3.
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_DELETE_PROFILE(NO_PARAM, <ISD_P_AID3>)	SW=0x6985

2	S_LPAd → eUICC	MTD_STORE_DATA(#GET_PROFILES_INFO_ALL)	response ProfileInfoListResponse::= profileInfoListOk: {
			#PROFILE_INFO1_DISABLED,
			#PROFILE_INFO2_ENABLED,
			#PROFILE_INFO3
			}
			SW=0x9000

Test Sequence #06 Error: Delete Profile (by ICCID) not possible when PPR2 is set

The purpose of this test is to ensure that it is NOT possible to delete an Operational Profile with the Policy Rule “Deletion of this Profile is not allowed”.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL3 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.
eUICC	The Nickname of the PROFILE_OPERATIONAL3 is equal to #NICKNAME3.
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_DELETE_PROFILE(#ICCID_OP_PROF3, NO_PARAM)	SW=0x6985

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response
			ProfileInfoListResponse::=
			profileInfoListOk: {
			#PROFILE_INFO1_DISABLED,
			#PROFILE_INFO2_ENABLED,
			#PROFILE_INFO3
			}
			SW=0x9000

4.2.23.2.3 TC_eUICC_ES10c.DeleteProfile_Case4

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.

Test Sequence #01 Nominal: Delete Profile by ISD-P AID

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DELETE_PROFILE(
		NO_PARAM,
		<ISD_P_AID1>)
			#R_DELETE_PROFILE_OK
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response
			ProfileInfoListResponse::=
			profileInfoListOk: {
			}
			SW=0x9000

Test Sequence #02 Nominal: Delete Profile by ICCID

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DELETE_PROFILE(
		#ICCID_OP_PROF1,
		NO_PARAM)
			#R_DELETE_PROFILE_OK
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::=
			profileInfoListOk: {
			}
			SW=0x9000

4.2.23.2.4 TC_eUICC_ES10c.DeleteProfile_ErrorCases_Case4

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.

Test Sequence #01 Error: Delete Profile not possible with unknown ISD-P AID

The purpose of this test is to ensure that it is NOT possible to delete an Operational Profile with an unknown ISD-P AID.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	A Operational Profile identified by the ISD-P AID <ISD_P_AIDX> is not loaded.
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		

IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DELETE_PROFILE(
		NO_PARAM,
		<ISD_P_AIDX>)
			#R_DELETE_PROFILE_ICCID_ISDP_NOTFOUND
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::=
			profileInfoListOk: {
			#PROFILE_INFO1_DISABLED,
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

Test Sequence #02 Error: Delete Profile not possible with unknown ICCID

The purpose of this test is to ensure that it is NOT possible to delete an Operational with an ICCID unknown.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The Operational Profile identified by the ICCID #ICCID_OP_PROFX is not loaded.
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DELETE_PROFILE(
		#ICCID_OP_PROFX,
		NO_PARAM**)**
			#R_DELETE_PROFILE_ICCID_ISDP_NOTFOUND
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::=
			profileInfoListOk: {
			#PROFILE_INFO1_DISABLED,
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

Test Sequence #03 Error: Delete Profile (by ISD-P AID) not possible when this Operational Profile is in Enabled state

The purpose of this test is to ensure that it is NOT possible to delete an Operational Profile in Enabled state.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 corresponds to <ISD_P_AID2>.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DELETE_PROFILE(
		NO_PARAM,
		<ISD_P_AID2>)
			#R_DELETE_PROFILE_NOTDISABLESTATE
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::= profileInfoListOk: {
			#PROFILE_INFO1_DISABLED,
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

Test Sequence #04 Error: Delete Profile (by ICCID) not possible when this Operational Profile is in Enabled state

The purpose of this test is to ensure that it is NOT possible to delete an Operational Profile in Enabled state.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		

IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DELETE_PROFILE(
		#ICCID_OP_PROF2,
		NO_PARAM**)**
			#R_DELETE_PROFILE_NOTDISABLESTATE
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::=
			profileInfoListOk: {
			#PROFILE_INFO1_DISABLED,
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

Test Sequence #05 Error: Delete Profile (by ISD-P AID) not possible when PPR2 is set

The purpose of this test is to ensure that it is NOT possible to delete an Operational Profile with the Policy Rule “Deletion of this Profile is not allowed”.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL3 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 corresponds to <ISD_P_AID3>.
eUICC	The Nickname of the PROFILE_OPERATIONAL3 is equal to #NICKNAME3.
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DELETE_PROFILE(
		NO_PARAM,
		<ISD_P_AID3>)
			#R_DELETE_PROFILE_DISALLOWEDBYPOLICY
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::=
			profileInfoListOk: {
			#PROFILE_INFO1_DISABLED,
			#PROFILE_INFO2_ENABLED, #PROFILE_INFO3
			}

			SW=0x9000

Test Sequence #06 Error: Delete Profile (by ICCID) not possible when PPR2 is set

The purpose of this test is to ensure that it is NOT possible to delete an Operational Profile with the Policy Rule “Deletion of this Profile is not allowed”.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL3 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.
eUICC	The Nickname of the PROFILE_OPERATIONAL3 is equal to #NICKNAME3.
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_DELETE_PROFILE(
		#ICCID_OP_PROF3,
		NO_PARAM)
			#R_DELETE_PROFILE_DISALLOWEDBYPOLICY
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::=
			profileInfoListOk: {
			#PROFILE_INFO1_DISABLED,
			#PROFILE_INFO2_ENABLED,
			#PROFILE_INFO3
			}
			SW=0x9000

4.2.24 ES10c (LPA -- eUICC): eUICCMemoryReset

4.2.24.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 2.4.4
- Section 2.9.2
- Section 3.0.1
- Section 3.3.2
- Section 3.5
- Section 5.7.8, 5.7.18, 5.7.19

4.2.24.2 Test Cases

4.2.24.2.1 TC_eUICC_ES10c.eUICCMemoryReset

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.
eUICC	The Default SM-DP+ Address #TEST_DP_ADDRESS1 has been set on the ISD-R.

Test Sequence #01 Nominal: Reset All Operational Profiles (without Enabled Profile)

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.
eUICC	The Nickname of the PROFILE_OPERATIONAL3 is equal to #NICKNAME3.
eUICC	No Notification is stored in the eUICC's Pending Notifications List.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(#GET_EUICC_INFO2)	Retrieve free non-volatile memory value (tag 0x82) from <EXT_CARD_RESOURCE> in EUICCInfo2 as <FREE_MEM_OP_PROF_INSTALLED>
2	S_LPAd → eUICC	MTD_STORE_DATA(#EUICC_MEMORY_RESET_OP_PRO)	#R_EUICC_MEMORY_RESET_OK SW=0x9000
3	S_LPAd → eUICC	MTD_STORE_DATA(#GET_PROFILES_INFO_ALL)	response ProfileInfoListResponse::= profileInfoListOk: { } SW=0x9000
4	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_DE1 SW = 0x9000
5	S_LPAd → eUICC	MTD_STORE_DATA(#GET_EUICC_INFO2)	Retrieve free non-volatile memory value (tag 0x82) from <EXT_CARD_RESOURCE> in EUICCInfo2 as <FREE_MEMORY_NO_PROFILE>

			Verify that <FREE_MEM_OP_PROF_INSTALLED> is lower than <FREE_MEMORY_NO_PROFILE>
6	S_LPAd → eUICC	MTD_STORE_DATA(#GET_EUICC_CONFIGURED_DATA)	#R_ES10a_GECA_DS_DP_1 SW = 0x9000

Test Sequence #02 Nominal: Reset All Operational Profiles (with Enabled Profile), SEP

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	No Notification is stored in the eUICC's Pending Notifications List.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(#EUICC_MEMORY_RESET_OP_PRO)	#R_EUICC_MEMORY_RESET_OK SW=0x91XX
2	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“UICC Reset”)
3	Repeat IC1 and IC2		
4	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_DE1 SW = 0x9000 Note : A Disable Notification for PROFILE_OPERATIONAL1 MAY be also present in the response
5	S_LPAd → eUICC	MTD_STORE_DATA(#GET_RAT)	#R_DEFAULT_RAT SW = 0x9000
6	S_LPAd → eUICC	MTD_STORE_DATA(#GET_PROFILES_INFO_ALL)	response ProfileInfoListResponse::= profileInfoListOk: { } SW=0x9000

Test Sequence #03 Nominal: Reset the Default SM-DP+ Address only

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.

eUICC	The Nickname of the PROFILE_OPERATIONAL3 is equal to #NICKNAME3.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(#EUICC_MEMORY_RESET_DEF_SMDPADDRESS)	#R_EUICC_MEMORY_RESET_OK SW=0x9000
2	S_LPAd → eUICC	MTD_STORE_DATA(#GET_PROFILES_INFO_ALL)	response ProfileInfoListResponse::= profileInfoListOk: { #PROFILE_INFO1_DISABLED, #PROFILE_INFO3 } SW=0x9000
3	S_LPAd → eUICC	MTD_STORE_DATA(#GET_EUICC_CONFIGURED_DATA)	#R_ES10a_GECA_DS SW = 0x9000

Test Sequence #04 Nominal: Reset All Operational Profiles and the Default SM-DP+ Address

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(#EUICC_MEMORY_RESET)	#R_EUICC_MEMORY_RESET_OK SW=0x9000
2	S_LPAd → eUICC	MTD_STORE_DATA(#GET_PROFILES_INFO_ALL)	response ProfileInfoListResponse::= profileInfoListOk: { } SW=0x9000
3	S_LPAd → eUICC	MTD_STORE_DATA(#GET_EUICC_CONFIGURED_DATA)	#R_ES10a_GECA_DS SW = 0x9000

Test Sequence #05 Nominal: eUICC Memory Reset, one Operational Profile Enabled, proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	No Notification is stored in the eUICC's Pending Notifications List.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC4	Do not send FETCH command		
1	S_LPAd → eUICC	MTD_STORE_DATA(#EUICC_MEMORY_RESET_OP_PRO)	#R_EUICC_MEMORY_RESET_OK SW=0x91YY
2	S_Device →eUICC	FETCH 'YY'	REFRESH Command (“UICC Reset”)
3	Repeat IC1 and IC2		

4	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_DE1
			SW = 0x9000
			NOTE : A Disable Notification for PROFILE_OPERATIONAL1 MAY be also present in the response.

5	S_LPAd → eUICC	MTD_STORE_DATA(#GET_RAT)	#R_DEFAULT_RAT
			SW = 0x9000

6	S_LPAd → eUICC	MTD_STORE_DATA(#GET_PROFILES_INFO_ALL)	response ProfileInfoListResponse::= profileInfoListOk : {
			}
			SW=0x9000

Test Sequence #06 Nominal: eUICC Memory Reset (with Enabled Profile) while proactive session is ongoing with Terminal Response outstanding – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.

eUICC	No Notification is stored in the eUICC's Pending Notifications List.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
1	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
2	S_LPAd → eUICC	MTD_STORE_DATA(#EUICC_MEMORY_RESET_OP_PRO)	#R_EUICC_MEMORY_RESET_OK SW=0x9000
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x91YY
4	S_Device →eUICC	FETCH 'YY'	REFRESH Command (“UICC Reset”)
5	Repeat IC1 and IC2		

6	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_DE1
			SW = 0x9000
			NOTE : A Disable Notification for PROFILE_OPERATIONAL1 MAY be also present in the response.

7	S_LPAd → eUICC	MTD_STORE_DATA(#GET_RAT)	#R_DEFAULT_RAT
			SW = 0x9000

8	S_LPAd → eUICC	MTD_STORE_DATA(#GET_PROFILES_INFO_ALL)	response ProfileInfoListResponse::= profileInfoListOk : {
			}
			SW=0x9000

4.2.24.2.2 TC_eUICC_ES10c.eUICCMemoryReset_ErrorCases

Test Sequence #01 Error: eUICC Memory Reset while proactive session is ongoing – catBusy supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC4	Do not send FETCH command		
1	S_LPAd → eUICC	MTD_STORE_DATA(#EUICC_MEMORY_RESET_OP_PRO)	resp EuiccMemoryResetResponse::= { resetResult catBusy } SW=0x9000 or 0x91XX
2	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
4	S_LPAd → eUICC	MTD_STORE_DATA(#GET_PROFILES_INFO_ALL)	response ProfileInfoListResponse::= profileInfoListOk : { #PROFILE_INFO1, #PROFILE_INFO2 } SW=0x9000

Test Sequence #02 Error: Nothing to delete

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	No Profile is loaded on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(#EUICC_MEMORY_RESET_OP_PRO)	resp EuiccMemoryResetResponse::= { resetResult nothingToDelete } SW=0x9000

4.2.24.2.3 TC_eUICC_ES10c.eUICCMemoryReset_MEPA1

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	eUICC is MEP capable
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.
eUICC	The Default SM-DP+ Address #TEST_DP_ADDRESS1 has been set on the ISD-R.

Test Sequence #01 Nominal: Reset All Operational Profiles (without Enabled Profile)

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.
eUICC	No Notification is stored in the eUICC's Pending Notifications List.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(#GET_EUICC_INFO2)	Retrieve free non-volatile memory value (tag 0x82) from <EXT_CARD_RESOURCE> in EUICCInfo2 as <FREE_MEM_OP_PROF_INSTALLED>

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#EUICC_MEMORY_RESET_OP_PRO)
			#R_EUICC_MEMORY_RESET_OK
			SW=0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(<NO_PARAM>,	response ProfileInfoListResponse::= profileInfoListOk: {

		<NO_PARAM>))	} SW=0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(
		#LIST_NOTIF_ALL)
			#R_LIST_NOTIF_DE1_DE2
			SW = 0x9000

5	S_LPAd → eUICC	MTD_STORE_DATA(#GET_EUICC_INFO2)	Retrieve free non-volatile memory value (tag 0x82) from <EXT_CARD_RESOURCE> in EUICCInfo2 as <FREE_MEMORY_NO_PROFILE> Verify that <FREE_MEM_OP_PROF_INSTALLED> is lower than <FREE_MEMORY_NO_PROFILE>

6	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_EUICC_CONFIGURED_DATA)
			#R_ES10a_GECA_DS_DP_1
			SW = 0x9000

Test Sequence #02 Nominal: Reset the Default SM-DP+ Address only

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.
eUICC	The Nickname of the PROFILE_OPERATIONAL3 is equal to #NICKNAME3.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#EUICC_MEMORY_RESET_DEF_SMDPADDRESS)
			#R_EUICC_MEMORY_RESET_OK
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::= profileInfoListOk: {
			#PROFILE_INFO1_DISABLED,
			#PROFILE_INFO3
			}
			SW=0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_EUICC_CONFIGURED_DATA)
			#R_ES10a_GECA_DS
			SW = 0x9000

Test Sequence #03 Nominal: Reset All Operational Profiles and the Default SM-DP+ Address

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#EUICC_MEMORY_RESET)
			#R_EUICC_MEMORY_RESET_OK
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::= profileInfoListOk: {
			}
			SW=0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_EUICC_CONFIGURED_DATA)
			#R_ES10a_GECA_DS
			SW = 0x9000

Test Sequence #04 Nominal: Reset All Operational Profiles (with 2 Enabled Profile)

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 2.
eUICC	No Notification is stored in the eUICC's Pending Notifications List.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#EUICC_MEMORY_RESET_OP_PRO)
			#R_EUICC_MEMORY_RESET_OK
			SW=0x91XX

2	S_Device →eUICC	FETCH 'XX'	LSI COMMAND ("UICC Platform Reset")
3	Repeat IC1 to IC4		

4	S_LPAd → eUICC	MTD_STORE_DATA(
		#LIST_NOTIF_ALL)
			#R_LIST_NOTIF_DE1_DE2
			SW = 0x9000
			NOTE : A Disable Notification for PROFILE_OPERATIONAL1 and PROFILE_OPERATIONAL2 MAY be also present in the response.

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			}
			SW=0x9000

Test Sequence #05 Nominal: Reset the Default SM-DP+ Address only (with 2 Enabled Profile)

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL3 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 is Enabled on the eUICC on Port 2.
eUICC	The Nickname of the PROFILE_OPERATIONAL3 is equal to #NICKNAME3.
eUICC	No Notification is stored in the eUICC's Pending Notifications List.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#EUICC_MEMORY_RESET_DEF_SMDPADDRESS)
			#R_EUICC_MEMORY_RESET_OK
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO3_ENABLED;
			}
			SW=0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_EUICC_CONFIGURED_DATA)
			#R_ES10a_GECA_DS
			SW = 0x9000

Test Sequence #06 Nominal: eUICC Memory Reset, 1 Operational Profile Enabled, proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	No Notification is stored in the eUICC's Pending Notifications List.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(0)		
IC8	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#EUICC_MEMORY_RESET_OP_PRO)
			#R_EUICC_MEMORY_RESET_OK
			SW=0x91YY

2	S_Device →eUICC	FETCH 'YY'	LSI COMMAND ("UICC Platform Reset")
3	Repeat IC1 to IC3		
4	Repeat IC8		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		#LIST_NOTIF_ALL)
			#R_LIST_NOTIF_DE1
			SW = 0x9000
			NOTE : A Disable Notification for PROFILE_OPERATIONAL1 MAY be also present in the response.

6	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(	response ProfileInfoListResponse::= profileInfoListOk : {

		<NO_PARAM>, <NO_PARAM>))	} SW=0x9000

Test Sequence #07 Nominal: eUICC Memory Reset, 2 Operational Profile Enabled, 2 proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 2.
eUICC	No Notification is stored in the eUICC's Pending Notifications List.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(2)		
IC8	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91YY
IC9	Do not send FETCH command		
IC10	PROC_MEP_LSI_MULTIPLEXING(0)		
IC11	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#EUICC_MEMORY_RESET_OP_PRO)
			#R_EUICC_MEMORY_RESET_OK
			SW=0x91ZZ

2	S_Device →eUICC	FETCH 'ZZ'	LSI COMMAND (“UICC Platform Reset”)
3	Repeat IC1 to IC3		
4	Repeat IC11		

5	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_DE1_DE2
			SW = 0x9000
			NOTE : A Disable Notification for PROFILE_OPERATIONAL1 and PROFILE_OPERATIONAL2 MAY be also present in the response.

6	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			}
			SW=0x9000

Test Sequence #08 Nominal: eUICC Memory Reset (with Enabled Profile) while proactive session is ongoing with Terminal Response outstanding – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL3 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC on Port 2.
eUICC	No Notification is stored in the eUICC's Pending Notifications List.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify ‘LSI Support’ is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, “010203”, 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX

IC6	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
IC7	PROC_MEP_LSI_MULTIPLEXING(0)		
IC8	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#EUICC_MEMORY_RESET_OP_PRO)
			#R_EUICC_MEMORY_RESET_OK
			SW=0x91YY

2	S_Device →eUICC	FETCH 'YY'	LSI COMMAND (“UICC Platform Reset”)
3	Repeat IC1 to IC3		
4	Repeat IC8		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		#LIST_NOTIF_ALL)
			#R_LIST_NOTIF_DE1
			SW = 0x9000
			NOTE : A Disable Notification for PROFILE_OPERATIONAL1 MAY be also present in the response.

6	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			}
			SW=0x9000

Test Sequence #09 Nominal: eUICC Memory Reset (with 2 Enabled Profile) while 2 proactive session is ongoing with Terminal Response outstanding – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 1.
eUICC	The PROFILE_OPERATIONAL3 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 is Enabled on the eUICC on Port 2.
eUICC	No Notification is stored in the eUICC's Pending Notifications List.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify ‘LSI Support’ is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2,	Verify <MEP_MODE> = 01, Verify

		#IUT_MEP_LSI_OPTIONS, “010203”, 2)	<MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC1_MEP_LSI_MULTIPLEXING(1)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
IC7	PROC_MEP_LSI_MULTIPLEXING(2)		
IC8	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91YY
IC9	S_Device → eUICC	FETCH 'YY'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
IC10	PROC_MEP_LSI_MULTIPLEXING(0)		
IC11	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#EUICC_MEMORY_RESET_OP_PRO)
			#R_EUICC_MEMORY_RESET_OK
			SW=0x91ZZ

3	S_Device →eUICC	FETCH 'ZZ'	LSI COMMAND (“UICC Platform Reset”)
4	Repeat IC1 to IC3		
5	Repeat IC11		

6	S_LPAd → eUICC	MTD_STORE_DATA(
		#LIST_NOTIF_ALL)
			#R_LIST_NOTIF_DE1
			SW = 0x9000
			NOTE : A Disable Notification for PROFILE_OPERATIONAL1 and PROFILE_OPERATIONAL3 MAY be also present in the response.

7	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			}
			SW=0x9000

4.2.24.2.4 TC_eUICC_ES10c.eUICCMemoryReset_ErrorCases_MEPA1

Test Sequence #01 Error: Nothing to delete

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	No Profile is loaded on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "010203", 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#EUICC_MEMORY_RESET_OP_PRO)
			resp EuiccMemoryResetResponse::= {
			resetResult nothingToDelete
			}
			SW=0x9000

Test Sequence #02 Error: eUICC Memory Reset while 2 proactive session is ongoing – catBusy supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR>

			Verify ‘LSI Support’ is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, “010203”, 2)	Verify <MEP_MODE> = 01, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(2)		
IC8	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91YY
IC9	Do not send FETCH command		
IC10	PROC_MEP_LSI_MULTIPLEXING(0)		
IC11	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(#EUICC_MEMORY_RESET_OP_PRO)	resp EuiccMemoryResetResponse::= { resetResult catBusy } SW=0x9000
2	PROC_MEP_LSI_MULTIPLEXING(2)		
3	S_Device → eUICC	FETCH 'YY'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
4	S_Device → eUICC	Send Terminal Response	SW=0x9000
5	PROC_MEP_LSI_MULTIPLEXING(1)		
6	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
7	S_Device → eUICC	Send Terminal Response	SW=0x9000

8	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

Test Sequence #03 Error: VOID

4.2.24.2.5 TC_eUICC_ES10c.eUICCMemoryReset_MEPA2

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	eUICC is MEP capable
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.
eUICC	The Default SM-DP+ Address #TEST_DP_ADDRESS1 has been set on the ISD-R.

Test Sequence #01 Nominal: Reset All Operational Profiles (without Enabled Profile)

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.
eUICC	No Notification is stored in the eUICC's Pending Notifications List.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(#GET_EUICC_INFO2)	Retrieve free non-volatile memory value (tag 0x82) from <EXT_CARD_RESOURCE> in EUICCInfo2 as

			<FREE_MEM_OP_PROF_INSTALLED>
2	S_LPAd → eUICC	MTD_STORE_DATA(#EUICC_MEMORY_RESET_OP_PRO)	#R_EUICC_MEMORY_RESET_OK SW=0x9000
3	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(<NO_PARAM>, <NO_PARAM>))	response ProfileInfoListResponse::= profileInfoListOk: { } SW=0x9000
4	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_DE1_DE2 SW = 0x9000
5	S_LPAd → eUICC	MTD_STORE_DATA(#GET_EUICC_INFO2)	Retrieve free non-volatile memory value (tag 0x82) from <EXT_CARD_RESOURCE> in EUICCInfo2 as <FREE_MEMORY_NO_PROFILE> Verify that <FREE_MEM_OP_PROF_INSTALLED> is lower than <FREE_MEMORY_NO_PROFILE>
6	S_LPAd → eUICC	MTD_STORE_DATA(#GET_EUICC_CONFIGURED_DATA)	#R_ES10a_GECA_DS_DP_1 SW = 0x9000

Test Sequence #02 Nominal: Reset the Default SM-DP+ Address only

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.
eUICC	The Nickname of the PROFILE_OPERATIONAL3 is equal to #NICKNAME3.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify

			<MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#EUICC_MEMORY_RESET_DEF_SMDPADDRESS)
			#R_EUICC_MEMORY_RESET_OK
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::= profileInfoListOk: {
			#PROFILE_INFO1_DISABLED,
			#PROFILE_INFO3
			}
			SW=0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_EUICC_CONFIGURED_DATA)
			#R_ES10a_GECA_DS
			SW = 0x9000

Test Sequence #03 Nominal: Reset All Operational Profiles and the Default SM-DP+ Address

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#EUICC_MEMORY_RESET)
			#R_EUICC_MEMORY_RESET_OK
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		<NO_PARAM>,
		<NO_PARAM>))
			response
			ProfileInfoListResponse::=
			profileInfoListOk: {
			}
			SW=0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_EUICC_CONFIGURED_DATA)
			#R_ES10a_GECA_DS
			SW = 0x9000

Test Sequence #04 Nominal: Reset All Operational Profiles (with 2 Enabled Profile)

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC.
eUICC	No Notification is stored in the eUICC's Pending Notifications List.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#EUICC_MEMORY_RESET_OP_PRO)
			#R_EUICC_MEMORY_RESET_OK
			SW=0x91XX

2	S_Device →eUICC	FETCH 'XX'	LSI COMMAND ("UICC Platform Reset")
3	Repeat IC1 to IC4		

4	S_LPAd → eUICC	MTD_STORE_DATA(
		#LIST_NOTIF_ALL)
			#R_LIST_NOTIF_DE1_DE2
			SW = 0x9000
			NOTE : A Disable Notification for PROFILE_OPERATIONAL1 and

			PROFILE_OPERATIONAL2 MAY be also present in the response.

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			}
			SW=0x9000

Test Sequence #05 Nominal: Reset the Default SM-DP+ Address only (with 2 Enabled Profile)

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 is Enabled on the eUICC.
eUICC	The Nickname of the PROFILE_OPERATIONAL3 is equal to #NICKNAME3.
eUICC	No Notification is stored in the eUICC's Pending Notifications List.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#EUICC_MEMORY_RESET_DEF_SMDPADDRESS)
			#R_EUICC_MEMORY_RESET_OK
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO3_ENABLED;
			}
			SW=0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(#GET_EUICC_CONFIGURED _DATA)	#R_ES10a_GECA_DS
			SW = 0x9000

Test Sequence #06 Nominal: eUICC Memory Reset, 1 Operational Profile Enabled, proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on target port <MEP-A2_TARGET_ESIM_PORT> assigned by the eUICC.
eUICC	No Notification is stored in the eUICC's Pending Notifications List.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(<MEP-A2_TARGET_ESIM_PORT>)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(0)		
IC8	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(#EUICC_MEMORY_RESET_OP_PRO)	#R_EUICC_MEMORY_RESET_OK
			SW=0x91YY

2	S_Device →eUICC	FETCH 'YY'	LSI COMMAND ("UICC Platform Reset")
3	Repeat IC1 to IC3		
4	Repeat IC8		

5	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_DE1
			SW = 0x9000

			NOTE : A Disable Notification for PROFILE_OPERATIONAL1 MAY be also present in the response.

6	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			}
			SW=0x9000

Test Sequence #07 Nominal: eUICC Memory Reset, 2 Operational Profile Enabled, 2 proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC.
eUICC	No Notification is stored in the eUICC's Pending Notifications List.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(2)		
IC8	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91YY
IC9	Do not send FETCH command		
IC10	PROC_MEP_LSI_MULTIPLEXING(0)		

IC11	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(#EUICC_MEMORY_RESET_OP_PRO)	#R_EUICC_MEMORY_RESET_OK SW=0x91ZZ
2	S_Device →eUICC	FETCH 'ZZ'	LSI COMMAND (“UICC Platform Reset”)
3	Repeat IC1 to IC3		
4	Repeat IC11		

5	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_DE1_DE2
			SW = 0x9000
			NOTE : A Disable Notification for PROFILE_OPERATIONAL1 and PROFILE_OPERATIONAL2 MAY be also present in the response.

6	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(<NO_PARAM>, <NO_PARAM>))	response ProfileInfoListResponse::= profileInfoListOk : { } SW=0x9000

Test Sequence #08 Nominal: eUICC Memory Reset (with Enabled Profile) while proactive session is ongoing with Terminal Response outstanding – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on <MEP-A2_TARGET_ESIM_PORT> assigned by the eUICC.
eUICC	The PROFILE_OPERATIONAL3 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.
eUICC	No Notification is stored in the eUICC's Pending Notifications List.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify ‘LSI Support’ is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, “020103”, 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify

			<MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(<MEP-A2_TARGET_ESIM_PORT>)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
IC7	PROC_MEP_LSI_MULTIPLEXING(0)		
IC8	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(#EUICC_MEMORY_RESET_OP_PRO)	#R_EUICC_MEMORY_RESET_OK
			SW=0x91YY

2	S_Device →eUICC	FETCH 'YY'	LSI COMMAND (“UICC Platform Reset”)
3	Repeat IC1 to IC3		
4	Repeat IC8		

5	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_DE1
			SW = 0x9000
			NOTE : A Disable Notification for PROFILE_OPERATIONAL1 MAY be also present in the response.

6	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(<NO_PARAM>, <NO_PARAM>))	response ProfileInfoListResponse::= profileInfoListOk : {
			}
			SW=0x9000

Test Sequence #09 Nominal: eUICC Memory Reset (with 2 Enabled Profile) while 2 proactive session is ongoing with Terminal Response outstanding – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 is Enabled on the eUICC.
eUICC	No Notification is stored in the eUICC's Pending Notifications List.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
IC7	PROC_MEP_LSI_MULTIPLEXING(2)		
IC8	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91YY
IC9	S_Device → eUICC	FETCH 'YY'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
IC10	PROC_MEP_LSI_MULTIPLEXING(0)		
IC11	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#EUICC_MEMORY_RESET_OP_PRO)
			#R_EUICC_MEMORY_RESET_OK
			SW=0x91ZZ

2	S_Device →eUICC	FETCH 'ZZ'	LSI COMMAND ("UICC Platform Reset")
3	Repeat IC1 to IC3		
4	Repeat IC11		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		#LIST_NOTIF_ALL)
			#R_LIST_NOTIF_DE1
			SW = 0x9000
			NOTE : A Disable Notification for PROFILE_OPERATIONAL1 and PROFILE_OPERATIONAL3 MAY be also present in the response.

6	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO (	response ProfileInfoListResponse::= profileInfoListOk : {

		<NO_PARAM>, <NO_PARAM>))	} SW=0x9000

4.2.24.2.6 TC_eUICC_ES10c.eUICCMemoryReset_ErrorCases_MEPA2

Test Sequence #01 Error: Nothing to delete

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	No Profile is loaded on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#EUICC_MEMORY_RESET_OP_PRO)
			resp EuiccMemoryResetResponse::= {
			resetResult nothingToDelete
			}
			SW=0x9000

Test Sequence #02 Error: eUICC Memory Reset while 2 proactive session is ongoing – catBusy supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "020103", 2)	Verify <MEP_MODE> = 02, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(1)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(2)		
IC8	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91YY
IC9	Do not send FETCH command		
IC10	PROC_MEP_LSI_MULTIPLEXING(0)		
IC11	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(#EUICC_MEMORY_RESET_OP_PRO)	resp EuiccMemoryResetResponse::= { resetResult catBusy } SW=0x9000
2	PROC_MEP_LSI_MULTIPLEXING(2)		
3	S_Device → eUICC	FETCH 'YY'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
4	S_Device → eUICC	Send Terminal Response	SW=0x9000
5	PROC_MEP_LSI_MULTIPLEXING(1)		
6	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
7	S_Device → eUICC	Send Terminal Response	SW=0x9000

8	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1;
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

Test Sequence #03 Error: VOID

4.2.24.2.7 TC_eUICC_ES10c.eUICCMemoryReset_MEPB

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	eUICC is MEP capable
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.
eUICC	The Default SM-DP+ Address #TEST_DP_ADDRESS1 has been set on the ISD-R.

Test Sequence #01 Nominal: Reset All Operational Profiles (without Enabled Profile)

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Disabled on the eUICC.
eUICC	No Notification is stored in the eUICC's Pending Notifications List.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030102", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		

IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(#GET_EUICC_INFO2)	Retrieve free non-volatile memory value (tag 0x82) from <EXT_CARD_RESOURCE> in EUICCInfo2 as <FREE_MEM_OP_PROF_INSTALLED>
2	S_LPAd → eUICC	MTD_STORE_DATA(#EUICC_MEMORY_RESET_OP_PRO)	#R_EUICC_MEMORY_RESET_OK SW=0x9000
3	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(<NO_PARAM>, <NO_PARAM>))	response ProfileInfoListResponse::= profileInfoListOk: { } SW=0x9000
4	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_DE1_DE2 SW = 0x9000
5	S_LPAd → eUICC	MTD_STORE_DATA(#GET_EUICC_INFO2)	Retrieve free non-volatile memory value (tag 0x82) from <EXT_CARD_RESOURCE> in EUICCInfo2 as <FREE_MEMORY_NO_PROFILE> Verify that <FREE_MEM_OP_PROF_INSTALLED> is lower than <FREE_MEMORY_NO_PROFILE>
6	S_LPAd → eUICC	MTD_STORE_DATA(#GET_EUICC_CONFIGURED_DATA)	#R_ES10a_GECA_DS_DP_1 SW = 0x9000

Test Sequence #02 Nominal: Reset the Default SM-DP+ Address only

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.
eUICC	The Nickname of the PROFILE_OPERATIONAL3 is equal to #NICKNAME3.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (	Verify <MEP_MODE> = 03,

		2, #IUT_MEP_LSI_OPTIONS, “030102”, 2)	Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#EUICC_MEMORY_RESET_DEF_SMDPADDRESS)
			#R_EUICC_MEMORY_RESET_OK
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::= profileInfoListOk: {
			#PROFILE_INFO1_DISABLED,
			#PROFILE_INFO3
			}
			SW=0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_EUICC_CONFIGURED_DATA)
			#R_ES10a_GECA_DS
			SW = 0x9000

Test Sequence #03 Nominal: Reset All Operational Profiles and the Default SM-DP+ Address

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify ‘LSI Support’ is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, “030102”, 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		

IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#EUICC_MEMORY_RESET)
			#R_EUICC_MEMORY_RESET_OK
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response
			ProfileInfoListResponse::=
			profileInfoListOk: {
			}
			SW=0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_EUICC_CONFIGURED_DATA)
			#R_ES10a_GECA_DS
			SW = 0x9000

Test Sequence #04 Nominal: Reset All Operational Profiles (with 2 Enabled Profile)

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the Euicc on Port 0.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 1.
eUICC	No Notification is stored in the eUICC's Pending Notifications List.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030102", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#EUICC_MEMORY_RESET_OP_PRO)
			#R_EUICC_MEMORY_RESET_OK
			SW=0x91XX

2	S_Device →eUICC	FETCH 'XX'	LSI COMMAND ("UICC Platform Reset")
3	Repeat IC1 to IC4		

4	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_DE1_DE2
			SW = 0x9000
			NOTE : A Disable Notification for PROFILE_OPERATIONAL1 and PROFILE_OPERATIONAL2 MAY be also present in the response.

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			}
			SW=0x9000

Test Sequence #05 Nominal: Reset the Default SM-DP+ Address only (with 2 Enabled Profile)

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL3 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 is Enabled on the eUICC on Port 1.
eUICC	The Nickname of the PROFILE_OPERATIONAL3 is equal to #NICKNAME3.
eUICC	No Notification is stored in the eUICC's Pending Notifications List.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030102", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#EUICC_MEMORY_RESET_DEF_SMDPADDRESS)
			#R_EUICC_MEMORY_RESET_OK
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(<NO_PARAM>,	response ProfileInfoListResponse::= profileInfoListOk : { #PROFILE_INFO1,

		<NO_PARAM>))	#PROFILE_INFO3_MEPB; } SW=0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA(#GET_EUICC_CONFIGURED_DATA)	#R_ES10a_GECA_DS
			SW = 0x9000

Test Sequence #06 Nominal: eUICC Memory Reset, 1 Operational Profile Enabled, proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	No Notification is stored in the eUICC's Pending Notifications List.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030102", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
NOTE: for clarity, the current Port is already Port 0			
IC4	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC5	Do not send FETCH command		
IC6	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(#EUICC_MEMORY_RESET_OP_PRO)	#R_EUICC_MEMORY_RESET_OK
			SW=0x91YY

2	S_Device →eUICC	FETCH 'YY'	LSI COMMAND ("UICC Platform Reset")
3	Repeat IC1 to IC3		
4	Repeat IC6		

5	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_DE1
			SW = 0x9000
			NOTE : A Disable Notification for PROFILE_OPERATIONAL1 MAY be also present in the response.

6	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::= profileInfoListOk : {
			}
			SW=0x9000

Test Sequence #07 Nominal: eUICC Memory Reset, 2 Operational Profile Enabled, 2 proactive session is ongoing – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 1.
eUICC	No Notification is stored in the eUICC's Pending Notifications List.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030102", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(0)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(1)		
IC8	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91YY
IC9	Do not send FETCH command		

IC10	PROC_MEP_LSI_MULTIPLEXING(0)		
IC11	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#EUICC_MEMORY_RESET_OP_PRO)
			#R_EUICC_MEMORY_RESET_OK
			SW=0x91ZZ

2	S_Device →eUICC	FETCH 'ZZ'	LSI COMMAND (“UICC Platform Reset”)
3	Repeat IC1 to IC3		
4	Repeat IC11		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		#LIST_NOTIF_ALL)
			#R_LIST_NOTIF_DE1_DE2
			SW = 0x9000
			NOTE : A Disable Notification for PROFILE_OPERATIONAL1 and PROFILE_OPERATIONAL2 MAY be also present in the response.

6	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse ::= profileInfoListOk : {
			}
			SW=0x9000

Test Sequence #08 Nominal: eUICC Memory Reset (with Enabled Profile) while proactive session is ongoing with Terminal Response outstanding – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL3 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 is Disabled on the eUICC.
eUICC	No Notification is stored in the eUICC's Pending Notifications List.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify ‘LSI Support’ is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, “030102”, 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify

			<MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(0)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
IC7	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(#EUICC_MEMORY_RESET_OP_PRO)	#R_EUICC_MEMORY_RESET_OK
			SW=0x9000

2	S_Device → eUICC	TERMINAL RESPONSE	SW=0x91YY
3	S_Device →eUICC	FETCH 'YY'	LSI COMMAND (“UICC Platform Reset”)
4	Repeat IC1 to IC4		
5	Repeat IC7		

6	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_DE1
			SW = 0x9000
			NOTE : A Disable Notification for PROFILE_OPERATIONAL1 MAY be also present in the response.

7	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(<NO_PARAM>, <NO_PARAM>))	response ProfileInfoListResponse ::= profileInfoListOk : {
			}
			SW=0x9000

Test Sequence #09 Nominal: eUICC Memory Reset (with 2 Enabled Profile) while 2 proactive session is ongoing with Terminal Response outstanding – catBusy not supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.
eUICC	The PROFILE_OPERATIONAL3 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL3 is Enabled on the eUICC on Port 1.
eUICC	No Notification is stored in the eUICC's Pending Notifications List.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030102", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(0)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
IC7	PROC_MEP_LSI_MULTIPLEXING(1)		
IC8	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91YY
IC9	S_Device → eUICC	FETCH 'YY'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
IC10	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#EUICC_MEMORY_RESET_OP_PRO)
			#R_EUICC_MEMORY_RESET_OK
			SW=0x9000

2	S_Device → eUICC	TERMINAL RESPONSE	SW=0x91ZZ
3	S_Device →eUICC	FETCH 'ZZ'	LSI COMMAND ("UICC Platform Reset")
4	Repeat IC1 to IC4		
5	Repeat IC10		

6	S_LPAd → eUICC	MTD_STORE_DATA(
		#LIST_NOTIF_ALL)
			#R_LIST_NOTIF_DE1
			SW = 0x9000
			NOTE : A Disable Notification for PROFILE_OPERATIONAL1 and PROFILE_OPERATIONAL3 MAY be also present in the response.

7	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			}
			SW=0x9000

4.2.24.2.8 TC_eUICC_ES10c.eUICCMemoryReset_ErrorCases_MEPB

Test Sequence #01 Error: Nothing to delete

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	No Profile is loaded on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030102", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#EUICC_MEMORY_RESET_OP_PRO)
			resp EuiccMemoryResetResponse::= {
			resetResult nothingToDelete
			}
			SW=0x9000

Test Sequence #02 Error: eUICC Memory Reset while 2 proactive session is ongoing – catBusy supported

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled on the eUICC on Port 0.

eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 is Enabled on the eUICC on Port 1.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	RESET	Extract <ATR> Verify 'LSI Support' is present in <ATR>
IC2	S_Device	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP (2, #IUT_MEP_LSI_OPTIONS, "030102", 2)	Verify <MEP_MODE> = 03, Verify <MEP_LSI_OPTION> = #IUT_MEP_LSI_OPTIONS, Verify <MEP_MAX_LSIS> <= #IUT_MEP_MAX_LSIS
IC3	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP		
IC4	PROC_MEP_LSI_MULTIPLEXING(0)		
IC5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC6	Do not send FETCH command		
IC7	PROC_MEP_LSI_MULTIPLEXING(1)		
IC8	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91YY
IC9	Do not send FETCH command		
IC10	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(#EUICC_MEMORY_RESET_OP_PRO)	resp EuiccMemoryResetResponse::= { resetResult catBusy } SW=0x91YY
2	S_Device → eUICC	FETCH 'YY'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
3	S_Device → eUICC	Send Terminal Response	SW=0x9000
4	PROC_MEP_LSI_MULTIPLEXING(0)		
5	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
6	S_Device → eUICC	Send Terminal Response	SW=0x9000
7	PROC_MEP_LSI_MULTIPLEXING(1)		

8	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		<NO_PARAM>,
		<NO_PARAM>))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_MEPB,
			#PROFILE_INFO2_ENABLED;
			}
			SW=0x9000

Test Sequence #03 Error: VOID

4.2.25 ES10c (LPA -- eUICC): GetEID

4.2.25.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 3.3.1
- Section 5.7.20

4.2.25.2 Test Cases

4.2.25.2.1 TC_eUICC_ES10c.GetEID

Test Sequence #01 Nominal

The purpose of this test is to ensure that it is possible to retrieve the EID.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_EID)
			resp GetEuiccDataResponse ::= {
			eidValue #EID1
			}
			SW=0x9000

Test Sequence #02 Error

The purpose of this test is to ensure that if the provided tagList is invalid or unsupported, the eUICC returns an error status word.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(#GET_EID_INVALID)	No response data return and SW different than 0x9000

4.2.26 ES10c (LPA -- eUICC): SetNickname

4.2.26.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 5.7.21

4.2.26.2 Test Cases

4.2.26.2.1 TC_eUICC_ES10c.SetNickname

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 is loaded on the eUICC.

Test Sequence #01 Nominal: Add a Nickname to a Disabled Operational Profile

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled.
eUICC	The Nickname of the PROFILE_OPERATIONAL1 is empty.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#SET_NICKNAME_OP_PROF1)
			resp SetNicknameResponse ::= {
			setNicknameResult ok
			}
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			…
			profileNickname #NICKNAME2
			…
			}
			}
			SW=0x9000

Test Sequence #02 Nominal: Update a Nickname of a Disabled Operational Profile

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled.
eUICC	The Nickname of the PROFILE_OPERATIONAL1 is equal to #NICKNAME1.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(#SET_NICKNAME_OP_PROF1)	resp SetNicknameResponse ::= { setNicknameResult ok } SW=0x9000
2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(#ICCID_OP_PROF1, NO_PARAM))	resp ProfileInfoListResponse ::= profileInfoListOk :{ { … profileNickname #NICKNAME2 … } } SW=0x9000

Test Sequence #03 Nominal: Remove a Nickname from a Disabled Operational Profile

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled.
eUICC	The Nickname of the PROFILE_OPERATIONAL1 is equal to #NICKNAME1.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(#SET_NICKNAME_EMPTY_OP_PROF1)	resp SetNicknameResponse ::= { setNicknameResult ok } SW=0x9000
2	S_LPAd → eUICC	MTD_STORE_DATA(MTD_GET_PROFILE_INFO(	resp ProfileInfoListResponse ::= profileInfoListOk :{

		#ICCID_OP_PROF1,
		NO_PARAM))
			{
			…
			-- profileNickname SHALL not
			-- be present
			…
			}
			}
			SW=0x9000

Test Sequence #04 Nominal: Remove a non-existing Nickname from a Disabled Operational Profile

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled.
eUICC	The Nickname of the PROFILE_OPERATIONAL1 is empty.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#SET_NICKNAME_EMPTY_OP_PROF1)
			resp SetNicknameResponse ::= {
			setNicknameResult ok
			}
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			…
			-- profileNickname SHALL not
			-- be present
			…
			}
			}
			SW=0x9000

Test Sequence #05 Nominal: Add a Nickname to an Enabled Operational Profile

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled.
eUICC	The Nickname of the PROFILE_OPERATIONAL1 is empty.

This test sequence SHALL be the same as the Test Sequence #01 defined in this section.

Test Sequence #06 Nominal: Update a Nickname of an Enabled Operational Profile

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled.
eUICC	The Nickname of the PROFILE_OPERATIONAL1 is equal to #NICKNAME1.

This test sequence SHALL be the same as the Test Sequence #02 defined in this section.

Test Sequence #07 Nominal: Remove a Nickname from an Enabled Operational Profile

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled.
eUICC	The Nickname of the PROFILE_OPERATIONAL1 is equal to #NICKNAME1.

This test sequence SHALL be the same as the Test Sequence #03 defined in this section.

Test Sequence #08 Nominal: Remove a non-existing Nickname from an Enabled Operational Profile

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Enabled.
eUICC	The Nickname of the PROFILE_OPERATIONAL1 is empty.

This test sequence SHALL be the same as the Test Sequence #04 defined in this section.

Test Sequence #09 Error: ICCID not found

The purpose of this test is to ensure that the method ES10c.SetNickname returns an error in case the targeted Profile does not exist on the eUICC.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled.
eUICC	The Profile identified by the ICCID #ICCID_UNKNOWN is not present on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(#SET_NICKNAME_ICCID_UNKNOWN)	resp SetNicknameResponse ::= { setNicknameResult iccidNotFound } SW=0x9000

4.2.27 ES10b (LPA -- eUICC): GetRAT

4.2.27.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 2.4a.1.2
- Section 2.9.2.1, 2.9.2.3, 2.9.3.3
- Section 5.7.22

4.2.27.2 Test Cases

4.2.27.2.1 TC_eUICC_ES10b.GetRAT

Test Sequence #01 Nominal: Get Default RAT

The purpose of this test is to verify that the eUICC can be configured with a RAT as defined in SGP.22 [2] section 2.9.2.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The EUM has configured the eUICC's RAT as defined in section G.2.4.

Step	**Direction**	**Sequence / Description**	**Expected result**
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(#GET_RAT)	#R_DEFAULT_RAT SW = 0x9000

Test Sequence #02 Nominal: With additional PPARs

The purpose of this test is to verify that the eUICC can be configured with a RAT that contains custom rules reflecting agreements between some Operators and OEMs. After having checked the content of the RAT, Profiles with PPR1 and PPR2 are installed in order to make sure that the eUICC accepts such PPRs.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The EUM has configured the eUICC's RAT as defined in section G.2.5.

eUICC	There is no Profile installed in the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_LPAd → eUICC	MTD_STORE_DATA(#GET_RAT)	#R_RAT_WITH_OTHER_RULES with exact same structure and order
			SW = 0x9000

2	S_LPAd → eUICC	Install PROFILE_OPERATIONAL4	Profile successfully downloaded (i.e. ProfileInstallationResult contains a SuccessResult)
3	S_LPAd → eUICC	Delete PROFILE_OPERATIONAL4	
4	S_LPAd → eUICC	Install PROFILE_OPERATIONAL3	Profile successfully downloaded (i.e. ProfileInstallationResult contains a SuccessResult)

4.2.28 ES10b (LPA -- eUICC): LoadRPMPackage

4.2.28.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 2.10, 2.10.1, 2.10.2
- Section 5.1
- Section 5.7.14a

4.2.28.2 Test Cases

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 is loaded on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 is Disabled.
eUICC	The communication between the S_Device and the eUICC has been initialized and the S_LPAd has selected the ISD-R. Common Mutual Authentication procedure has been successfully executed between the eUICC and the S_SM-DP+ with the CtxParams1 as #CTX_PARAMS1_RPM_ICCID1

4.2.28.2.1 TC_eUICC_ES10b.LoadRPMPackage_EnableProfile

Test Sequence #01 Nominal: RPM Command EnableProfile - no profiles enabled already

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC with #METADATA_OP_PROF1_RPM_CONF_EN.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (enable, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x91XX with the response data MTD_RES_RPR_FOR_SINGLE_CMND (enableResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 0, -- OK response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, NO_PARAM) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG
2	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“UICC Reset”)
3	PROC_EUICC_INITIALIZATION_SEQUENCE		
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
			response ProfileInfoListResponse ::= profileInfoListOk : {
			#PROFILE_INFO1
			}

		NO_PARAM))	SW=0x9000
6	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

7	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #02 Error: RPM Command EnableProfile – ICCID unknown

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been loaded with #METADATA_OP_PROF1_RPM_CONF_EN.
eUICC	The Operational Profile identified by the ICCID #ICCID_OP_PROFX is not loaded.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (enable, <S_TRANSACTION_ID>, #ICCID_OP_PROFX, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (enableResult, <S_TRANSACTION_ID>, #ICCID_OP_PROFX, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, commandError) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG
2	S_LPAd → eUICC	MTD_STORE_DATA(#GET_PROFILES_INFO_ALL)	response ProfileInfoListResponse::= profileInfoListOk : { #PROFILE_INFO1_DISABLED

			} SW=0x9000

Test Sequence #03 Error: RPM Command EnableProfile – SM-DP+ OID not in Managing SM-DP+ List

The purpose of this Test Secuence is to ensure RPM Command EnableProfile is not executed if the SM-DP+ that sent the RPM Command is not included in the Managing SM-DP+ List in the Profile Metadata (rpmConfiguration).

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been loaded on the eUICC with #METADATA_OP_PROF1_RPM_CONF_EN_DP_OID2.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT (MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (enable, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (enableResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, commandError) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA (
		MTD_GET_PROFILE_INFO (
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

Test Sequence #04 Error: RPM Command EnableProfile – Allowed CI Public Key Identifier not matched

The purpose of this test sequence is to ensure RPM Command EnableProfile is not executed if Subject Key Identifier of the CI corresponding to CERT.DPauth.SIG attached to the ongoing session does not match with the Allowed CI Public Key Identifier in the Profile Metadata (rpmConfiguration).

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_OP_PROF1_RPM_CONF_EN_CI_PKI_RAND is loaded on the eUICC.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT (MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (enable, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (enableResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, commandError) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA (
		MTD_GET_PROFILE_INFO (
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

Test Sequence #05 Error: RPM Command EnableProfile – Another Profile currently Enabled

The purpose of this test sequence is to ensure RPM Command EnableProfile is not executed while another Profile is Enabled.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been loaded with #METADATA_OP_PROF1_RPM_CONF_EN.
eUICC	The PROFILE_OPERATIONAL2 with #METADATA_OP_PROF2_RPM_CONF_EN_OWNER_OID1 (without PPR1 present) is loaded and Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (enable, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with unsuccessful enableResult in the response data for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

Test Sequence #06 Error: RPM Command EnableProfile – Profile is not in Disabled state

The purpose of this Test Sequence is to ensure RPM Command EnableProfile is not executed if the target Profile is not in Disabled state.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been loaded on the eUICC with #METADATA_OP_PROF1_RPM_CONF_EN and it is Enabled.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT (MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (enable, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (enableResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, profileNotInDisabledState) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA (
		MTD_GET_PROFILE_INFO (
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1}
			SW=0x9000

Test Sequence #07 Error: RPM Command EnableProfile – disallowed by policy

The purpose of this Test Sequence is to ensure RPM Command EnableProfile is not executed if the currently Enabled Profile cannot be disabled due to Profile Policy Rule.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been loaded on the eUICC with #METADATA_OP_PROF1_RPM_CONF_EN.
eUICC	The PROFILE_OPERATIONAL2 with #METADATA_OP_PROF2_RPM_CONF_EN_OWNER_OID1_PPR1 has been loaded before the PROFILE_OPERATIONAL1 and is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT (MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (enable, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (enableResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, disallowedByPolicy) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA (
		MTD_GET_PROFILE_INFO (
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

Test Sequence #08 Error: RPM Command EnableProfile – Invalid Transaction Id

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been loaded with #METADATA_OP_PROF1_RPM_CONF_EN.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (enable, <INVALID_TRANSACTION_ID>,	SW=0x9000 without response data for all STORE DATA commands except for the last one

		#ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM)) -- The <INVALID_TRANSACTION_ID> SHALL NOT be equal to the Transaction Id in the AuthenticateServerRequest	SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (enableResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 3, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, invalidTransactionId) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG • The <S_TRANSACTION_ID> SHALL be equal to the one set in the AuthenticateServerRequest.

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

Test Sequence #09 Error: RPM Command EnableProfile – Invalid Signature

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been loaded with #METADATA_OP_PROF1_RPM_CONF_EN.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (enable, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM,	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (enableResult,

		NO_PARAM))	<S_TRANSACTION_ID>, #ICCID_OP_PROF1, 3, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, invalidSignature) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

Test Sequence #10 Error: RPM Command EnableProfile – No RPM Session on going

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been loaded with #METADATA_OP_PROF1_RPM_CONF_EN
eUICC	No RPM session is on-going (i.e. No common mutual authentication procedure has been completed for a RPM session)

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (enable, <INVALID_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (NO_PARAM, <S_TRANSACTION_ID>, NO_PARAM,

		-- The <INVALID_TRANSACTION_ID> any Transaction Id.	4, -- error response, LoadRpmPackageErrorCodeNotSigned NO_PARAM, NO_PARAM, NO_PARAM, NO_PARAM, noSession) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

4.2.28.2.2 TC_eUICC_ES10b.LoadRPMPackage_DisableProfile

Test Sequence #01 Nominal: RPM Command DisableProfile – by ICCID

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC with #METADATA_OP_PROF1_RPM_CONF_DI. (PPR1 is not set in the Metadata)
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (disable, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM,	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x91XX with the response data MTD_RES_RPR_FOR_SINGLE_CMND (

		NO_PARAM))	disableResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 0, -- OK response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, NO_PARAM) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG
2	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“UICC Reset”)
3	PROC_EUICC_INITIALIZATION_SEQUENCE		
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse ::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

6	S_Device → eUICC	[SELECT_ICCID]	SW=6A82

Test Sequence #02 Error: RPM Command DisableProfile – ICCID unknown

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been loaded with #METADATA_OP_PROF1_RPM_CONF_DI. (PPR1 is not set in the Metadata)
eUICC	The PROFILE_OPERATIONAL1 has been Enabled.
eUICC	The Operational Profile identified by the ICCID #ICCID_OP_PROFX is not loaded.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (	SW=0x9000 without response data for all STORE DATA commands except for the last one

		disable, <S_TRANSACTION_ID>, #ICCID_OP_PROFX, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM))	SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (disableResult, <S_TRANSACTION_ID>, #ICCID_OP_PROFX, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, commandError) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #03 Error: RPM Command DisableProfile – SM-DP+ OID not in Managing SM-DP+ List

The purpose of this Test Sequence is to ensure RPM Command DisableProfile is not executed if the SM-DP+ that sent the RPM Command is not included in the Managing SM-DP+ List in the Profile Metadata (rpmConfiguration).

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been loaded with #METADATA_OP_PROF1_RPM_CONF_DI_DP_OID2 and is Enabled on the eUICC (PPR1 is not set in the Metadata). NOTE: #S_SM_DP+_OID2 set in the Profile Metadata is different from SM-DP+ OID in CERT.DPauth.SIG attached to the ongoing RSP session.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT (MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (disable, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (disableResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, commandError) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA (
		MTD_GET_PROFILE_INFO (
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #04 Error: RPM Command DisableProfile – Allowed CI Public Key Identifier not matched

The purpose of this test sequence is to ensure RPM Command DisableProfile is not executed if Subject Key Identifier of the CI corresponding to CERT.DPauth.SIG attached to the ongoing session does not match with the Allowed CI Public Key Identifier in the Profile Metadata (rpmConfiguration).

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_OP_PROF1_RPM_CONF_DI_CI_PKI_RAND is loaded and Enabled on the eUICC. (PPR1 is not set in the Metadata)

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT (MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (disable, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (disableResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, commandError) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA (
		MTD_GET_PROFILE_INFO (
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #05 Error: RPM Command DisableProfile – Profile is not in Enabled state

The purpose of this Test Sequence is to ensure RPM Command DisableProfile is not executed if the target Profile is not in Enabled state.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been loaded on the eUICC with #METADATA_OP_PROF1_RPM_CONF_DI and it is in Disabled state.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT (MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (disable, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (disableResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, profileNotInEnabledState) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA (
		MTD_GET_PROFILE_INFO (
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED}
			SW=0x9000

Test Sequence #06 Error: RPM Command DisableProfile – disallowed by policy

The purpose of this Test Sequence is to ensure RPM Command DisableProfile is not executed if the currently Enabled Profile cannot be disabled due to Profile Policy Rule.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been loaded with #METADATA_OP_PROF1_RPM_CONF_DI_PPR1 and is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT (MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (	SW=0x9000 without response data for all STORE DATA commands except for the last one

		disable, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM))	SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (disableResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, disallowedByPolicy) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA (
		MTD_GET_PROFILE_INFO (
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

4.2.28.2.3 TC_eUICC_ES10b.LoadRPMPackage_DeleteProfile

Test Sequence #01 Nominal: RPM Command DeleteProfile – by ICCID

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC with #METADATA_OP_PROF1_RPM_CONF_DE. (PPR2 is not set in the Metadata)

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (delete, <S_TRANSACTION_ID>, #ICCID_OP_PROF1,	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND

		<S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM))	(deleteResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 0, -- OK response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, NO_PARAM) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse ::= profileInfoListOk : {
			}
			SW=0x9000

Test Sequence #02 Error: RPM Command DeleteProfile_ICCID unknown

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been loaded with #METADATA_OP_PROF1_RPM_CONF_DE. (PPR2 is not set in the Metadata)
eUICC	The Operational Profile identified by the ICCID #ICCID_OP_PROFX is not loaded.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (delete, <S_TRANSACTION_ID>, #ICCID_OP_PROFX, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM)	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (deleteResult, <S_TRANSACTION_ID>, #ICCID_OP_PROFX,

		) 	1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, commandError) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

Test Sequence #03 Error: RPM Command DeleteProfile_SM-DP+ OID not in Managing SM-DP+ List

The purpose of this Test Sequence is to ensure RPM Command DeleteProfile is not executed if the SM-DP+ that sends the RPM Command is not included in the Managing SM-DP+ List in the Profile Metadata (rpmConfiguration).

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been loaded with #METADATA_OP_PROF1_RPM_CONF_DE_DP_OID2. (PPR2 is not set in the Metadata) NOTE: #S_SM_DP+_OID2 set in the Profile Metadata is different from SM-DP+ OID in CERT.DPauth.SIG attached to the ongoing RSP session.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT (MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (delete, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM,	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (deleteResult,

		NO_PARAM))	<S_TRANSACTION_ID>, #ICCID_OP_PROF1, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, commandError) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA (
		MTD_GET_PROFILE_INFO (
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

Test Sequence #04 Error: RPM Command DeleteProfile – Allowed CI Public Key Identifier not matched

The purpose of this test sequence is to ensure RPM Command DeleteProfile is not executed if Subject Key Identifier of the CI corresponding to CERT.DPauth.SIG attached to the ongoing session does not match with the Allowed CI Public Key Identifier in the Profile Metadata (rpmConfiguration).

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_OP_PROF1_RPM_CONF_DE_CI_PKI_RAND is loaded on the eUICC. (PPR2 is not set in the Metadata)

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT (MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (delete, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>,	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (

		NO_PARAM, NO_PARAM))	deleteResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, commandError) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA (
		MTD_GET_PROFILE_INFO (
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

Test Sequence #05 Error: RPM Command DeleteProfile – Profile is not in Disabled state

The purpose of this Test Sequence is to ensure RPM Command DeleteProfile is not executed if the target Profile is in Enabled state.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been loaded on the eUICC with #METADATA_OP_PROF1_RPM_CONF_DE and it is in Enabled state. (PPR2 is not set in the Metadata)

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT (MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (delete, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (deleteResult, <S_TRANSACTION_ID>,

		))	#ICCID_OP_PROF1, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, profileNotInDisabledState) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA (
		MTD_GET_PROFILE_INFO (
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1}
			SW=0x9000

Test Sequence #06 Error: RPM Command DeleteProfile – disallowed by policy

The purpose of this Test Sequence is to ensure RPM Command DeleteProfile is not executed if the currently Disabled Profile can not be Deleted due to Profile Prolicy Rule.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been loaded with #METADATA_OP_PROF1_RPM_CONF_DE_PPR2 and is Disabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT (MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (delete, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (deleteResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR,

			#S_SM_DP+_OID, NO_PARAM, NO_PARAM, disallowedByPolicy) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA (
		MTD_GET_PROFILE_INFO (
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

4.2.28.2.4 TC_eUICC_ES10b.LoadRPMPackage_ListProfileInfo

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC with #METADATA_OP_PROF1_RPM_CONF_ALL_PPR1.
eUICC	The PROFILE_OPERATIONAL2 has been installed on the eUICC with #METADATA_OP_PROF2_RPM_CONF_ALL_OWNER2.
eUICC	The PROFILE_OPERATIONAL3 has been installed on the eUICC with #METADATA_OP_PROF3_RPM_CONF_ALL.
eUICC	The Nickname of the PROFILE_OPERATIONAL3 is #NICKNAME3.
eUICC	The PROFILE_OPERATIONAL1 is enabled.

Test Sequence #01 Nominal: RPM Command_ListProfileInfo_byICCID

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_FOR_LIST_PROFILE_INFO (<S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (listProfileInfoResult, <S_TRANSACTION_ID>, NO_PARAM, 0, -- OK response #NOTIF_METADATA_PROF1_DP1_RPR,

			#S_SM_DP+_OID, #PROFILE_INFO1, NO_PARAM, NO_PARAM) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

Test Sequence #02 Nominal: RPM Command_ListProfileInfo_byProfileOwnerOID

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_FOR_LIST_PROFILE_INFO (<S_TRANSACTION_ID>, NO_PARAM, <S_SM_DP+_SIGNATURE3>, #S_PROFILE_OWNER_OID, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (listProfileInfoResult, <S_TRANSACTION_ID>, NO_PARAM, 0, -- OK response { #NOTIF_METADATA_PROF1_DP1_RPR }, #S_SM_DP+_OID, { #PROFILE_INFO1, #PROFILE_INFO3 }, NO_PARAM, NO_PARAM) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

Test Sequence #03 Nominal: RPM Command ListProfileInfo - ICCID and tagList PPR present

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_FOR_LIST_PROFILE_INFO (<S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, '99'H))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (listProfileInfoResult, <S_TRANSACTION_ID>, NO_PARAM, 0, -- OK response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, { PROFILES_INFO_ICCID_TAGLIST6}, NO_PARAM, NO_PARAM) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

Test Sequence #04 Nominal: RPM Command_ListProfileInfo_- Profile Owner ID and tagList LPR Configuration present

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_FOR_LIST_PROFILE_INFO (<S_TRANSACTION_ID>, NO_PARAMS, <S_SM_DP+_SIGNATURE3>, #S_PROFILE_OWNER_OID, 'BA5A'H))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (listProfileInfoResult, <S_TRANSACTION_ID>, NO_PARAM, 0, -- OK response {

			#NOTIF_METADATA_PROF1_DP1_RPR }, #S_SM_DP+_OID, { PROFILES_INFO_RPM_TAGLIST1, PROFILES_INFO_RPM_TAGLIST2 }, NO_PARAM, NO_PARAM) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

Test Sequence #05 Nominal: RPM Command_ListProfileInfo_- Profile Owner ID and tagList with multiple Tags present

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_FOR_LIST_PROFILE_INFO (<S_TRANSACTION_ID>, NO_PARAM, <S_SM_DP+_SIGNATURE3>, #S_PROFILE_OWNER_OID, 'BA9BBC5A'H))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (listProfileInfoResult, <S_TRANSACTION_ID>, NO_PARAM, 0, -- OK response { #NOTIF_METADATA_PROF1_DP1_RPR }, #S_SM_DP+_OID, { PROFILES_INFO_RPM_TAGLIST3, PROFILES_INFO_RPM_TAGLIST4 }, NO_PARAM,

			NO_PARAM) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

Test Sequence #06 Nominal: RPM Command_ListProfileInfo_- ICCID specified and tagList with multiple Tags present

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_FOR_LIST_PROFILE_INFO (<S_TRANSACTION_ID>, #ICCID_OP_PROF2, <S_SM_DP+_SIGNATURE3>, NO_PARAM, '5ABA9F1F'H))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (listProfileInfoResult, <S_TRANSACTION_ID>, NO_PARAM, 0, -- OK response { #NOTIF_METADATA_PROF1_DP1_RPR }, #S_SM_DP+_OID, { PROFILES_INFO_RPM_TAGLIST 5 }, NO_PARAM, NO_PARAM) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

Test Sequence #07 Nominal: RPM Command_ListProfileInfo_- ICCID specified, no profile found

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_FOR_LIST_PROFILE_INFO (<S_TRANSACTION_ID>, #ICCID_OP_PROFX, <S_SM_DP+_SIGNATURE3>, NO_PARAM, '5A'H))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (listProfileInfoResult, <S_TRANSACTION_ID>, NO_PARAM, 0, -- OK response { #NOTIF_METADATA_PROF1_DP1_RPR }, #S_SM_DP+_OID, {}, NO_PARAM, NO_PARAM) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

Test Sequence #08 Nominal: RPM Command_ListProfileInfo_- Both ICCID and TagList specified, TagList not present

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_FOR_LIST_PROFILE_INFO (<S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, 'BD9F1F'H))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (listProfileInfoResult, <S_TRANSACTION_ID>,

			NO_PARAM, 0, -- OK response { #NOTIF_METADATA_PROF1_DP1_RPR }, #S_SM_DP+_OID, {}, NO_PARAM, NO_PARAM) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

Test Sequence #09 Nominal: RPM Command_ListProfileInfo_- Profile Owner ID specified, no profile found

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_FOR_LIST_PROFILE_INFO (<S_TRANSACTION_ID>, NO_PARAM, <S_SM_DP+_SIGNATURE3>, #S_PROFILE_OWNER_OIDX, '5A'H))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (listProfileInfoResult, <S_TRANSACTION_ID>, NO_PARAM, 0, -- OK response { #NOTIF_METADATA_PROF1_DP1_RPR }, #S_SM_DP+_OID, {}, NO_PARAM, NO_PARAM) for the last STORE DATA command

			• Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

Test Sequence #10 Error: RPM Command_ListProfileInfo_- ICCID specified, SM-DP+ OID not in Managing SM-DP+ list

The purpose of this Test Secuence is to ensure RPM Command ListProfileInfo returns an empty ProfileInfoListOk if the SM-DP+ that sent the RPM Command is not included in the Managing SM-DP+ List in the Profile Metadata (rpmConfiguration).

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been loaded on the eUICC with #METADATA_OP_PROF1_RPM_CONF_EN_DP_OID2.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_FOR_LIST_PROFILE_INFO (<S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, '5A'H))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (listProfileInfoResult, <S_TRANSACTION_ID>, NO_PARAM, 0, -- OK response { #NOTIF_METADATA_PROF1_DP1_RPR }, #S_SM_DP+_OID, {}, NO_PARAM, NO_PARAM) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

Test Sequence #11 Error: RPM Command_ListProfileInfo_- ICCID specified, allowed CI public key identifier does not match

The purpose of this test sequence is to ensure RPM Command ListProfileInfo returns an empty ProfileInfoListOk if Subject Key Identifier of the CI corresponding to CERT.DPauth.SIG attached to the ongoing session does not match with the Allowed CI Public Key Identifier in the Profile Metadata (rpmConfiguration).

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_OP_PROF1_RPM_CONF_ALL_CI_PKI_RAND is loaded on the eUICC.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_FOR_LIST_PROFILE_INFO (<S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, '5A'H))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (listProfileInfoResult, <S_TRANSACTION_ID>, NO_PARAM, 0, -- OK response { #NOTIF_METADATA_PROF1_DP1_RPR }, #S_SM_DP+_OID, {}, NO_PARAM, NO_PARAM) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

Test Sequence #12 Error: RPM Command_ListProfileInfo: ICCID specified, Enable on going

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA_Case3(MTD_ENABLE_PROFILE(NO_PARAM, <ISD_P_AID1>, TRUE))	No response data is returned SW=0x91XX
2	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_FOR_LIST_PROFILE_INFO (<S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, '5A'H))	SW=0x91xx without response data for all STORE DATA commands except for the last one SW=0x91xx with the response data MTD_RES_RPR_FOR_SINGLE_CMND (listProfileInfoResult, <S_TRANSACTION_ID>, NO_PARAM, 1, -- error response { #NOTIF_METADATA_PROF1_DP1_RPR }, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, profileChangeOngoing) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

3	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“UICC Reset”)
4	Repeat IC1 and IC2		

Test Sequence #13 Error: RPM Command_ListProfileInfo_- Profile Owner ID specified, SM-DP+ OID not in Managing SM-DP+ list

The purpose of this Test Sequence is to ensure RPM Command ListProfileInfo returns an empty ProfileInfoListOk if the SM-DP+ that sent the RPM Command is not included in the Managing SM-DP+ List in the Profile Metadata (rpmConfiguration).

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been loaded on the eUICC with #METADATA_OP_PROF1_RPM_CONF_ALL _DP_OID2.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_FOR_LIST_PROFILE_INFO (<S_TRANSACTION_ID>, NO_PARAM, <S_SM_DP+_SIGNATURE3>, #S_PROFILE_OWNER_OID, ‘5A’H))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (listProfileInfoResult, <S_TRANSACTION_ID>, NO_PARAM, 0, -- OK response { #NOTIF_METADATA_PROF1_DP1_RPR }, #S_SM_DP+_OID, {}, NO_PARAM, NO_PARAM) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

NOTE: #S_SM_DP+_OID2 set in the Profile Metadata is different from SM-DP+ OID in CERT.DPauth.SIG attached to the ongoing RSP session.

Test Sequence #14 Error: RPM Command_ListProfileInfo_- Profile Owner ID specified, allowed CI public key identifier does not match

The purpose of this test sequence is to ensure RPM Command ListProfileInfo returns an empty ProfileInfoListOk if Subject Key Identifier of the CI corresponding to CERT.DPauth.SIG attached to the ongoing session does not match with the Allowed CI Public Key Identifier in the Profile Metadata (rpmConfiguration).

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_OP_PROF1_RPM_CONF_ALL_CI_PKI_RAND is loaded on the eUICC.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_FOR_LIST_PROFILE_INFO (<S_TRANSACTION_ID>, NO_PARAM, <S_SM_DP+_SIGNATURE3>, #S_PROFILE_OWNER_OID, '5A'H))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (listProfileInfoResult, <S_TRANSACTION_ID>, NO_PARAM, 0, -- OK response { #NOTIF_METADATA_PROF1_DP1_RPR }, #S_SM_DP+_OID, {}, NO_PARAM, NO_PARAM) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

4.2.28.2.5 TC_eUICC_ES10b.LoadRPMPackage_UpdateMetadata

Test Sequence #01 Nominal: RPM Command UpdateMetadata – Remove PPR1

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC with #METADATA_OP_PROF1_RPM_CONF_UPDATE_MD_PPR.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (updateMetadata, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, profilePolicyRules {ppr2}, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (updateMetadataResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 0, -- OK response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, NO_PARAM) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PPR_OP_PROF1)
			resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			iccid #ICCID_OP_PROF1,
			profilePolicyRules {ppr2}
			}
			}
			SW=0x9000

Test Sequence #02 Nominal: RPM Command UpdateMetadata – RPM Configuration

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC with #METADATA_OP_PROF1_RPM_CONF_ALL_PPR1.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (updateMetadata, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, rpmConfiguration #RPM_CONFIG_OP_PROF1, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (updateMetadataResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 0, -- OK response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, NO_PARAM) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_RPM_CONFIG_OP_PROF1)
			resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			iccid #ICCID_OP_PROF1,
			rpmConfiguration {
			managingDpList {
			{ managingDP #S_SM_DP+_OID,
			rpmType {enable, disable, delete, listProfileInfo}
			tagList '99BA9BBC'H
			}
			},
			profileOwnerOid #S_PROFILE_OWNER_OID
			}
			}

			} SW=0x9000

Test Sequence #03 Nominal: RPM Command UpdateMetadata – HRI Server Address

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC with #METADATA_OP_PROF1_RPM_CONF_ALL_PPR1.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (updateMetadata, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, hriServerAddress #TEST_HRI_ADDRESS3, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (updateMetadataResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 0, -- OK response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, NO_PARAM) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(
		
		#GET_HRI_SRV_ADDRESS_OP_PROF1
		)
			resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			iccid #ICCID_OP_PROF1,
			hriServerAddress {#TEST_HRI_ADDRESS3}
			}
			}
			SW=0x9000

Test Sequence #04 Nominal: RPM Command UpdateMetadata – LPR Configuration

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC with #METADATA_OP_PROF1_RPM_CONF_ALL_PPR1.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (updateMetadata, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, lprConfiguration { pcmpAddress #TEST_PCMP_ADDRESS1 }, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (updateMetadataResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 0, -- OK response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, NO_PARAM) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_LPR_CONFIG_OP_PROF1)
			resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			iccid #ICCID_OP_PROF1,
			lprConfiguration {
			pcmpAddress
			#TEST_PCMP_ADDRESS1
			}
			}
			}
			SW=0x9000

Test Sequence #05 Nominal: RPM Command UpdateMetadata – Multiple Tags

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC with #METADATA_OP_PROF1_RPM_CONF_ALL_PPR1.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (updateMetadata, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, #MULTIPLE_TAGS_OP_PROF1, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (updateMetadataResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 0, -- OK response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, NO_PARAM) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_MULTIPLE_TAGS _OP_PROF1)
			resp ProfileInfoListResponse ::=
			profileInfoListOk : {
			{ #MULTIPLE_TAGS_OP_PROF1 }
			}
			SW=0x9000

Test Sequence #06 Error: RPM Command UpdateMetadata – ICCID unknown

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC with #METADATA_OP_PROF1_RPM_CONF_ALL_PPR1.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (updateMetadata, <S_TRANSACTION_ID>, #ICCID_OP_PROFX, <S_SM_DP+_SIGNATURE3>, lprConfiguration { pcmpAddress #TEST_PCMP_ADDRESS1 }, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (updateMetadataResult, <S_TRANSACTION_ID>, #ICCID_OP_PROFX, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, commandError) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_LPR_CONFIG_OP_PROF1)
			resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			iccid #ICCID_OP_PROF1
			}
			}
			SW=0x9000

Test Sequence #07 Error: RPM Command_UpdateMetadata – SM-DP+ OID not in Managing SM-DP+ list

The purpose of this Test Sequence is to ensure RPM Command UpdateMetadata is not executed if the SM-DP+ that sent the RPM Command is not included in the Managing SM-DP+ List in the Profile Metadata (rpmConfiguration).

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been loaded on the eUICC with #METADATA_OP_PROF1_RPM_CONF_ALL_DP_OID2.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (updateMetadata, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, lprConfiguration { pcmpAddress #TEST_PCMP_ADDRESS1 }, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (updateMetadataResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, commandError) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_LPR_CONFIG_OP_PROF1)
			resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			iccid #ICCID_OP_PROF1
			}
			}
			SW=0x9000

Test Sequence #08 Error: RPM Command_UpdateMetadata – TagList not in Managing SM-DP+ list

The purpose of this Test Sequence is to ensure RPM Command UpdateMetadata is not executed if the TagList in the RPM Command is not included in the Managing SM-DP+ List in the Profile Metadata (rpmConfiguration).

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been loaded on the eUICC with #METADATA_OP_PROF1_RPM_CONF_ALL_LPR_CONF_NOT_ALLOWED.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (updateMetadata, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, lprConfiguration { pcmpAddress #TEST_PCMP_ADDRESS1 }, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (updateMetadataResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, commandError) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_LPR_CONFIG_OP_PROF1
			resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			iccid #ICCID_OP_PROF1
			}
			}
			SW=0x9000

Test Sequence #9 Error: RPM Command UpdateMetadata – Allowed CI Public Key Identifier not matched

The purpose of this Test Sequence is to ensure RPM Command UpdateMetadata is not executed if Subject Key Identifier of the CI corresponding to CERT.DPauth.SIG attached to the ongoing session does not match with the Allowed CI Public Key Identifier in the Profile Metadata (rpmConfiguration).

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_OP_PROF1_RPM_CONF_UM_CI_PKI_RAND is loaded on the eUICC.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (updateMetadata, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, lprConfiguration { pcmpAddress #TEST_PCMP_ADDRESS1 }, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (updateMetadataResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, commandError) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_LPR_CONFIG_OP_PROF1)
			resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			iccid #ICCID_OP_PROF1
			}
			}
			SW=0x9000

Test Sequence #10 Error: RPM Command UpdateMetadata – Case4 - Remove PPR, PPR Update Control Bit set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC with #METADATA_OP_PROF1_RPM_CONF_UM_PPR_CTRL_BIT.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (updateMetadata, <S_TRANSACTION_ID>, #ICCID_OP_PROF1,	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND

		<S_SM_DP+_SIGNATURE3>, profilePolicyRules {pprUpdateControl, ppr2}, NO_PARAM))	(updateMetadataResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, pprUpdateInvalidSetting) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PPR_OP_PROF1)
			resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			iccid #ICCID_OP_PROF1,
			profilePolicyRules { pprUpdateControl,ppr1,ppr2 }
			}
			}
			SW=0x9000

Test Sequence #11 Error: RPM Command_UpdateMetadata – Case4 - Invalid RPM Configuration, zero length

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been loaded on the eUICC with #METADATA_OP_PROF1_RPM_CONF_ALL_PPR1.

Step	**Direction**	**Sequence / Description**	**Expected result**
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (updateMetadata, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, rpmConfiguration {}, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (updateMetadataResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1,

			1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, invalidRpmConfiguration) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_RPM_CONFIG_OP_PROF1)
			resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			iccid #ICCID_OP_PROF1,
			rpmConfiguration {
			managingDpList {
			{ managingDP #S_SM_DP+_OID,
			rpmType {enable, disable, delete, listProfileInfo, contactPcmp }
			tagList '99BA9BBC'H
			}
			},
			profileOwnerOid #S_PROFILE_OWNER_OID
			}
			}
			}
			SW=0x9000

Test Sequence #12 Error: RPM Command_UpdateMetadata – Case4 - Invalid RPM Comfiguration, Profile Owner OID mismatch

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been loaded on the eUICC with #METADATA_OP_PROF1_RPM_CONF_ALL_PPR1.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (updateMetadata, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>,	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (

		rpmConfiguration #RPM_CONFIG_OID2, NO_PARAM))	updateMetadataResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, invalidRpmConfiguration) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_RPM_CONFIG_OP_PROF1)
			resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			iccid #ICCID_OP_PROF1,
			rpmConfiguration {
			managingDpList {
			{ managingDP #S_SM_DP+_OID,
			rpmType {enable, disable, delete, listProfileInfo, contactPcmp }
			tagList '99BA9BBC'H
			}
			},
			profileOwnerOid #S_PROFILE_OWNER_OID
			}
			}
			}
			SW=0x9000

Test Sequence #13 Error: VOID

Test Sequence #14 Error: VOID

Test Sequence #15 Error: VOID

4.2.28.2.6 TC_eUICC_ES10b.LoadRPMPackage_ContactPCMP

Test Sequence #01 Nominal: RPM Command ContactPCMP_without DPI

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC with #METADATA_OP_PROF1_RPM_CONF_PCMP.

eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (contactPcmp, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (contactPcmpResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 0, -- OK response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, TEST_PCMP_ADDRESS1, NO_PARAM) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

Test Sequence #02 Nominal: RPM Command ContactPCMP_with dpiRpm

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC with #METADATA_OP_PROF1_RPM_CONF_PCMP.
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (contactPcmp, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, #DPI_RPM1	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (contactPcmpResult,

		))	<S_TRANSACTION_ID>, #ICCID_OP_PROF1, 0, -- OK response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, TEST_PCMP_ADDRESS1, NO_PARAM) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

Test Sequence #03 Error: RPM Command ContactPCMP – ICCID unknown

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC with #METADATA_OP_PROF1_RPM_CONF_PCMP.
eUICC	The PROFILE_OPERATIONAL1 is Enabled.
eUICC	The Operational Profile identified by the ICCID #ICCID_OP_PROFX is not loaded.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (contactPcmp, <S_TRANSACTION_ID>, #ICCID_OP_PROFX, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (contactPcmpResult, <S_TRANSACTION_ID>, #ICCID_OP_PROFX, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, commandError)

			for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

Test Sequence #04 Error: RPM Command ContactPCMP – SM-DP+ OID not in Managing SM-DP+ List

The purpose of this Test Secuence is to ensure RPM Command ContactPCMP is not executed if the SM-DP+ that sent the RPM Command is not included in the Managing SM-DP+ List in the Profile Metadata (rpmConfiguration).

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC with # METADATA_OP_PROF1_RPM_CONF_ALL_DP_OID2.
eUICC	The PROFILE_OPERATIONAL1 is Enabled.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT (MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (contactPcmp, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (contactPcmpResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, commandError) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

Test Sequence #05 Error: RPM Command ContactPCMP – Allowed CI Public Key Identifier not matched

The purpose of this test sequence is to ensure RPM Command ContactPCMP is not executed if Subject Key Identifier of the CI corresponding to CERT.DPauth.SIG attached to the ongoing session does not match with the Allowed CI Public Key Identifier in the Profile Metadata (rpmConfiguration).

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_OP_PROF1_RPM_CONF_CP_CI_PKI_RAND is loaded on the eUICC.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (contactPcmp, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (contactPcmpResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, commandError) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

Test Sequence #06 Error: RPM Command ContactPCMP – Profile is not in Enabled state

The purpose of this Test Sequence is to ensure RPM Command ContactPCMP is not executed if the target Profile is not in Enabled state.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been loaded on the eUICC with #METADATA_OP_PROF1_RPM_CONF_PCMP.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT (MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (contactPcmp, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (contactPcmpResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, profileNotEnabled) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

Test Sequence #07 Error: RPM Command ContactPCMP – PCMP Address not configured in Metadata

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been loaded on the eUICC with #METADATA_OP_PROF1_RPM_CONF_NO_PCMP_ADDR and it is in Enabled state.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT (MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (contactPcmp,	SW=0x9000 without response data for all STORE DATA commands except for the last one

		<S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM, NO_PARAM))	SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (contactPcmpResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, noLprConfiguration) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

4.2.28.2.7 TC_eUICC_ES10b.LoadRPMPackage – Multiple RPM Commands

Test Sequence #01 Nominal: RPM Commands Disable, List Profile Info and Delete, profiles of same profile owner

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_OP_PROF1_RPM_CONF_ALL (without PPR1 and PPR2 present) is loaded and Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 with #METADATA_OP_PROF2_RPM_CONF_ALL_NO_ENTERP_CONF (without PPR1 present) is loaded on the eUICC.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_MULT_CMNDS ({ MTD_REQ_RPM_CMND (disable, #ICCID_OP_PROF1, TRUE, NO_PARAM), MTD_REQ_RPM_CMND (listProfileInfo, #ICCID_OP_PROF1, TRUE, NO_PARAM), MTD_REQ_RPM_CMND (delete, #ICCID_OP_PROF1,	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x91XX with the response data MTD_RES_RPR_FOR_MULT_CMNDS ({ MTD_RES_RPR_CMND_RESULT (

		TRUE, NO_PARAM) }))	0, disableResult, #ICCID_OP_PROF1, NO_PARAM, NO_PARAM), MTD_RES_RPR_CMND_RESULT (1, listProfileInfoResult, NO_PARAM, NO_PARAM, profileChangeOngoing), MTD_RES_RPR_CMND_RESULT (0, deleteResult, #ICCID_OP_PROF1, NO_PARAM, NO_PARAM) }) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG
2	S_Device →eUICC	FETCH 'XX'	REFRESH Command (“UICC Reset”)
3	PROC_EUICC_INITIALIZATION_SEQUENCE		
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

5	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		NO_PARAM))
			response ProfileInfoListResponse ::= profileInfoListOk : {
			#PROFILE_INFO2
			}
			SW=0x9000

Test Sequence #02 Nominal: RPM Commands Enable, List Profile Info and Delete, profiles of different profile owner

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_OP_PROF1_RPM_CONF_ALL (without PPR1 present) is loaded on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 with #METADATA_OP_PROF2_RPM_CONF_ALL_OWNER2 (without PPR2 present) is loaded on the eUICC.

Step	Direction	Sequence / Description	Expected result

1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(
		MTD_LOAD_RPM_PKG_REQ_MULT_CMNDS (
		{
		MTD_REQ_RPM_CMND (
		enable, #ICCID_OP_PROF2,
		TRUE, NO_PARAM),
		MTD_REQ_RPM_CMND (
		listProfileInfo, #ICCID_OP_PROF2,
		TRUE, NO_PARAM),
		MTD_REQ_RPM_CMND (
		delete, #ICCID_OP_PROF1,
		TRUE, NO_PARAM)
		)
		)
		)
			SW=0x9000 without response data for all STORE DATA commands except for the last one
			
			SW=0x91XX with the response data
			MTD_RES_RPR_FOR_MULT_CMNDS (
			{
			MTD_RES_RPR_CMND_RESULT (
			0, enableResult, #ICCID_OP_PROF2,
			NO_PARAM, NO_PARAM),
			MTD_RES_RPR_CMND_RESULT (
			1, listProfileInfoResult, NO_PARAM,
			NO_PARAM,
			profileChangeOngoing),
			MTD_RES_RPR_CMND_RESULT (
			0, deleteResult, #ICCID_OP_PROF1,
			NO_PARAM, NO_PARAM)
			}
			)
			for the last STORE DATA command
			
			• Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_Device →eUICC	FETCH 'XX'	REFRESH Command (“UICC Reset”)
3	PROC_EUICC_INITIALIZATION_SEQUENCE		
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

5	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		NO_PARAM,
		NO_PARAM))
			response ProfileInfoListResponse ::= profileInfoListOk : {
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

Test Sequence #03 Nominal: RPM Commands UpdateMetadata, Enable, List Profile Info, Delete and ContactPCMP, profiles of different profile owner

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_OP_PROF1_RPM_CONF_ALL_PPR1 (with PPR1 present) is loaded on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 with #METADATA_OP_PROF2_RPM_CONF_ALL_OWNER2 (without PPR2 present) is loaded on the eUICC.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_MULT_CMNDS ({ MTD_REQ_RPM_CMND (updateMetadata, #ICCID_OP_PROF1, TRUE,{ profilePolicyRules { } }), MTD_REQ_RPM_CMND (enable, #ICCID_OP_PROF2, TRUE, NO_PARAM), MTD_REQ_RPM_CMND (listProfileInfo, #ICCID_OP_PROF2, TRUE, NO_PARAM), MTD_REQ_RPM_CMND (delete, #ICCID_OP_PROF1, TRUE, NO_PARAM), MTD_REQ_RPM_CMND (contactPcmp, #ICCID_OP_PROF1, TRUE, NO_PARAM)))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x91XX with the response data MTD_RES_RPR_FOR_MULT_CMNDS ({ MTD_RES_RPR_CMND_RESULT (0, updateMetadataResult, #ICCID_OP_PROF1, NO_PARAM, NO_PARAM), MTD_RES_RPR_CMND_RESULT (0, enableResult, #ICCID_OP_PROF2, NO_PARAM, NO_PARAM), MTD_RES_RPR_CMND_RESULT (1, listProfileInfoResult, NO_PARAM, NO_PARAM, profileChangeOngoing), MTD_RES_RPR_CMND_RESULT (0, deleteResult, #ICCID_OP_PROF1, NO_PARAM, NO_PARAM), MTD_RES_RPR_CMND_RESULT (1, contactPcmp Result, #ICCID_OP_PROF1, NO_PARAM, commandError) }) for the last STORE DATA command

			• Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG
2	S_Device →eUICC	FETCH 'XX'	REFRESH Command ("UICC Reset")
3	PROC_EUICC_INITIALIZATION_SEQUENCE		
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

5	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		NO_PARAM,
		NO_PARAM))
			response ProfileInfoListResponse ::= profileInfoListOk : {
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

Test Sequence #04 Nominal: RPM Commands List Profile Info, Disable, Delete and ContactPCMP - Disable command failure

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_OP_PROF1_RPM_CONF_ALL_PPR1 (with PPR1 present) is loaded and Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 with #METADATA_OP_PROF2_RPM_CONF_ALL_OWNER2 (without PPR2 present) is loaded on the eUICC.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_MULT_CMNDS ({ MTD_REQ_RPM_CMND (listProfileInfo, #ICCID_OP_PROF1, TRUE, NO_PARAM), MTD_REQ_RPM_CMND (disable, #ICCID_OP_PROF1, TRUE, NO_PARAM), MTD_REQ_RPM_CMND (delete, #ICCID_OP_PROF2, TRUE, NO_PARAM), MTD_REQ_RPM_CMND (	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_MULT_CMNDS ({ MTD_RES_RPR_CMND_RESULT (0, listProfileInfoResult, NO_PARAM, { #PROFILE_INFO1 }, NO_PARAM), MTD_RES_RPR_CMND_RESULT (1, disableResult, #ICCID_OP_PROF1,

		contactPcmp, #ICCID_OP_PROF1, TRUE, NO_PARAM))))	NO_PARAM, disallowedByPolicy), MTD_RES_RPR_CMND_RESULT (0, deleteResult, #ICCID_OP_PROF2, NO_PARAM, NO_PARAM), MTD_RES_RPR_CMND_RESULT (1, contactPcmp Result, #ICCID_OP_PROF1, NO_PARAM, noLprConfiguration) }) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		NO_PARAM,
		NO_PARAM))
			response ProfileInfoListResponse ::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #05 Nominal: RPM Commands List Profile Info, Disable, Delete and ContactPCMP - Disable command failure and ContinueOnFailure not set

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_OP_PROF1_RPM_CONF_ALL_PPR1 (with PPR1 present) is loaded and Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 with #METADATA_OP_PROF2_RPM_CONF_ALL_OWNER2 (without PPR2 present) is loaded on the eUICC.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_MULT_CMNDS ({ MTD_REQ_RPM_CMND (listProfileInfo, #ICCID_OP_PROF1, TRUE, NO_PARAM),	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_MULT_CMNDS ({

		MTD_REQ_RPM_CMND (disable, #ICCID_OP_PROF1, NO_PARAM, NO_PARAM), MTD_REQ_RPM_CMND (delete, #ICCID_OP_PROF2, TRUE, NO_PARAM), MTD_REQ_RPM_CMND (contactPcmp, #ICCID_OP_PROF1, TRUE, NO_PARAM))))	MTD_RES_RPR_CMND_RESULT (0, listProfileInfoResult, NO_PARAM, { #PROFILE_INFO1 }, NO_PARAM), MTD_RES_RPR_CMND_RESULT (1, disableResult, #ICCID_OP_PROF1, NO_PARAM, disallowedByPolicy), }) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2.	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		NO_PARAM,
		NO_PARAM))
			response ProfileInfoListResponse ::= profileInfoListOk : {
			#PROFILE_INFO1,
			#PROFILE_INFO2
			}
			SW=0x9000

Test Sequence #06 Error: VOID

Test Sequence #07 Error: RPM Commands command after ContactPCMP

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_OP_PROF1_RPM_CONF_ALL (without PPR1 present) is loaded and Enabled on the eUICC.
eUICC	The PROFILE_OPERATIONAL2 with #METADATA_OP_PROF2_RPM_CONF_ALL_OWNER2 (without PPR2 present) is loaded on the eUICC.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_MULT_CMNDS ({ MTD_REQ_RPM_CMND (	SW=0x9000 without response data for all STORE DATA commands except for the last one

		enable, #ICCID_OP_PROF2, TRUE,
		NO_PARAM),
		MTD_REQ_RPM_CMND (
		listProfileInfo, #ICCID_OP_PROF2, TRUE,
		NO_PARAM),
		MTD_REQ_RPM_CMND (
		contactPcmp, #ICCID_OP_PROF1,
		TRUE, NO_PARAM)
		MTD_REQ_RPM_CMND (
		delete, #ICCID_OP_PROF1, TRUE,
		NO_PARAM),
		)
		)
		)
			SW=0x9000 with the response data
			MTD_RES_RPR_FOR_MULT_CMNDS (
			{
			MTD_RES_RPR_CMND_RESULT (
			1, enableResult,
			#ICCID_OP_PROF2,
			NO_PARAM,
			commandError),
			
			MTD_RES_RPR_CMND_RESULT (
			0, listProfileInfoResult,
			NO_PARAM,
			{ #PROFILE_INFO2 },
			NO_PARAM),
			
			MTD_RES_RPR_CMND_RESULT (
			0, contactPcmp Result,
			#ICCID_OP_PROF1,
			NO_PARAM,
			NO_PARAM)
			
			MTD_RES_RPR_CMND_RESULT (
			2, NO_PARAM, NO_PARAM, NO_PARAM,
			commandAfterContactPcmp
			)
			}
			)
			for the last STORE DATA command
			
			• Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO (
		NO_PARAM,
		NO_PARAM))
			response ProfileInfoListResponse ::= profileInfoListOk : {
			#PROFILE_INFO1,
			#PROFILE_INFO2
			}
			SW=0x9000

4.2.28.2.8 TC_eUICC_ES10b.LoadRPMPackage_ErrorCases

[VOID]

4.2.28.2.9 TC_eUICC_ES10b.LoadRPMPackage_Enterprise_Profiles

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 is loaded on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 is Disabled.
eUICC	The communication between the S_Device and the eUICC has been initialized by using the [TERMINAL_CAPABILITY_Enterprise] and the S_LPAd has selected the ISD-R. Common Mutual Authentication procedure has been successfully executed between the eUICC and the S_SM-DP+ with the `operationType` is set rpm(1) and enterpriseCapableDevice is present in `DeviceInfo` (in `CtxParams1)`: • #GET_EUICC_INFO1, #GET_EUICC_CHALLENGE and #AUTHENTICATE_SMDP_RPM have been sent to the eUICC • the same GSMA CI has been chosen for signing and for verification

Test Sequence #1 Nominal: RPM Command EnableProfile – Reference Enterprise Profile

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC with #METADATA_OP_PROF1_RPM_CONF_EN_REF_ENTERPRISE_PROF.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (enable, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x91XX with the response data MTD_RES_RPR_FOR_SINGLE_CMND (enableResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 0, -- OK response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, NO_PARAM)

			for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG
2	S_Device → eUICC	FETCH 'XX'	REFRESH Command ("UICC Reset")
3	PROC_EUICC_INITIALIZATION_SEQUENCE_Enterprise		
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

5	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse ::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

6	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

7	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #2 Error: RPM Command EnableProfile – Other Enterprise Profile, Reference Enterprise Profile is already enabled

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL2 has been installed with #METADATA_OP_PROF2_RPM_CONF_EN_REF_ENTERPRISE_PROF before the PROFILE_OPERATIONAL1 and is Enabled on the eUICC. (PPR1 is not set)
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC with #METADATA_OP_PROF1_RPM_CONF_EN_OTHER_ENTERPRISE_PROF.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (enable, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM)	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x91XX with the response data MTD_RES_RPR_FOR_SINGLE_CMND (enableResult, <S_TRANSACTION_ID>,

		)	#ICCID_OP_PROF1, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, commandError) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG
2	PROC_EUICC_INITIALIZATION_SEQUENCE_Enterprise		
3	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

4	S_LPAd → eUICC	MTD_STORE_DATA
		(#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

5	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

6	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

Test Sequence #03 Nominal: RPM Command EnableProfile - target profile is the Reference Enterprise profile, only this profile can be enabled

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC with #METADATA_OP_PROF1_RPM_CONF_REF_PROF_ONLY_THIS_EN

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (enable, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>,	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x91XX with the response data MTD_RES_RPR_FOR_SINGLE_CMND

		NO_PARAM, NO_PARAM))	(enableResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 0, -- OK response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, NO_PARAM) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG
2	S_Device →eUICC	FETCH 'XX'	REFRESH Command (“UICC Reset”)
3	PROC_EUICC_INITIALIZATION_SEQUENCE_Enterprise		
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

5	S_LPAd →eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		#ICCID_OP_PROF1,
		NO_PARAM))
			response ProfileInfoListResponse ::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

6	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

7	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #4 Nominal: RPM Command UpdateMetadata – Enterprise Configuration - set referenceEnterpriseRule bit - profile is enabled

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC with #METADATA_OP_PROF1_RPM_CONF_EN_OTHER_ENTERPRISE_PROF and is Enabled on the eUICC.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (updateMetadata, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, enterpriseConfiguration #ENTERPRISE_CONFIG1, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (updateMetadataResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 0, -- OK response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, NO_PARAM) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd →eUICC	MTD_STORE_DATA(
		#GET_ENTERPRISE_CONFIG_OP_PROF1)
			resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			iccid #ICCID_OP_PROF1,
			enterpriseConfiguration #ENTERPRISE_CONFIG1
			}
			}
			SW=0x9000

Test Sequence #05 Nominal: RPM Command UpdateMetadata – Enterprise Configuration - set referenceEnterpriseRule bit for another enterprise profile

The purpose of this test sequence is to verify if the eUICC unset the referenceEnterpriseRule bit of the Enterprise Profile for which it is currently set when an RPM command is executed to update the target enterprise profile with the referenceEnterpriseRule bit being set.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL2 has been installed with #METADATA_OP_PROF2_RPM_CONF_EN_REF_ENTERPRISE_PROF on the eUICC .
eUICC	The PROFILE_OPERATIONAL1 has been installed and enabled on the eUICC with #METADATA_OP_PROF1_RPM_CONF_EN_OTHER_ENTERPRISE_PROF.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (updateMetadata, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, enterpriseConfiguration #ENTERPRISE_CONFIG2, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (updateMetadataResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 0, -- OK response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, NO_PARAM) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd →eUICC	MTD_STORE_DATA(
		#GET_ENTERPRISE_CONFIG_OP_PROF1)
			resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			iccid #ICCID_OP_PROF1,
			enterpriseConfiguration #ENTERPRISE_CONFIG2
			}
			}
			SW=0x9000

3	S_LPAd →eUICC	MTD_STORE_DATA(	resp ProfileInfoListResponse ::= profileInfoListOk :{

		#GET_ENTERPRISE_CONFIG_ OP_PROF2)	{ iccid #ICCID_OP_PROF2, enterpriseConfiguration { enterpriseOid #S_ENTERPRISE_OID, enterpriseName #ENTERPRISE_NAME1, enterpriseRules { enterpriseRuleBits{ priorityEnterpriseProfile, onlyEnterpriseProfilesCanBeInstalled }, numberOfNonEnterpriseProfiles 0 } } } } SW=0x9000

Test Sequence #06 Nominal: RPM Command UpdateMetadata – Enterprise Configuration - Rule 1

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed and enabled on the eUICC with #METADATA_OP_PROF1_RPM_CONF_ENTERPRISE_REF_RULE3

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (updateMetadata, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, enterpriseConfiguration #ENTERPRISE_CONFIG1, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (updateMetadataResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 0, -- OK response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID,

			NO_PARAM, NO_PARAM, NO_PARAM) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd →eUICC	MTD_STORE_DATA(
		#GET_ENTERPRISE_CONFIG_ OP_PROF1)
			resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			iccid #ICCID_OP_PROF1,
			enterpriseConfiguration #ENTERPRISE_CONFIG1
			}
			}
			SW=0x9000

Test Sequence #07 Nominal: RPM Command UpdateMetadata – Enterprise Configuration - Rule 2

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed and enabled on the eUICC with #METADATA_OP_PROF1_RPM_CONF_ENTERPRISE_REF_RULE3

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (updateMetadata, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, enterpriseConfiguration #ENTERPRISE_CONFIG2, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (updateMetadataResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 0, -- OK response #NOTIF_METADATA_PROF1_DP1_RPR,

			#S_SM_DP+_OID, NO_PARAM, NO_PARAM, NO_PARAM) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd →eUICC	MTD_STORE_DATA(
		#GET_ENTERPRISE_CONFIG_ OP_PROF1)
			resp ProfileInfoListResponse ::=
			profileInfoListOk :{
			{
			iccid #ICCID_OP_PROF1,
			enterpriseConfiguration #ENTERPRISE_CONFIG2
			}
			}
			SW=0x9000

Test Sequence #08 Nominal: RPM Command_ListProfileInfo_- ICCID specified and tagList with enterprise config tag

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL2 has been installed and enabled on the eUICC with #METADATA_OP_PROF2_RPM_CONF_ALL_ENTERP_RULES

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_FOR_LIST_PROFILE_INFO (<S_TRANSACTION_ID>, #ICCID_OP_PROF2, <S_SM_DP+_SIGNATURE3>, NO_PARAM, '5ABD9F1F'H))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (listProfileInfoResult, <S_TRANSACTION_ID>, NO_PARAM,

			0, -- OK response { #NOTIF_METADATA_PROF1_DP1_RPR }, #S_SM_DP+_OID, { PROFILES_INFO_RPM_TAGLIST 6 }, NO_PARAM, NO_PARAM) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

Test Sequence #09 Nominal: RPM Command Enable - Other Enterprise profile on a non-enterprise device

The purpose of this test sequence is to verify that an Other Enterprise profile without Enterprise Rules can be enabled on a non-enterprise device using RPM command.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been loaded on the eUICC with #METADATA_OP_PROF1_RPM_CONF_EN_OTHER_ENTERPRISE_PROF_UM_ENT_CONF in a non-Enterprise S_Device
eUICC	The PROFILE_OPERATIONAL2 that is not an enterprise profile has been installed on the eUICC with #METADATA_OP_PROF2_RPM_CONF_EN.
eUICC	The communication between the S_Device (non-Enterprise S_Device) and the eUICC has been initialized and the S_LPAd has selected the ISD-R. Common Mutual Authentication procedure has been successfully executed between the eUICC and the S_SM-DP+: • #GET_EUICC_INFO1, #GET_EUICC_CHALLENGE and #AUTHENTICATE_SMDP_RPM have been sent to the eUICC the same GSMA CI has been chosen for signing and for verification

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (	SW=0x9000 without response data for all STORE DATA commands except for the last one

		enable, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM))	SW=0x91XX with the response data MTD_RES_RPR_FOR_SINGLE_CMND (enableResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 0, -- OK response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, NO_PARAM) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG
2	S_Device →eUICC	FETCH 'XX'	REFRESH Command
3	PROC_EUICC_INITIALIZATION_SEQUENCE		
4	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

5	S_LPAd →eUICC	MTD_STORE_DATA
		(#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			#PROFILE_INFO2
			}
			SW=0x9000

Test Sequence #10 Error: RPM Command UpdateMetadata - Enterprise Configuration with Enterprise Rules on an Enterprise Profile without Enterprise Rules

The purpose of this test sequence is to verify that eUICC returns enterpriseConfigurationNotAllowed when RPM command tries to update Enterprise Configuration with Enterprise Rules on an Enterprise Profile does not already contain Enterprise Rules.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been loaded on the eUICC and enabled with #METADATA_OP_PROF1_RPM_CONF_EN_OTHER_ENTERPRISE_PROF_UM_ENT_CONF

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (updateMetadata, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, enterpriseConfiguration #ENTERPRISE_CONFIG2, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (updateMetadataResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, enterpriseConfigurationNotAllowed) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd →eUICC	MTD_STORE_DATA
		(#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #11 Error: RPM Command EnableProfile – non-Enterprise Profile, Reference Enterprise Profile is already enabled with Rule 2

The purpose of this test sequence is to verify if eUICC returns the error enterpriseRulesNotAllowed when attempting to enable a Profile via RPM command and if the eUICC contains an enabled Profile with a Reference Enterprise Rule indicating "Only Enterprise Profiles can be enabled" (Rule 2) and the target Profile is not an Enterprise Profile.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL2 has been installed with #METADATA_OP_PROF2_RPM_CONF_ALLOW_NON_ENTERPRISE_PROF_INS before the PROFILE_OPERATIONAL1 on the eUICC. (PPR1 is not set)
eUICC	The PROFILE_OPERATIONAL1 that is not an enterprise profile has been installed on the eUICC with #METADATA_OP_PROF1_RPM_CONF_EN.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (enable, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (enableResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, disallowedByEnterpriseRule) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd →eUICC	MTD_STORE_DATA
		(#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO2,
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

Test Sequence #12 Error: RPM Command EnableProfile – Other Enterprise Profile, Reference Enterprise Profile is already enabled with Rule 1

The purpose of this test sequence is to verify if eUICC returns the error enterpriseRulesNotAllowed when attempting to enable another Enterprise Profile via RPM command and if the eUICC contains an enabled Profile with a Reference Enterprise Rule indicating "Only this Profile can be enabled" (Rule 1).

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL2 has been installed with #METADATA_OP_PROF2_RPM_CONF_EN_ONLY_THIS_REF_ENTERPRISE_PROF before the PROFILE_OPERATIONAL1 on the eUICC. (PPR1 is not set)
eUICC	The PROFILE_OPERATIONAL1 another enterprise profile has been installed on the eUICC with #METADATA_OP_PROF1_RPM_CONF_EN_OTHER_ENTERPRISE_PROF.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (enable, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM))	SW=0x9000 without response data for all STORE DATA commands except for the last one SW=0x9000 with the response data MTD_RES_RPR_FOR_SINGLE_CMND (enableResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF1, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, disallowedByEnterpriseRule) for the last STORE DATA command • Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG

2	S_LPAd →eUICC	MTD_STORE_DATA
		(#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO2,
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

Test Sequence #13 Error: VOID

4.2.28.2.10 TC_eUICC_ES10b.LoadRPMPackage_CatBusy

Test Sequence #01 Error: RPM Command EnableProfile – Cat busy

The purpose of this Test Sequence is to ensure RPM Command EnableProfile is not executed if there is an active proactive session.

Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL2 has been loaded on the eUICC with #METADATA_OP_PROF2_RPM_CONF_EN.
eUICC	The PROFILE_OPERATIONAL2 is in Disabled state.
eUICC	The PROFILE_OPERATIONAL1 has been loaded with #METADATA_OP_PROF1_RPM_CONF_EN on the eUICC and is Enabled.

Step	Direction	Sequence / Description	Expected result
IC1	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
IC2	Do not send FETCH command		
1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT (MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND (enable, <S_TRANSACTION_ID>, #ICCID_OP_PROF2, <S_SM_DP+_SIGNATURE3>, NO_PARAM, NO_PARAM, NO_PARAM))	SW= 0x91XX without response data for all STORE DATA commands except for the last one SW=0x91XX with the response data MTD_RES_RPR_FOR_SINGLE_CMND (enableResult, <S_TRANSACTION_ID>, #ICCID_OP_PROF2, 1, -- error response #NOTIF_METADATA_PROF1_DP1_RPR, #S_SM_DP+_OID, NO_PARAM, NO_PARAM, catBusy) for the last STORE DATA command

			• Verify the euiccSignRPR <EUICC_SIGN_RPR> using the #PK_EUICC_SIG
2	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
3	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000

4	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			#PROFILE_INFO2
			}
			SW=0x9000

4.3 VOID

4.4 VOID

4.5 VOID

4.6 VOID

4.7 LPAe Interfaces

This section is defined as FFS.

5 Procedure - Behaviour Testing

5.1 General Overview

This section focuses on the functional behaviour of the eUICC according to the GSMA RSP Technical Specification [2].

5.2 eUICC Behaviour

5.2.1 Retry mechanism

5.2.1.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 2.6.7.1
- Section 3.1.3.2
- Section 5.7.5, 5.7.7, 5.7.14

5.2.1.2 Test Cases

5.2.1.2.1 TC_eUICC_PrepareDownload_Retry_ReuseOTKeys

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 is not loaded on the eUICC.
eUICC	The communication between the S_Device and the eUICC has been initialized and the S_LPAd has selected the ISD-R. Common Mutual Authentication procedure has been successfully executed between the eUICC and the S_SM-DP+

Test Sequence #01 Nominal: Confirmation Code retry mechanism by reusing previous One-Time key pair

The purpose of this test is to verify the Confirmation Code retry mechanism. The S_LPAd simulates that an incorrect Confirmation Code has been filled by the End User. Then, the S_LPAd sends another ES10b.PrepareDownload function with a correct Confirmation Code value. In this case, the eUICC does not have to generate a new one-time key pair and uses the previous one given by the SM-DP+.

Step	Direction	Sequence / Description	Expected result
IC1	<S_HASHED_CC> = MTD_GENERATE_HASHED_CC(#CONFIRMATION_CODE1, <S_TRANSACTION_ID>)		

1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(
		#PREP_DOWNLOAD_WITH_CC)
			#R_PREP_DOWNLOAD_WITH_CC
			SW=0x9000
			The <EUICC_SIGNATURE2> SHALL be verified with the #PK_EUICC_SIG.
			Verify that the <S_TRANSACTION_ID> present in the euiccSigned2 is the same as in #PREP_DOWNLOAD_WITH_CC.
			Verify that the <S_HASHED_CC> present in the euiccSigned2 is the same as in #PREP_DOWNLOAD_WITH_CC.
			Extract the <OTPK_EUICC_ECKA> and reuse the same value in step 4

2	Execute the Common Mutual Authentication procedure between the eUICC and the S_SM-DP+ • the same GSMA CI as for the first attempt has been chosen for signing and for verification		
3	<S_HASHED_CC> = MTD_GENERATE_HASHED_CC(#CONFIRMATION_CODE2, <S_TRANSACTION_ID>)		

4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(
		#PREP_DOWNLOAD_RETRY_CC)
			#R_PREP_DOWNLOAD_WITH_CC
			SW=0x9000
			The <EUICC_SIGNATURE2> SHALL be verified with the #PK_EUICC_SIG.

			Verify that the <S_TRANSACTION_ID> present in the euiccSigned2 is the same as in #PREP_DOWNLOAD_RETRY_CC. Verify that the <S_HASHED_CC> present in the euiccSigned2 is the same as in #PREP_DOWNLOAD_RETRY_CC. Verify that the <OTPK_EUICC_ECKA> present in the euiccSigned2 is the same as in #PREP_DOWNLOAD_RETRY_CC.

Test Sequence #02 Nominal: Retry after a CancelSession Reason "Postponed"

The purpose of this test is to verify that the eUICC can reuse the one-time key pair generated during a previous attempt. In this case, the S_LPAd simulates that the End User has postponed the download of the Profile.

Step	Direction	Sequence / Description	Expected result
IC1	<S_HASHED_CC> = MTD_GENERATE_HASHED_CC(#CONFIRMATION_CODE1, <S_TRANSACTION_ID>)		

1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(
		#PREP_DOWNLOAD_WITH_CC)
			#R_PREP_DOWNLOAD_WITH_CC
			SW=0x9000
			The <EUICC_SIGNATURE2> SHALL be verified with the #PK_EUICC_SIG.
			Verify that the <S_TRANSACTION_ID> present in the euiccSigned2 is the same as in #PREP_DOWNLOAD_WITH_CC.
			Verify that the <S_HASHED_CC> present in the euiccSigned2 is the same as in #PREP_DOWNLOAD_WITH_CC.
			Extract the <OTPK_EUICC_ECKA> and reuse the same value in step 4

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#CANCEL_SESSION_POSTPONED)
			#R_CANCEL_SESSION_POSTPONED
			SW = 0x9000

3	Execute the Common Mutual Authentication procedure between the eUICC and the S_SM-DP+ • the same GSMA CI as for the first attempt has been chosen for signing and for verification		

4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(
		#PREP_DOWNLOAD_RETRY_C)
			#R_PREP_DOWNLOAD_WITH_CC
			SW=0x9000
			The <EUICC_SIGNATURE2> SHALL be verified with the #PK_EUICC_SIG.

			Verify that the <S_TRANSACTION_ID> present in the euiccSigned2 is the same as in #PREP_DOWNLOAD_RETRY_CC. Verify that the <S_HASHED_CC> present in the euiccSigned2 is the same as in #PREP_DOWNLOAD_RETRY_CC. Verify that the <OTPK_EUICC_ECKA> present in the euiccSigned2 is the same as in #PREP_DOWNLOAD_RETRY_CC.

Test Sequence #03 Nominal: Retry after a CancelSession Reason "Timeout"

The purpose of this test is to verify that the eUICC can reuse the one-time key pair generated during a previous attempt. In this case, the S_LPAd simulates that the End User does not confirm the download of the Profile within the timeout interval.

Step	Direction	Sequence / Description	Expected result
IC1	<S_HASHED_CC> = MTD_GENERATE_HASHED_CC(#CONFIRMATION_CODE1, <S_TRANSACTION_ID>)		

1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(
		#PREP_DOWNLOAD_WITH_CC)
			#R_PREP_DOWNLOAD_WITH_CC
			SW=0x9000
			The <EUICC_SIGNATURE2> SHALL be verified with the #PK_EUICC_SIG.
			Verify that the <S_TRANSACTION_ID> present in the euiccSigned2 is the same as in #PREP_DOWNLOAD_WITH_CC.
			Verify that the <S_HASHED_CC> present in the euiccSigned2 is the same as in #PREP_DOWNLOAD_WITH_CC.
			Extract the <OTPK_EUICC_ECKA> and reuse the same value in step 4

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#CANCEL_SESSION_TIMEOUT)
			#R_CANCEL_SESSION_TIMEOUT
			SW = 0x9000

3	Execute the Common Mutual Authentication procedure between the eUICC and the S_SM-DP+ • the same GSMA CI as for the first attempt has been chosen for signing and for verification		

4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(
		#PREP_DOWNLOAD_RETRY_CC)
			#R_PREP_DOWNLOAD_WITH_CC
			SW=0x9000

			The <EUICC_SIGNATURE2> SHALL be verified with the #PK_EUICC_SIG. Verify that the <S_TRANSACTION_ID> present in the euiccSigned2 is the same as in #PREP_DOWNLOAD_RETRY_CC. Verify that the <S_HASHED_CC> present in the euiccSigned2 is the same as in #PREP_DOWNLOAD_RETRY_CC. Verify that the <OTPK_EUICC_ECKA> present in the euiccSigned2 is the same as in #PREP_DOWNLOAD_RETRY_CC.

5.2.1.2.2 TC_eUICC_PrepareDownload_Retry_NewOTKeys

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 is not loaded on the eUICC
eUICC	The communication between the S_Device and the eUICC has been initialized and the S_LPAd has selected the ISD-R. Common Mutual Authentication procedure has been successfully executed between the eUICC and the S_SM-DP+

Test Sequence #01 Nominal: Confirmation Code retry mechanism by not reusing previous One-Time key pair

The purpose of this test is to verify the Confirmation Code retry mechanism. The S_LPAd simulates that an incorrect Confirmation Code has been filled by the End User. Then, the S_LPAd sends another ES10b.PrepareDownload function with a correct Confirmation Code value. In this case, the eUICC does not support the storage of unused one-time key pair or the eUICC has discarded the previous one-time public key: we expect the eUICC to generate a new set of keys.

Step	Direction	Sequence / Description	Expected result
IC1	<S_HASHED_CC> = MTD_GENERATE_HASHED_CC(#CONFIRMATION_CODE1, <S_TRANSACTION_ID>)		

1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(
		#PREP_DOWNLOAD_WITH_CC)
			#R_PREP_DOWNLOAD_WITH_CC
			SW=0x9000
			The <EUICC_SIGNATURE2> SHALL be verified with the #PK_EUICC_SIG.
			Verify that the <S_TRANSACTION_ID> present in the euiccSigned2 is the same as in #PREP_DOWNLOAD_WITH_CC.

			Verify that the <S_HASHED_CC> present in the euiccSigned2 is the same as in #PREP_DOWNLOAD_WITH_CC. Extract the <OTPK_EUICC_ECKA> and reuse the same value in step 4
2	Execute the Common Mutual Authentication procedure between the eUICC and the S_SM-DP+ • the same GSMA CI as for the first attempt has been chosen for signing and for verification		
3	<S_HASHED_CC> = MTD_GENERATE_HASHED_CC(#CONFIRMATION_CODE2, <S_TRANSACTION_ID>)		

4	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(#PREP_DOWNLOAD_RETRY_CC)	#R_PREP_DOWNLOAD_WITH_CC
			SW=0x9000
			The <EUICC_SIGNATURE2> SHALL be verified with the #PK_EUICC_SIG.
			Verify that the <S_TRANSACTION_ID> present in the euiccSigned2 is the same as in #PREP_DOWNLOAD_RETRY_CC.
			Verify that the <S_HASHED_CC> present in the euiccSigned2 is the same as in #PREP_DOWNLOAD_RETRY_CC.
			Verify that the <OTPK_EUICC_ECKA> present in the euiccSigned2 is NOT the same as in #PREP_DOWNLOAD_RETRY_CC.

5.2.2 Forbidden PPRs

5.2.2.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 2.5.6.1
- Section 4.3
- Section 5.5.3

5.2.2.2 Test Cases

5.2.2.2.1 TC_eUICC_ForbiddenPPRs

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	There is no Profile installed in the eUICC.

Test Sequence #01 Nominal: PPR1 management and handling when Operational Profile is installed

The purpose of this test is to verify that the eUICC automatically sets PPR1 in the forbiddenProfilePolicyRules of EUICCInfo2 when an Operational Profile is installed. Any Operational Profile with PPR1 SHALL be rejected by the eUICC once an Operational Profile has been installed.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(#GET_EUICC_INFO2)	`forbiddenProfilePolicyRules` in `EUICCInfo2` does not contain ppr1
2	Install PROFILE_OPERATIONAL1		
3	S_LPAd → eUICC	MTD_STORE_DATA(#GET_EUICC_INFO2)	`forbiddenProfilePolicyRules` in `EUICCInfo2` contains ppr1
4	Execute the Common Mutual Authentication procedure between the eUICC and the S_SM-DP+		
5	Execute the Sub-procedure Profile Download and Installation – End User Confirmation between the eUICC and the S_SM-DP+ • #PREP_DOWNLOAD_NO_CC is sent to the eUICC		
6	Generate the <OTPK_S_SM_DP+_ECKA> and <OT_SK_S_SM_DP+_ECKA>		
7	<BPP> = MTD_GENERATE_BPP(#S_INIT_SC_PROF1, #CONF_ISDP_EMPTY, #METADATA_OP_PROF4, NO_PARAM, #UPP_OP_PROF4		
8	Split the <BPP> into several segments arrays named: • <BPP_SEG_INIT> • <BPP_SEG_A0> • <BPP_SEG_A1> • <BPP_SEG_A3>		
9	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_INIT>)	SW=0x9000 without response data for all STORE DATA commands
10	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A0>)	SW=0x9000 without response data for all STORE DATA commands
11	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(<BPP_SEG_A1>)	SW=0x9000 with the response data #R_PIR_PPR_NOT_ALLOWED

12	S_LPAd → eUICC	MTD_STORE_DATA(#GET_PROFILES_INFO_ALL)	response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

13	Delete PROFILE_OPERATIONAL1		
14	S_LPAd → eUICC	MTD_STORE_DATA(#GET_EUICC_INFO2)	`forbiddenProfilePolicyRules` in `EUICCInfo2` does not contain ppr1

5.2.3 eUICC's RAT

5.2.3.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 3.1.3.2
- Section 3.2.4
- Section 5.7.15, 5.7.22

5.2.3.2 Test Cases

5.2.3.2.1 TC_eUICC_GetProfilesInfo_GetRAT_RSPSession

Test Sequence #01 Nominal: GetProfilesInfo and GetRAT during RSP session

The purpose of this test is to ensure that the eUICC can be requested during a RSP session context to retrieve the list of installed Profiles and the Rules Authorization Table.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The eUICC's RAT is configured as defined in section G.2.4: • one PPAR authorizing PPR1 and PPR2 for all MNOs with End User consent required (i.e. #PPRS_ALLOWED) • no additional rules
eUICC	The PROFILE_OPERATIONAL1 is installed and Enabled on the eUICC

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

IC3	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_EUICC_INFO1)
			#R_EUICC_INFO1
			SW = 0x9000
			Extract the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> and <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION> from response data and verify if they contain at least one same GSMA CI Key ID

IC4	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_EUICC_CHALLENGE)
			#R_CHALLENGE
			SW = 0x9000
			Extract the <EUICC_CHALLENGE>

IC5	The following inputs are required for Step IC6 as described in the InitiateAuthentication function: • <S_TRANSACTION_ID> • <EUICC_CHALLENGE> • <S_SMDP_CHALLENGE>		

	• <S_SMDP_SIGNATURE1> • Set the <EUICC_CI_PK_ID_TO_BE_USED> to the CI Key ID in highest priority from the <EUICC_CI_PK_ID_LIST_FOR_SIGNING> • Choose the #CERT_S_SM_DPauth_SIG leading to the same Root CI certificate		

IC6	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(
		MTD_AUTHENTICATE_SMDP(
		#TEST_DP_ADDRESS1,
		<S_SMDP_CHALLENGE>,
		#CTX_PARAMS1,
		<S_SMDP_SIGNATURE1>,
		#CERT_S_SM_DPauth_SIG,
		NO_PARAM,
		#CRL_LIST,
		FALSE
		)
		)
			MTD_CHECK_AUTH_SERVER_RESP(
			#TEST_DP_ADDRESS1,
			<S_SMDP_CHALLENGE>,
			#CTX_PARAMS1)
			SW = 0x9000

1	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_RAT)
			#R_DEFAULT_RAT with exact same structure and order
			SW = 0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::=
			profileInfoListOk: {
			#PROFILE_INFO1
			}
			SW = 0x9000

3	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(
		#PREP_DOWNLOAD_NO_CC)
			#R_PREP_DOWNLOAD_NO_CC
			SW=0x9000

5.2.4 eUICC File Structure

5.2.4.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 3.4.1, 3.4.2, 3.4.3

5.2.4.2 Test Cases

5.2.4.2.1 TC_eUICC_Default_FileSystem

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	There is no Profile installed in the eUICC.

Test Sequence #01 Nominal: Default file system available

The purpose of this test is to verify that if there is no Profile on the eUICC, the eUICC still ensures a file system to the Device.

Step	Direction	Sequence / Description	Expected result
1	S_Device → eUICC	RESET	ATR present
2	S_Device → eUICC	[SELECT_MF]	FCP Template present with tag 0xA5 (Proprietary Information) containing 0x87 01 01 (Supported system commands = TERMINAL CAPABILITY) SW=0x9000
3	S_Device → eUICC	[TERMINAL_CAPABILITY_LPAd]	SW=0x9000
4	S_Device → eUICC	[TERMINAL_PROFILE]	Toolkit initialization THEN SW=0x9000

5.2.5 eUICC Delete Profile Process

5.2.5.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 2.4.4
- Section 3.0.1, 3.1.4
- Section 5.7.8

5.2.5.2 Test Cases

5.2.5.2.1 TC_eUICC_DeleteProfile_ISDP_And_Components

Test Sequence #01 Nominal: ISD-P and Profile Components Deletion

The purpose of this test is to verify that when a Profile is deleted, the eUICC removes the ISD-P and all Profile Components related to it. In order to do so, we are checking the eUICC Non-Volatile Memory variation.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	There is no Profile installed on the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	S_LPAd → eUICC	MTD_STORE_DATA(#GET_EUICC_INFO2)	Retrieve free non-volatile memory value (tag 0x82) from <EXT_CARD_RESOURCE> in `EUICCInfo2` as <FREE_MEMORY_NO_PROFILE>

IC4	Install PROFILE_OPERATIONAL1		
IC5	Remove all Install Notifications from eUICC		
1	S_LPAd → eUICC	MTD_STORE_DATA(#GET_EUICC_INFO2)	Retrieve free non-volatile memory value (tag 0x82) from <EXT_CARD_RESOURCE> in `EUICCInfo2` as <FREE_MEM_OP_PROF1_INSTALLED> Verify that <FREE_MEM_OP_PROF1_INSTALLED> is lower than <FREE_MEMORY_NO_PROFILE>
2	Delete PROFILE_OPERATIONAL1		
3	Remove the Delete Notification from eUICC		
4	S_LPAd → eUICC	MTD_STORE_DATA(#GET_EUICC_INFO2)	Retrieve free non-volatile memory value (tag 0x82) from <EXT_CARD_RESOURCE> in `EUICCInfo2` as <FREE_MEM_OP_PROF1_DELETED> Verify that <FREE_MEM_OP_PROF1_DELETED> is higher than <FREE_MEM_OP_PROF1_INSTALLED>

5.2.6 eUICC Enable Profile Process

5.2.6.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 3.5
- Section 5.5.5
- Section 5.7.16

5.2.6.2 Test Cases

5.2.6.2.1 TC_eUICC_EnableProfile_Twice_Notifications

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 is installed and Enabled on the eUICC.
eUICC	No Notification is stored in the eUICC's Pending Notifications List.

Test Sequence #01 Nominal: Notifications generation

The purpose of this test is to verify that when an Enable Profile operation is performed and the current Enabled Profile is implicitly Disabled, both Notifications are generated. The

eUICC automatically increments its sequence number each time a Notification is generated across all Profiles.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Install PROFILE_OPERATIONAL2 The default Profile downloading procedure defined in section 2.2.3.1 SHALL be used with the following exceptions: • #CERT_S_SM_DP2auth_SIG SHALL be set in `MTD_AUTHENTICATE_SMDP` rather than #CERT_S_SM_DPauth_SIG • #TEST_DP_ADDRESS2 SHALL be set in `MTD_AUTHENTICATE_SMDP` rather than #TEST_DP_ADDRESS1 • #CERT_S_SM_DP2pb_SIG SHALL be set in `#PREP_DOWNLOAD_NO_CC` rather than #CERT_S_SM_DPpb_SIG		
1	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	
2	Remove the ProfileInstallationResult and OtherSignedNotification for Install		
3	Enable PROFILE_OPERATIONAL2		
4	PROC_EUICC_INITIALIZATION_SEQUENCE		
5	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

6	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_DI1_EN2
			SW = 0x9000
			
			Verify that
			
			<NOTIF_SEQ_NO_IN2> is lower than <NOTIF_SEQ_NO_DI1>.
			
			Verify that
			
			<NOTIF_SEQ_NO_DI1> and <NOTIF_SEQ_NO_EN2> follow this order in an incremental sequence

NOTE: In order to compare the sequence numbers, the test tool can retrieve the <NOTIF_SEQ_NO_IN2_PIR> value through the PIR returned at the end of the step IC3.			

5.2.7 eUICC Disable Profile Process

5.2.7.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 2.4.5

5.2.7.2 Test Cases

5.2.7.2.1 TC_eUICC_DisableProfile_ApplicationManagement

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	PROFILE_OPERATIONAL1 is installed and Enabled.

Test Sequence #01 Nominal: Application Selection/Deletion not available on Disabled Profile

The purpose of this test is to verify that when a Profile is Disabled, the eUICC does not allow the selection or deletion of any application within the Profile.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	S_Device → eUICC	[SELECT_USIM]	FCP Template present SW=0x9000
IC3	S_Device → eUICC	MTD_SELECT(#SSD_AID)	SSD is selected SW=0x9000
IC4	Disable PROFILE_OPERATIONAL1		
1	S_Device → eUICC	[SELECT_USIM]	USIM is not found SW=0x6A82
2	S_Device → eUICC	MTD_SELECT(#SSD_AID)	SSD is not found SW=0x6A82
3	PROC_EUICC_INITIALIZATION_SEQUENCE		
4	S_Device → eUICC	MTD_SEND_SMS_PP([DELETE_SSD])	SW=0x91XX or SW=0x9000 (i.e. envelope rejected, see Note) or any error SW (i.e. envelope rejected, see Note)
5	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x06 (Unidentified security error) or 0x09 (TAR unknown)
6	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000
7	Enable PROFILE_OPERATIONAL1		
8	S_Device → eUICC	MTD_SELECT(#SSD_AID)	SSD is selected SW=0x9000
NOTE: Depending on the implementation, the eUICC MAY decide to not send back a POR (e.g. SW=0x9000 on the ENVELOPE command). Therefore, the steps 5 and 6 SHALL only be executed in case SW=0x91XX.			

5.2.8 eUICC Notifications

5.2.8.1 Conformance Requirements

References

GSMA RSP Technical Specification [2]:

- Section 5.7.16, 5.7.17, 5.7.18

5.2.8.2 Test Cases

5.2.8.2.1 TC_eUICC_Enable_Disable_Delete_Notifications

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 with #METADATA_EN_DI_DE_NOTIFS is loaded on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 is Disabled.
eUICC	No Notification is stored in the eUICC's Pending Notifications List.

Test Sequence #01 Nominal: Multiple Enable, Disable and Delete Notifications

The purpose of this test is to verify that when a Local Profile Management Operation (i.e. Enable, Disable and Delete Profile) is performed, all Notifications configured in the `notificationConfigurationInfo` are generated by the eUICC.

NOTE: In this sequence, the maximum number of Notifications simultaneously tested has been set as to two as there is not minimum defined in SGP.21 or SGP.22 [2] for the number of Notifications that can be stored by the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
IC3	Enable PROFILE_OPERATIONAL1		
IC4	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC5	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_EN1_EN1 SW = 0x9000
2	Remove all the pending notifications		
3	Disable PROFILE_OPERATIONAL1		
4	PROC_EUICC_INITIALIZATION_SEQUENCE		
5	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
6	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_DI1_DI1 SW = 0x9000

7	Remove all the pending notifications		
8	Delete PROFILE_OPERATIONAL1		

9	S_LPAd → eUICC	MTD_STORE_DATA(#LIST_NOTIF_ALL)	#R_LIST_NOTIF_DE1_DE1
			SW = 0x9000

5.3 VOID

5.4 VOID

6 VOID

7 VOID

Annex A Constants

A.1 Generic Constants

Name	Content
ACTIVATION_CODE_1	1$#TEST_DP_ADDRESS1$#MATCHING_ID_1
ACTIVATION_CODE_2	1$#TEST_DP_ADDRESS1$#MATCHING_ID_2$#S_SM_DP+_OID
ADDITIONAL_SMDP_DATA_EXCEEDED_MAX	0x01 02 03…76 77 78 -- additional data objects defined by the S_SM-DP+ depending on the length of the SM-DP+ OID, to ensure that total length of dpProprietaryData is 129 bytes
ADDITIONAL_SMDP_DATA_MAX_LENGTH	0x01 02 03…75 76 77 -- additional data objects defined by the S_SM-DP+ depending on the length of the SM-DP+ OID, to ensure that total length of dpProprietaryData is 128 bytes
CONFIRMATION_CODE1	0102030405
CONFIRMATION_CODE2	ABCDEFGHIJ

CTX_PARAMS1
(CtxParams1)
	ctxParamsForCommonAuthentication : {
	#S_DEVICE_INFO,
	operationType { profileDownload },
	matchingIdSource none : NULL
	}

CTX_PARAMS1_DEVICE_INFO_NAI	ctxParamsForCommonAuthentication : {
	matchingId #MATCHING_ID_EMPTY,
	deviceInfo #S_DEVICE_INFO_NAI,
	operationType { profileDownload }
	}

CTX_PARAMS1_EVENT_ID
(CtxParams1)
	ctxParamsForCommonAuthentication : {
	matchingId #EVENT_ID_1,
	#S_DEVICE_INFO,
	matchingIdSource smdsOid : { #S_SM_DS_OID }
	}

CTX_PARAMS1_EVENT_ID_IMEI
(CtxParams1)
	ctxParamsForCommonAuthentication : {
	matchingId #EVENT_ID_1,
	#S_DEVICE_INFO_IMEI,
	matchingIdSource smdsOid : { #S_SM_DS_OID }
	}

CTX_PARAMS1_IMEI
(CtxParams1)
	ctxParamsForCommonAuthentication : {
	#S_DEVICE_INFO_IMEI,
	matchingIdSource none : NULL
	}

CTX_PARAMS1_MATCH_ID
(CtxParams1)
	ctxParamsForCommonAuthentication : {
	matchingId #MATCHING_ID_1,
	#S_DEVICE_INFO,
	matchingIdSource activationCode : NULL
	}

CTX_PARAMS1_RPM
(CtxParams1)
	ctxParamsForCommonAuthentication : {
	#S_DEVICE_INFO,
	operationType { rpm },
	matchingIdSource none : NULL
	}

CTX_PARAMS1_RPM_ICCID1	ctxParamsForCommonAuthentication : {
	#S_DEVICE_INFO,
	operationType {rpm},
	iccid #ICCID_OP_PROF1
	matchingIdSource none : NULL
	}

DPI_RPM1	/dpi_1
EF_UST1	0x0A 2E 14 8C E7 32 04 00 00 00 00 00 00 NOTE: Service n°17 (GID1) and n°18 (GID2) not available.
EF_UST2	0x0A 2E 17 8C E7 32 04 00 00 00 00 00 00 NOTE: Service n°17 (GID1) and n°18 (GID2) available.
EID1	0x89 04 90 32 12 34 51 23 45 12 34 56 78 90 12 35
ENTERPRISE_NAME1	Enterprise Name 1
ESIM_PORT_INIT	eSIM Port 0 to be used for subsequent eSIM initialization APDUs after MANAGE LSI (Configure LSI) command
ESIM_PORT_INIT_1	eSIM Port 1 to be used for subsequent eSIM initialization APDUs after MANAGE LSI (Configure LSI) command

EUICC_SIGNED1	{
	transactionId <S_TRANSACTION_ID>,
	serverAddress #TEST_DP_ADDRESS1,
	serverChallenge <S_SMDP_CHALLENGE>,
	euiccInfo2 #R_EUICC_INFO2, -- check only that the field is present and has a valid TLV asn.1 structure
	ctxParams1 #CTX_PARAMS1
	}

EVENT_ID_1	07399-BGH7E-T8779
GID1	0x47 53 4D 41
GID2	0x52 53 50 FF
HOST_ID	0x47 53 4D 41 20 53 4D 2D 58 58 NOTE: 'GSMA SM-XX' in ASCII.
ICCID_OP_PROF1	0x98 92 09 01 21 43 65 87 09 F5
ICCID_OP_PROF2	0x98 92 09 01 32 54 76 98 10 F9
ICCID_OP_PROF3	0x98 92 09 01 43 65 87 09 21 F5
ICCID_OP_PROF4	0x98 92 09 01 54 76 98 10 32 F9
ICCID_OP_PROF9	0x98 92 09 01 21 43 65 87 76 F5
ICCID_OP_PROF10	0x98 10 99 09 00 21 43 65 87 79
ICCID_OP_PROFX	0x98 92 09 01 43 65 87 09 FF FF
ICCID_UNKNOWN	0x98 92 01 0A 21 43 65 87 09 F8
ICON_JPG	ICON_JPG.jpg as defined in Annex H
ICON_OP_PROF1	profile_O1.png as defined in Annex H
ICON_OP_PROF2	profile_O2.png as defined in Annex H
ICON_OP_PROF3	profile_O3.png as defined in Annex H
ICON_OP_PROF4	profile_O4.png as defined in Annex H
ICON_OP_PROF5	profile_O5.png as defined in Annex H

IMSI_OP_PROF1	0x08 29 99 18 11 32 54 76 98
IMSI_OP_PROF2	0x08 29 99 28 11 32 54 76 97
IMSI_OP_PROF3	0x08 29 99 28 11 32 54 76 96
IMSI_OP_PROF4	0x08 29 99 48 43 65 87 09 21
IMSI_OP_PROF9	0x08 29 99 98 43 65 87 09 21
INVALID_KEY_TYPE	0x80
INVALID_REMOTE_OP_ID	8
ISD_R_AID	0xA0 00 00 05 59 10 10 FF FF FF FF 89 00 00 01 00
KEY_LENGTH	0x10
KEY_TYPE	0x88
MATCHING_ID_1	04386-AGYFT-A74Y8-3F815
MATCHING_ID_2	04386-AGYFT-A74Y8-3F816
MATCHING_ID_EMPTY	<zero-length string>
MCC_MNC_WILDCARD	0x92 F9 EE
MCC_MNC1	0x92 F9 18
MCC_MNC2	0x92 F9 28
MCC_MNC3	0x92 F9 28
MCC_MNC4	0x92 F9 48
MCC_MNC9	0x92 F9 98
MEP_REFRESH_FLAG	TRUE or FALSE depending on the TC capability.
MNO_SCP80_AUTH_KEY	0x11 22 33 44 55 66 77 88 99 AA BB CC DD EE FF 10
MNO_SCP80_DATA_ENC_KEY	0x99 AA BB CC DD EE FF 10 11 22 33 44 55 66 77 88
MNO_SCP80_ENC_KEY	0x66 77 88 99 AA BB CC DD 11 22 33 44 55 EE FF 10
NAME_OP_PROF1	Operational Profile Name 1
NAME_OP_PROF2	Operational Profile Name 2
NAME_OP_PROF3	Operational Profile Name 3
NAME_OP_PROF4	Operational Profile Name 4
NAME_OP_PROF5	Operational Profile Name 5
NAME_OP_PROF9	Operational Profile Name 9
NICKNAME1	Nickname 1
NICKNAME2	Nickname 2
NICKNAME3	Nickname 3
OWNER_OP_PROF1	{ mccMnc #MCC_MNC1 }
OWNER_OP_PROF2	{ mccMnc #MCC_MNC2 }
PO1_PIN1	0x32 34 36 38 FF FF FF FF
PPK_ENC_INV_SIZE	0x01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 0D 0E 0F 10 0D 0E 0F 10
PPK_INIT_MAC_INV_SIZE	0x05 0A 04 0B 03 0C 02 0D 01 0E 00 0F 09 01 08 02 09 01 08 02 09 01 08 02
PPK_MAC_INV_SIZE	0x01 0E 00 0F 09 01 08 02 05 0A 04 0B 03 0C 02 0D 03 0C 02 0D 03 0C 02 0D

REMOTE_OP_ID_INSTALL	1
RSP_VERSION_LOWEST_H	This field is set to the lowest RSP version as defined in SGP.22 Annex O, encoded as the value part of an ASN.1 VersionType (i.e. `0x02 01 00`)

S_DEVICE_INFO	`deviceInfo {`
	`  tac #S_TAC,`
	`  deviceCapabilities {`
	`  gsmSupportedRelease '050000'H,`
	`  utranSupportedRelease '080000'H,`
	`  cdma2000onexSupportedRelease '010000'H,`
	`  cdma2000hrpdSupportedRelease '010000'H,`
	`  cdma2000ehrpdSupportedRelease '020000'H,`
	`  eutranSupportedRelease '020000'H,`
	`  contactlessSupportedRelease '090000'H,`
	`  nrEpcSupportedRelease '170000'H,`
	`  nr5gcSupportedRelease '170000'H,`
	`  eutran5gcSupportedRelease '170000'H,`
	`  lpaSvn '030000'H,`
	`  catSupportedClasses`
	`'11111111111111111111111111111'B,`
	`  euiccFormFactorType #IUT_EUICC_FF_TYPE`
	`  },`
	`lpaRspCapability '11'B`
	`}`

S_DEVICE_INFO_IMEI	`deviceInfo {`
	`  tac #S_TAC,`
	`  deviceCapabilities {`
	`  gsmSupportedRelease '050000'H,`
	`  utranSupportedRelease '080000'H,`
	`  cdma2000onexSupportedRelease '01000'H,`
	`  eutranSupportedRelease '020000'H,`
	`  lpaSvn '030000'H,`
	`  catSupportedClasses`
	`'11111111111111111111111111111'B,`
	`  euiccFormFactorType #IUT_EUICC_FF_TYPE`
	`  },`
	`  imei #S_IMEI,`
	`  lpaRspCapability '11'B`
	`}`

S_DEVICE_INFO_NAI	deviceInfo DeviceInfo {
	  tac #S_TAC,
	  deviceCapabilities {
	  gsmSupportedRelease '050000'H,
	  utranSupportedRelease '080000'H,
	  cdma2000onexSupportedRelease '010000'H,
	  cdma2000hrpdSupportedRelease '010000'H,
	  cdma2000ehrpdSupportedRelease
	  '020000'H,
	  eutranEpcSupportedRelease '020000'H,
	  contactlessSupportedRelease '090000'H,
	  rspCrlSupportedVersion, #RSP_SVN_H
	  nrEpcSupportedRelease '0F0000'H,
	  nr5gcSupportedRelease '100000'H,
	  eutran5gcSupportedRelease '100000'H,
	  -- No lpaSvn field
	  -- No catSupportedClasses field
	  -- No euiccFormFactorType field
	  deviceAdditionalFeatureSupport {
	  naiSupport '100000'H
	  }
	  }
	}
	}

S_ENTERPRISE_OID	2.888.99
S_ENTERPRISE_OID2	2.888.999
S_EUICC_CHALLENGE	0x01 02 03 04 05 06 07 08 01 02 03 04 05 06 07 08
S_IMEI	0x00 00 00 00 11 11 11 11
S_PROFILE_OWNER_OID	2.888.9
S_PROFILE_OWNER_OID2	2.888.99
S_PROFILE_OWNER_OIDX	2.888.999
S_SERVER_SVN_H	0x03 00 00
S_SM_DP+_OID	2.999.10
S_SM_DP+_OID2	2.999.12
S_SM_DS_OID	2.999.15
S_TAC	0x00 00 00 00

SIMA_RESULT_OK	simaresp EUICCResponse ::= {
	  peStatus {
	  {status ok}
	  }
	}

SP_NAME1	SP Name 1
SP_NAME2	SP Name 2
SP_NAME3	SP Name 3
SP_NAME4	SP Name 4
SP_NAME9	SP Name 9
SSD_AID	0xA0 00 00 05 59 10 10 01 02 73 64 56 61 6C 75 65
TARGET_ESIM_PORT	Target eSIM Port to be used for ES10c.EnableProfileRequest and ES10c.DisableProfileRequest

TEST_DP_ADDRESS1	testsmdpplus1.example.com
TEST_DP_ADDRESS2	testsmdpplus2.example.com
TEST_DP_ADDRESS3	testsmdpplus3.example.com
TEST_DP_ADDRESS4	testsmdpplus4.example.com
TEST_HRI_ADDRESS1	testhriserver1.example.com
TEST_HRI_ADDRESS3	testhriserver3.example.com
TEST_PCMP_ADDRESS1	testpcmp1.example.com
TEST_PCMP_ADDRESS3	testpcmp3.example.com
TEST_ROOT_DS_ADDRESS	testrootsmds.example.com
UNKNOWN_BPP_SEGMENT	0xC9 05 01 02 03 04 05
UPP_OP_PROF1	The Unprotected Profile Package related to the PROFILE_OPERATIONAL1 (see Annex E).
UPP_OP_PROF2	The Unprotected Profile Package related to the PROFILE_OPERATIONAL2 (see Annex E).
UPP_OP_PROF3	The Unprotected Profile Package related to the PROFILE_OPERATIONAL3 (see Annex E).
UPP_OP_PROF4	The Unprotected Profile Package related to the PROFILE_OPERATIONAL4 (see Annex E).
UPP_OP_PROF9	The Unprotected Profile Package related to the PROFILE_OPERATIONAL9 (see Annex E).
UPP_OP_PROF10	The Unprotected Profile Package related to the PROFILE_OPERATIONAL10 (see Annex E).
USIM_AID	0xA0 00 00 00 87 10 02 FF 33 FF 01 89 00 00 01 00

VENDOR_SPECIFIC_EXTENSION1	VendorSpecificExtension : {
	{
	vendorOid 2.999.16,
	vendorSpecificData ‘C1020304’
	}
	}

VENDOR_SPECIFIC_EXTENSION2	VendorSpecificExtension : {
	{
	vendorOid 2.999.17,
	vendorSpecificData ‘02020202’
	}
	}

A.2 Test Certificates and Test Keys

All ECC certificates and keys described below are based on descriptions and security requirements (such as algorithms to be used) from SGP.22[2].

NOTE: SGP.26 [25] contains test keys, valid test certificates and instructions for how to generate invalid certificates. All test keys and test certificates used in the present document are bundled with SGP.26 [25].

Name	Description
CERT_CI_SIG	Certificate of the CI for its Public ECDSA Key.

Name	Description
CERT_CI_SubCA_SIG	The intermediate CI SubCA Certificate through which #CERT_EUM_SIG, #CERT_S_SM_DP_SubCA_SIG or #CERT_S_SM_DPauth_SIG and the #CERT_S_SM_DPpb_SIG are chained to the #CERT_CI_SIG in Variants B and C. This certificate contains the same Extension for subjectAltName value as in #CERT_CI_SIG of that same GSMA CI.
CERT_EUICC_SIG	Certificate of the eUICC for its Public ECDSA key CERT.EUICC.SIG in the X.509 format signed by the EUM with SK.EUM.SIG or SK.EUMSubCA.SIG.
CERT_EUM_SIG	Certificate of the EUM for its Public ECDSA key CERT.EUM.SIG in the X.509 format signed by the requested CI with SK.CI.SIG or SK.CISubCA.SIG.
CERT_EUM_SubCA_SIG	The intermediate SubCA Certificate through which #CERT_EUICC_SIG is chained to the #CERT_EUM_SIG in Variants A and C. This certificate contains the same Extension for subjectAltName value as in #CERT_EUM_SIG.
CERT_S_SM_DP_SubCA_SIG	The intermediate SubCA Certificate through which #CERT_S_SM_DPauth_SIG and the #CERT_S_SM_DPpb_SIG are chained to the #CERT_CI_SIG in Variant A or to the #CERT_CI_SubCA_SIG in Variant C. This certificate contains the Extension for subjectAltName as OID #S_SM_DP+_OID.
CERT_S_SM_DP_SubCAList_SIG (CertificateChain)	#CERT_S_SM_DP_SubCA_SIG, #CERT_CI_SubCA_SIG
CERT_S_SM_DP2auth_SIG	Certificate of the S_SM-DP+ for its Public ECDSA key used for SM-DP+ authentication. This certificate contains the OID #S_SM_DP+_OID2.
CERT_S_SM_DP2pb_SIG	Certificate of the S_SM-DP+ for its Public ECDSA key used for Profile Package Binding. This certificate contains the OID #S_SM_DP+_OID2.
CERT_S_SM_DPauth_SIG	Certificate of the S_SM-DP+ for its Public ECDSA key used for SM-DP+ authentication. This certificate contains the OID #S_SM_DP+_OID.
CERT_S_SM_DPauth_INV_CURVE	Certificate of the S_SM-DP+ for its Public ECDSA key used for Authentication. This certificate contains the OID #S_SM_DP+_OID and a public key based on a curve different from the allowed curves defined in SGP.22[2], section 2.6. •
CERT_S_SM_DPauth_INV_SIGN	Invalid certificate of the S_SM-DP+ for its Public ECDSA key used for authentication. This certificate contains the OID #S_SM_DP+_OID and contains an invalid signature (i.e. not generated with the #SK_CI_SIG but with the same tag and length as a valid signature).
CERT_S_SM_DPpb_SIG	Certificate of the S_SM-DP+ for its Public ECDSA key used for Profile Package Binding. This certificate contains the OID #S_SM_DP+_OID.
CERT_S_SM_DPpb_INV_CURVE	Certificate of the S_SM-DP+ for its Public ECDSA key used for Profile Package Binding. This certificate contains the OID

Name	Description
	#S_SM_DP+_OID and a public key based on a curve different from the allowed curves defined in SGP.22[2], section 2.6. •
CERT_S_SM_DPpb_INV_SIGN	Invalid certificate of the S_SM-DP+ for its Public ECDSA key used for Profile Package Binding. This certificate contains the OID #S_SM_DP+_OID and contains an invalid signature (i.e. not generated with the #SK_CI_SIG but with the same tag and length as a valid signature).
CERT_S_SM_DSauth_SIG	Certificate of the S_SM-DS for its Public ECDSA key used for SM-DS authentication. This certificate contains the OID #S_SM_DS_OID.
CERT_S_SM_DSauth_INV_CURVE	Certificate of the S_SM-DS for its Public ECDSA key used for Authentication. This certificate contains the OID #S_SM_DS_OID and a public key based on a curve different from the allowed curves defined in SGP.22[2], section 2.6. •
CERT_S_SM_DSauth_INV_SIGN	Invalid certificate of the S_SM-DS for its Public ECDSA key used for SM-DS authentication. This certificate contains an invalid signature, (i.e. not generated with the #SK_CI_SIG but with the same tag and length as a valid signature).
CRL_LIST	The list of CRLs needed to verify the revocation status of each Certificate that contains a `cRLDistributionPoints` extension in the returned Certificate chain.
PK_CI_SIG	Public Key of the CI, contained within #CERT_CI_SIG.
PK_EUICC_SIG	Public Key of the eUICC, contained within #CERT_EUICC_SIG.
SK_CI_SIG	Private Key of the CI.
SK_EUICC_SIG	Private key of the eUICC for creating signatures.
SK_S_SM_DPauth_SIG	Private Key of the S_SM-DP+ for creating signatures for SM-DP+ authentication.
SK_S_SM_DPpb_SIG	Private key of the S_SM-DP+ used to provide signatures for Profile binding.
SK_S_SM_DSauth_SIG	Private Key of the S_SM-DS for creating signatures for SM-DS authentication.

Annex B Dynamic Content

Variable	Description
ANY_PORT_VALUE	Any valid positive INTEGER value
ANY_SW_IN_ERROR	Any Status Word in error (different from 0x9000).
ATR	ATR received from eUICC after RESET
BPP	Content of a Bound Profile Package to download within the eUICC.
BPP_SEG_A0	Bound Profile Package TLV segment containing the tag and length fields of the firstSequenceOf87 TLV plus the first 0x87 TLV containing the ConfigureISDP command.
BPP_SEG_A1	Bound Profile Package following TLV segment array, as defined in SGP.22 [2] – section 2.5.5: • array first element containing the Tag and length fields of the sequenceOf88 TLV • array following elements containing each of the '88' TLVs containing the StoreMetadata command
BPP_SEG_A2	Bound Profile Package TLV segment containing the Tag and length fields of the secondSequenceOf87 TLV plus the first '87' TLV, containing the ReplaceSessionKeys command.
BPP_SEG_A3	Bound Profile Package following TLV segment array, as defined in SGP.22 [2] – section 2.5.5: • array first element containing the tag and length fields of the sequenceOf86 TLV • array following elements containing each of the '86' TLVs containing the Protected Profile Package (PPP)
BPP_SEG_INIT	Bound Profile Package TLV segment containing the tag and length fields of the BoundProfilePackage TLV plus the initialiseSecureChannelRequest command.
C_APDUS_SCRIPT	List of Command APDUs formatted as an expanded structure with definite length coding as defined in ETSI TS 102 226 [14].
CC	SCP80 cryptographic checksum as defined in ETSI TS 102 225 [13] (8 bytes long).
CHANNEL_NUMBER	The logical channel number newly opened in the eUICC. If no logical channel is opened, the value is set to 0x00 (i.e. Basic Channel). There is one value per LSI.
CI_PKI_RANDOM	Random Subject Key Identifier of the PK CI different from all the PK CI Identifiers defined in SGP.26 [25]. This random value has the same length as the ones defined in SGP.26 [25].
EUICC_CHALLENGE	Random eUICC challenge, coded as asn.1 OCTET STRING, 16 bytes.
EUICC_CI_PK_ID_LIST_FOR_SIGNING	List of CI Public Key Identifiers supported on the eUICC for signature creation that can be verified by a certificate chain Variant O, coded as ASN.1 sequence of SubjectKeyIdentifier. The CI Public Key Identifiers are from the list of possible identifiers as defined in SGP.26 [25]. This list can be empty.
EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3	List of CI Public Key Identifiers supported on the eUICC for signature creation that can be verified by a certificate chain

Variable	Description
	Variant A, B or C, coded as ASN.1 sequence of SubjectKeyIdentifier. The CI Public Key Identifiers are from the list of possible identifiers as defined in SGP.26 [25].
EUICC_CI_PK_ID_LIST_FOR_VERIFICATION	List of CI Public Key Identifiers supported on the eUICC for signature verification, coded as ASN.1 sequence of SubjectKeyIdentifier. The CI Public Key Identifiers are from the list of possible identifiers as defined in SGP.26 [25].
EUICC_CI_PK_ID_TO_BE_USED	CI Public Key Identifier to be used by the eUICC for signature, coded as ASN.1 sequence of SubjectKeyIdentifier, selected from <EUICC_CI_PK_ID_LIST_FOR_SIGNING>.
EUICC_CI_PK_ID_TO_BE_USED_V3	CI Public Key Identifier to be used by the eUICC for signature, coded as ASN.1 sequence of SubjectKeyIdentifier, selected from <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3>.
EUICC_CS_SIGNATURE	The eUICC cancel session signature computed using the #SK_EUICC_SIG across the EuiccCancelSessionSigned present in the CancelSessionResponse structure
EUICC_RSP_CAPABILITY	`EuiccRspCapability`, coded as ASN.1 BIT STRING to be used for indication of additionalProfile, loadcrlSupport, rpmSupport , testProfileSupport, etc.
EUICC_SIGN_PIR	The eUICC signature of the Profile Installation Result (PIR). The input data used to generate the <EUICC_SIGN_PIR> is the profileInstallationResultData TLV.
EUICC_SIGN_RPR	The eUICC signature of the Load RPM Package Result (RPR). The input data used to generate the <EUICC_SIGN_RPR> is the loadRpmPackageResultDataSigned and smdpSignature3. euiccSignRPR shall be created using the SK.EUICC.SIG and verified using the PK.EUICC.SIG.
EUICC_SIGNATURE1	The eUICC signature 1 (euiccSignature1) computed using #SK_EUICC_SIG across the euiccSigned1 present in the AuthenticateServerResponse structure, coded as ASN.1 OCTET STRING.
EUICC_SIGNATURE2	The eUICC signature 2 (euiccSignature2) computed using the #SK_EUICC_SIG across the following data objects: • euiccSigned2 • smdpSignature2 present in the PrepareDownloadRequest structure
EXT_CARD_RESOURCE	Extended Card Resource Information according to ETSI TS 102 226 [14], coded as ASN.1 OCTET STRING. 'Number of installed application' value field is '00'.
FREE_MEM_OP_PROF_INSTALLED	Non-volatile memory (tag 0x82) available in the eUICC when two or more PROFILE_OPERATIONAL are installed.
FREE_MEM_OP_PROF1_DELETED	Non-volatile memory (tag 0x82) available in the eUICC after PROFILE_OPERATIONAL1 deletion.
FREE_MEM_OP_PROF1_INSTALLED	Non-volatile memory (tag 0x82) available in the eUICC when only PROFILE_OPERATIONAL1 is installed.
FREE_MEMORY_NO_PROFILE	Non-volatile memory (tag 0x82) available in the eUICC when there is no Profile installed.
INVALID_TRANSACTION_ID	A Transaction Identifier generated by the S_SM-DP+ or the S_SM-DS that SHALL be different from

Variable	Description
	<S_TRANSACTION_ID> if exists. Otherwise, a random value is generated.
ISD_P_AID	The ISD-P AID newly created in the eUICC. This AID value is in the range from 0xA0 00 00 05 59 10 10 FF FF FF FF 89 00 00 10 00 to 0xA0 00 00 05 59 10 10 FF FF FF FF 89 00 FF FF 00. Last byte is set to '00' as defined in SGP.02[1].
ISD_P_AID1	The ISD-P AID created in the eUICC for the PROFILE_OPERATIONAL1. This AID value belongs to the range from 0xA0 00 00 05 59 10 10 FF FF FF FF 89 00 00 10 00 to 0xA0 00 00 05 59 10 10 FF FF FF FF 89 00 FF FF 00. Last byte is set to '00' as defined in SGP.02[1].
ISD_P_AID2	The ISD-P AID created in the eUICC for the PROFILE_OPERATIONAL2. This AID value belongs to the range from 0xA0 00 00 05 59 10 10 FF FF FF FF 89 00 00 10 00 to 0xA0 00 00 05 59 10 10 FF FF FF FF 89 00 FF FF 00. Last byte is set to '00' as defined in SGP.02[1].
ISD_P_AID3	The ISD-P AID created in the eUICC for the PROFILE_OPERATIONAL3. This AID value belongs to the range from 0xA0 00 00 05 59 10 10 FF FF FF FF 89 00 00 10 00 to 0xA0 00 00 05 59 10 10 FF FF FF FF 89 00 FF FF 00. Last byte is set to '00' as defined in SGP.02[1].
ISD_P_AID4	The ISD-P AID created in the eUICC for the PROFILE_OPERATIONAL4. This AID value belongs to the range from 0xA0 00 00 05 59 10 10 FF FF FF FF 89 00 00 10 00 to 0xA0 00 00 05 59 10 10 FF FF FF FF 89 00 FF FF 00. Last byte is set to '00' as defined in SGP.02[1].
ISD_P_AIDX	An invalid ISD-P AID not present on the eUICC. This AID value is in the range from 0xA0 00 00 05 59 10 10 FF FF FF FF 89 00 00 10 00 to 0xA0 00 00 05 59 10 10 FF FF FF FF 89 00 FF FF 00.
L	Exact length of the corresponding tag or of the remaining data.
LSI_COMMAND_ACTION	Action returned in the proactive command LSI COMMAND
LSI_NUMBER	LSI numbers returned in the proactive command LSI COMMAND
LSI_SUPPORT	TRUE if 'LSI support' is present in the ATR
MEP_LSI_OPTIONS	LSI Options returned in the MANAGE LSI(Configure LSI) response
MEP_MAX_LSIS	Maximum number of LSIs supported for Enabled Profiles
MEP_MODE	The 'Jointly supported MEP mode' returned in the MANAGE LSI(Configure LSI) response
MEP-A2_TARGET_ESIM_PORT	Target eSIM Port returned in the ES10c.EnableProfileResponse for MEP-A2.
MNO_SCP80_COUNTER	SCP80 counter of the MNO-SD related to the KVN 0x01 (5 bytes long). Initial value is set to 0x00 00 00 00 01 and is incremented by one each time a secured packet is sent.
NB_EXECUTED_C_APDUS	Number of executed Command TLV objects as defined in ETSI TS 102 226 [14].

Variable	Description
NEXT_EUICC_CERT_IN_CHAIN	The Certificate certifying the eUICC Certificate #CERT_EUICC_SIG.
NOTIF_SEQ_NO_DE1	The Sequence Number of the Delete Notification related to the PROFILE_OPERATIONAL1.
NOTIF_SEQ_NO_DE2	The Sequence Number of the Delete Notification related to the PROFILE_OPERATIONAL2.
NOTIF_SEQ_NO_DI1	The Sequence Number of the Disable Notification related to the PROFILE_OPERATIONAL1.
NOTIF_SEQ_NO_EN1	The Sequence Number of the Enable Notification related to the PROFILE_OPERATIONAL1.
NOTIF_SEQ_NO_EN1_RPM	The Sequence Number of the RPM Enable Notification related to the PROFILE_OPERATIONAL1.
NOTIF_SEQ_NO_EN2	The Sequence Number of the Enable Notification related to the PROFILE_OPERATIONAL2.
NOTIF_SEQ_NO_IN1	The Sequence Number of the Install Notification related to the PROFILE_OPERATIONAL1.
NOTIF_SEQ_NO_IN1_PIR	The Sequence Number of the Install Notification (PIR) related to the PROFILE_OPERATIONAL1.
NOTIF_SEQ_NO_IN2	The Sequence Number of the Install Notification related to the PROFILE_OPERATIONAL2.
NOTIF_SEQ_NO_IN2_PIR	The Sequence Number of the Install Notification (PIR) related to the PROFILE_OPERATIONAL2.
NOTIF_SEQ_NO_PROF1_RPR	The Sequence Number of the Notification RPM Package Result related to the PROFILE_OPERATIONAL1.
NOTIF_SEQ_NO2_DE1	The Sequence Number of the second Delete Notification related to the PROFILE_OPERATIONAL1.
NOTIF_SEQ_NO2_DI1	The Sequence Number of the second Disable Notification related to the PROFILE_OPERATIONAL1.
NOTIF_SEQ_NO2_EN1	The Sequence Number of the second Enable Notification related to the PROFILE_OPERATIONAL1.
OT_SK_S_SM_DP+_ECKA	One-time Private Key generated by the S_SM-DP+ for ECKA. Depending on the eUICC configuration, this key is based on NIST P-256, brainpoolP256r1 or FRP256V1.
OTHER_EUICC_CERT_IN_CHAIN	Remaining part of the certificate chain certifying the <NEXT_EUICC_CERT_IN_CHAIN>.
OTPK_S_SM_DP+_ECKA	One-time Public Key generated by the S_SM-DP+ for ECKA. Depending on the eUICC configuration, this key is based on NIST P-256, brainpoolP256r1 or FRP256V1.
PPK_ENC	Random PPK-ENC value (16 bytes key length). This value is different from <S_ENC> value.
PPK_INIT_MAC	Random initial MAC chaining value (16 bytes). This value is different from the <S_MAC_CHAIN> value.
PPK_MAC	Random PPK-MAC value (16 bytes key length). This value is different from <S_MAC> value.
PPR_IDS	Forbidden Profile Policy Rules. This PPR list MAY be empty or MAY contain either PPR1 or PPR2 or both.

Variable	Description
PROPRIETARY_DATA	Proprietary Data returned by the eUICC as part of FCI template.
RANDOM_SM_DP+_SIGN	Random SM-DP+ signature (i.e. content of the tag 0x5F37) with a size corresponding to a valid one.
RANDOM_SM_DS_SIGN	Random SM-DS signature (i.e. content of the tag 0x5F37) with a size corresponding to a valid one.
S_ENC	SCP03T Encryption Session key (128 bits length) resulting from the key agreement with eUICC.
S_HASHED_CC	Hashed Confirmation Code generated by the LPA.
S_INIT_MAC	SCP03T Initial MAC chaining value (128 bits length) resulting from the key agreement with eUICC.
S_MAC	SCP03T MACing Session key (128 bits length) resulting from the key agreement with eUICC.
S_MAC_CHAIN	Current MAC chaining value used for SCP03t BPP protection.
S_SM_DP+_SIGN	The S_SM-DP+ signature (smdpSign), computed using the #SK_S_SM_DPpb_SIG across the following data objects: • remoteOpId • transactionId • controlRefTemplate • smdpOtpk • euiccOtpk, as provided earlier in the prepareDownloadResponse data object
S_SM_DP+_SIGNATURE2	The ASN.1 OCTET STRING encoded SM-DP+ signature 2 (field smdpSignature2) computed using the private key related to the server certificate (field smdpCertificate) present in the PrepareDownloadRequest structure. This signature SHALL be generated across the following data objects: • smdpSignature2 • euiccSignature1 present in the AuthenticateServerResponse structure
S_SM_DP+_SIGNATURE3	The ASN.1 OCTET STRING encoded SM-DP+ signature 3 (field `smdpSignature3`) is computed using the SK.DPauth.SIG. This signature SHALL be generated across the following data objects: • smdpSigned3 • euiccSignature1 present in the AuthenticateServerResponse structure
S_SMDP_CHALLENGE	The SM-DP+ Challenge (serverChallenge) randomly chosen by the simulated SM-DP+ to be signed later by the eUICC for the eUICC authentication, coded as ASN.1 OCTET STRING of 16 bytes.
S_SMDP_SIGNATURE1	The ASN.1 OCTET STRING encoded SM-DP+ signature (field serverSignature1) computed using the private key related to the server certificate (field serverCertificate) present in the AuthenticateServerRequest structure.
S_SMDS_CHALLENGE	The SM-DS Challenge (serverChallenge) randomly chosen by the simulated SM-DS to be signed later by the eUICC for the eUICC authentication, coded as ASN.1 OCTET STRING of 16 bytes.

Variable	Description
S_SMDS_SIGNATURE1	The SM-DS signature 1 (serverSignature1) computed using #SK_S_SM_DSauth_SIG across the serverSigned1 present in the AuthenticateServerRequest structure, coded as ASN.1 OCTET STRING.
S_TRANSACTION_ID	The TransactionID (Unique Transaction Identifier) generated by the (S_)SM-DP+, or (S_)SM-DS which is used to uniquely identify the RSP session and to correlate the multiple ESXX request messages that belong to the same RSP session. This value (binary value) can start from 0x01 and can be increased by 1 each time a Profile is downloaded in the eUICC. 1-16 bytes (ASN.1 OCTET STRING).
SEQ_NUMBER	Sequence Number related to a Notification Metadata generated by the eUICC.
SHS	Shared Secret resulting from the key agreement with eUICC.
STORE_DATA_BLOCK_NUM	The STORE DATA block number coded sequentially from 0x00 to 0xFF. If the value 0xFF has been reached and more STORE DATA commands are needed to complete the transfer, the numbering restarts and the next STORE DATA block number is set to 0x00.
TBS_EUICC_NOTIF_SIG	The eUICC signature generated over tbsOtherNotification. NotificationMetadata, coded as ASN.1 OCTET STRING.
TRE_PROPERTIES	The value of the treProperties field in EUICCInfo2.
TRE_REFERENCE	The value of the treProductReference field in EUICCInfo2.

Annex C Methods And Procedures

This section describes methods and procedures used in the interfaces compliance test cases. They are part of test cases and SHALL not be executed in standalone mode.

C.1 Methods

If the method is used in the “expected result” column, all parameters SHALL be verified by the simulated entity (test tool). If the method is used in the “Sequence / Description” column, the command SHALL be generated by the simulated entity.

Method	MTD_AUTHENTICATE_SMDP
Description	Generate an ASN.1 `AuthenticateServerRequest` structure according to the input parameters
Parameter(s)	• paramServerAddress: the SM-DP+ or the SM-DS FQDN • paramServerChallenge: the SM-DP+ or the SM-DS challenge • paramCtxParams1: the CtxParams1 to use • paramServerSignature: the RSP Server signature • paramServerCertificate: the RSP Server Certificate CERT.XXauth.SIG • paramOtherCertsInChain (OPTIONAL): the remaining part of the CERT.XXauth.SIG certificate chain, if any • paramCrlList: CRL list to use • paramNewCertVariant: true if new certificates variants used

Details	Generate the following ASN.1 structure:
	
	`req AuthenticateServerRequest ::= {`
	`  serverSigned1 {`
	`  transactionId <S_TRANSACTION_ID>,`
	`  euiccChallenge <EUICC_CHALLENGE>,`
	`  serverAddress paramServerAddress,`
	`  serverChallenge paramServerChallenge,`
	`  sessionContext {`
	`  serverSvn #IUT_RSP_VERSION_HIGHEST,`
	`  crlStaplingV3Used TRUE,`
	`  euiccCiPKIdToBeUsedV3 <EUICC_CI_PK_ID_TO_BE_USED_V3>`
	`  },`
	`  serverRspCapability {`
	`  crlStaplingV3Support,`
	`  cancelForEmptySpnPnSupport`
	`  }`
	`  },`
	`  serverSignature1 paramServerSignature,`
	`  euiccCiPKIdToBeUsed`
	`  <EUICC_CI_PK_ID_TO_BE_USED>,`
	`  serverCertificate paramServerCertificate,`
	`  ctxParams1 paramCtxParams1,`
	`  otherCertsInChain { paramOtherCertsInChain },`
	`  crlList { paramCrlList }`
	`}`
	
	`otherCertsInChain` SHALL be present in the `AuthenticateServerRequest` only if `paramOtherCertsInChain` is provided
	if paramNewCertVariant is true then `euiccCiPKIdToBeUsedV3` SHALL be present in `sessionContext` and `euiccCiPKIdToBeUsed` SHALL be omitted. Otherwise, if if

	paramNewCertVariant is false then `euiccCiPKIdToBeUsed` SHALL be present and `euiccCiPKIdToBeUsedV3` SHALL be omitted in `sessionContext`.

Method	MTD_AUTHENTICATE_SMDP_USING_IUT_VARIANT
Description	Generate an ASN.1 `AuthenticateServerRequest` structure according to the input parameters, specifying the CI PK to use for signing accordingly with the IUT setting IUT_EUICC_CERT_CHAIN_VARIANT.
Parameter(s)	• paramServerAddress: the SM-DP+ or the SM-DS FQDN • paramServerChallenge: the SM-DP+ or the SM-DS challenge • paramCtxParams1: the CtxParams1 to use • paramServerSignature: the RSP Server signature • paramServerCertificate: the RSP Server Certificate CERT.XXauth.SIG • paramOtherCertsInChain (OPTIONAL): the remaining part of the CERT.XXauth.SIG certificate chain, if any • paramCrlList: CRL list to use

Details	Generate the following ASN.1 structure:
	
	`req AuthenticateServerRequest ::= {`
	`  serverSigned1 {`
	`  transactionId <S_TRANSACTION_ID>,`
	`  euiccChallenge <EUICC_CHALLENGE>,`
	`  serverAddress paramServerAddress,`
	`  serverChallenge paramServerChallenge,`
	`  sessionContext {`
	`  serverSvn #IUT_RSP_VERSION_HIGHEST,`
	`  crlStaplingV3Used TRUE,`
	`  euiccCiPKIdToBeUsedV3 <EUICC_CI_PK_ID_TO_BE_USED_V3>`
	`  },`
	`  serverRspCapability {`
	`  crlStaplingV3Support,`
	`  cancelForEmptySpnPnSupport`
	`  }`
	`  },`
	`  serverSignature1 paramServerSignature,`
	`  euiccCiPKIdToBeUsed`
	`  <EUICC_CI_PK_ID_TO_BE_USED>,`
	`  serverCertificate paramServerCertificate,`
	`  ctxParams1 paramCtxParams1,`
	`  otherCertsInChain { paramOtherCertsInChain },`
	`  crlList { paramCrlList }`
	`}`
	
	`otherCertsInChain` SHALL be present in the `AuthenticateServerRequest` only if `paramOtherCertsInChain` is provided
	if IUT_EUICC_CERT_CHAIN_VARIANT is one of Ov3, A, B or C, then `euiccCiPKIdToBeUsedV3` SHALL be present in `sessionContext` and `euiccCiPKIdToBeUsed` SHALL be omitted. Otherwise, `euiccCiPKIdToBeUsed` SHALL be present, and `euiccCiPKIdToBeUsedV3` SHALL be omitted in `sessionContext`.

Method	MTD_AUTHENTICATE_SMDS
Description	Generate an ASN.1 `AuthenticateServerRequest` structure according to the input parameters
Parameter(s)	• paramServerAddress: the SM-DP+ or the SM-DS FQDN • paramServerChallenge: the SM-DP+ or the SM-DS challenge • paramCtxParams1: the CtxParams1 to use • paramServerSignature: the RSP Server signature • paramServerCertificate: the RSP Server Certificate CERT.XXauth.SIG • paramOtherCertsInChain (OPTIONAL): the remaining part of the CERT.XXauth.SIG certificate chain, if any • paramCrlList: CRL list to use • paramNewCertVariant: true if new certificates variants used

Details	Generate the following ASN.1 structure:
	
	`req AuthenticateServerRequest ::= {`
	`  serverSigned1 {`
	`  transactionId <S_TRANSACTION_ID>,`
	`  euiccChallenge <EUICC_CHALLENGE>,`
	`  serverAddress paramServerAddress,`
	`  serverChallenge paramServerChallenge,`
	`  sessionContext {`
	`  serverSvn #IUT_RSP_VERSION_HIGHEST,`
	`  crlStaplingV3Used TRUE,`
	`  euiccCiPKIdToBeUsedV3 <EUICC_CI_PK_ID_TO_BE_USED_V3>`
	`  },`
	`  serverRspCapability {`
	`  crlStaplingV3Support,`
	`  eventListSigningV3Support`
	`  }`
	`  },`
	`  serverSignature1 paramServerSignature,`
	`  euiccCiPKIdToBeUsed`
	`  <EUICC_CI_PK_ID_TO_BE_USED>,`
	`  serverCertificate paramServerCertificate,`
	`  ctxParams1 paramCtxParams1,`
	`  otherCertsInChain { paramOtherCertsInChain },`
	`  crlList { paramCrlList }`
	`}`
	
	`otherCertsInChain` SHALL be present in the `AuthenticateServerRequest` only if paramOtherCertsInChain is provided
	
	if paramNewCertVariant is true then `euiccCiPKIdToBeUsedV3` SHALL be present in `sessionContext` and `euiccCiPKIdToBeUsed` SHALL be omitted. Otherwise, if if paramNewCertVariant is false then `euiccCiPKIdToBeUsed` SHALL be present and `euiccCiPKIdToBeUsedV3` SHALL be omitted in `sessionContext`.

Method	MTD_AUTHENTICATE_SMDS_USING_IUT_VARIANT
Description	Generate an ASN.1 `AuthenticateServerRequest` structure according to the input parameters, specifying the CI PK to use for signing accordingly with the IUT setting IUT_EUICC_CERT_CHAIN_VARIANT.

Parameter(s)	• paramServerAddress: the SM-DP+ or the SM-DS FQDN • paramServerChallenge: the SM-DP+ or the SM-DS challenge • paramCtxParams1: the CtxParams1 to use • paramServerSignature: the RSP Server signature • paramServerCertificate: the RSP Server Certificate CERT.XXauth.SIG • paramOtherCertsInChain (OPTIONAL): the remaining part of the CERT.XXauth.SIG certificate chain, if any • paramCrlList: CRL list to use

Details	Generate the following ASN.1 structure:
	
	`req AuthenticateServerRequest ::= {`
	`  serverSigned1 {`
	`  transactionId <S_TRANSACTION_ID>,`
	`  euiccChallenge <EUICC_CHALLENGE>,`
	`  serverAddress paramServerAddress,`
	`  serverChallenge paramServerChallenge,`
	`  sessionContext {`
	`  serverSvn #IUT_RSP_VERSION_HIGHEST,`
	`  crlStaplingV3Used TRUE,`
	`  euiccCiPKIdToBeUsedV3 <EUICC_CI_PK_ID_TO_BE_USED_V3>`
	`  },`
	`  serverRspCapability {`
	`  crlStaplingV3Support,`
	`  eventListSigningV3Support`
	`  }`
	`  },`
	`  serverSignature1 paramServerSignature,`
	`  euiccCiPKIdToBeUsed`
	`  <EUICC_CI_PK_ID_TO_BE_USED>,`
	`  serverCertificate paramServerCertificate,`
	`  ctxParams1 paramCtxParams1,`
	`  otherCertsInChain { paramOtherCertsInChain },`
	`  crlList { paramCrlList }`
	`}`
	
	`otherCertsInChain` SHALL be present in the `AuthenticateServerRequest` only if paramOtherCertsInChain is provided
	
	If IUT_EUICC_CERT_CHAIN_VARIANT is one of Ov3, A, B or C then `euiccCiPKIdToBeUsedV3` SHALL be present in `sessionContext` and `euiccCiPKIdToBeUsed` SHALL be omitted. Otherwise, `euiccCiPKIdToBeUsed` SHALL be present and `euiccCiPKIdToBeUsedV3` SHALL be omitted in `sessionContext`.

Method	MTD_CHECK_AUTH_ERROR_RESP
Description	Check an ASN.1 `AuthenticateServerResponse` structure in case of failure according to the input parameters
Parameter(s)	• paramAuthenticateErrorCode: the expected error code
Details	Verify that the received AuthenticateServerResponse is structured as follow:

	`resp AuthenticateServerResponse ::= authenticateResponseError : {`
	`  transactionId <S_TRANSACTION_ID>,`
	`  authenticateErrorCode paramAuthenticateErrorCode`
	`}`
	
	Verify that the <S_TRANSACTION_ID> present in the authenticateResponseError is the same as in the AuthenticateServerRequest.

Method	MTD_CHECK_AUTH_SERVER_RESP
Description	Check an ASN.1 `AuthenticateServerResponse` structure according to the input parameters
Parameter(s)	• paramServerAddress: the expected SM-DP+ or the SM-DS FQDN • paramServerChallenge: the expected SM-DP+ or SM-DS challenge • paramCtxParams1: the expected `CtxParams1`

Details	Verify that the received AuthenticateServerResponse is structured as follow:
	
	`resp AuthenticateServerResponse ::= authenticateResponseOk : {`
	`  euiccSigned1 {`
	`  transactionId <S_TRANSACTION_ID>,`
	`  serverAddress paramServerAddress,`
	`  serverChallenge paramServerChallenge,`
	`  euiccInfo2 {...},`
	`  ctxParams1 paramCtxParams1`
	`  },`
	`  euiccSignature1 <EUICC_SIGNATURE1>,`
	`  euiccCertificate #CERT_EUICC_SIG,`
	`  nextCertInChain <NEXT_EUICC_CERT_IN_CHAIN>,`
	`  otherCertsInChain <OTHER_EUICC_CERT_IN_CHAIN> -- OPTIONAL`
	`}`
	
	Verify that `euiccInfo2` is present and contains a valid TLV ASN.1 structure. At this level, only the format of the TLV structure is verified.
	
	Verify the `<EUICC_SIGNATURE1>` across the `euiccSigned1` by using the #PK_EUICC_SIG.
	
	Verify that the `<S_TRANSACTION_ID>` present in the `euiccSigned1` is the same as in the `AuthenticateServerRequest`.
	
	If `#IUT_EUICC_CERT_CHAIN_VARIANT` = ‘A’ Then
	• Verify that `<NEXT_EUICC_CERT_IN_CHAIN>` matches #CERT_EUM_SubCA_SIG
	• Verify that `<OTHER_EUICC_CERT_IN_CHAIN>` matches #CERT_EUM_SIG
	Else If `#IUT_EUICC_CERT_CHAIN_VARIANT` = ‘B’ Then
	• Verify that `<NEXT_EUICC_CERT_IN_CHAIN>` matches #CERT_EUM_SIG
	• Verify that `<OTHER_EUICC_CERT_IN_CHAIN>` matches #CERT_CI_SubCA_SIG
	Else If `#IUT_EUICC_CERT_CHAIN_VARIANT` = ‘C’ Then
	-- meaning that Variant C is supported

	• Verify that <NEXT_EUICC_CERT_IN_CHAIN> matches #CERT_EUM_SubCA_SIG
	• Verify that <OTHER_EUICC_CERT_IN_CHAIN> matches { #CERT_EUM_SIG, #CERT_CI_SubCA_SIG }
	Else
	-- meaning that Variant O or Variant Ov3 is supported
	• Verify that <NEXT_EUICC_CERT_IN_CHAIN> matches #CERT_EUM_SIG
	• Verify that <OTHER_EUICC_CERT_IN_CHAIN> is not present
	End if
	
	Verify that the eUICC Certificate chain leads to the Root CI Key Identifier set in the euiccCiPKIdToBeUsedV3 or in the euiccCiPKIdToBeUsed of the corresponding AuthenticateServerRequest.

Method	MTD_CHECK_SMS_POR
Description	Check the content of the SMS POR containing the response of the ES6.UpdateMetadata request
Parameter(s)	• paramExpectedSW: the expected Status Word of the last STORE DATA command

Details	Parse and retrieve the SCP80 response packet from the SMS.
	SCP80 response status code SHALL be equal to 0x00 – POR OK.
	The additional data from the response packet SHALL be formatted as an expanded structure with definite length as defined in ETSI TS 102 226 [14] and contains the following TLV:
	`AB <L>`
	`  80 <L> <NB_EXECUTED_C_APDUS> -- Number of executed C-APDUs`
	`  23 <L> 00 90 00 -- R-APDU of the INSTALL FOR PERSONALIZATION command`
	`  23 <L> paramExpectedSW -- SW of the last STORE DATA command executed`
	
	<NB_EXECUTED_C_APDUS> SHALL be equal to the number of executed C-APDUs (i.e. one INSTALL FOR PERSONALIZATION + n STORE DATA command(s))

Method	MTD_DELETE_PROFILE
Description	Generate the ASN.1 DeleteProfileRequest structure according to the input parameters.
Parameter(s)	• paramIccidValue: The ICCID of the Profile to Delete (optional) • paramIsdpAidValue: The ISD-P AID of the Profile to Delete (optional) Either paramIccidValue or paramIsdpAidValue is passed as a parameter.

Details	IF paramIccidValue is provided Then
	`req DeleteProfileRequest ::= iccid : paramIccidValue`
	Else
	`req DeleteProfileRequest ::= isdpAid : paramIsdpAidValue`
	End if

Method	MTD_DISABLE_PROFILE
Description	Generate the ASN.1 DisableProfileRequest structure according to the input parameters.
Parameter(s)	• paramIccidValue: The ICCID of the Profile to Disable (optional) • paramIsdpAidValue: The ISD-P AID of the Profile to Disable (optional) • paramRefreshFlag: Boolean, TRUE if refreshFlag SHALL be set, FALSE otherwise Either paramIccidValue or paramIsdpAidValue is passed as a parameter.

Details	IF paramIccidValue is provided Then
	`req DisableProfileRequest::= {`
	`profileIdentifier iccid : paramIccidValue,`
	`refreshFlag paramRefreshFlag`
	`}`
	Else
	`req DisableProfileRequest::= {`
	`profileIdentifier isdpAid : paramIsdpAidValue,`
	`refreshFlag paramRefreshFlag`
	`}`
	End if

Method	MTD_ENABLE_PROFILE
Description	Generate the ASN.1 EnableProfileRequest structure according to the input parameters.
Parameter(s)	• paramIccidValue: The ICCID of the Profile to Enable (optional) • paramIsdpAidValue: The ISD-P AID of the Profile to Enable (optional) • paramRefreshFlag: Boolean, TRUE if refreshFlag SHALL be set, FALSE otherwise Either paramIccidValue or paramIsdpAidValue is passed as a parameter.

Details	IF paramIccidValue is provided Then
	`req EnableProfileRequest ::= {`
	`profileIdentifier iccid : paramIccidValue,`
	`refreshFlag paramRefreshFlag`
	`}`
	Else
	`req EnableProfileRequest ::= {`
	`profileIdentifier isdpAid : paramIsdpAidValue,`
	`refreshFlag paramRefreshFlag`
	`}`
	End if

Method	MTD_ENABLE_PROFILE_MEP_A1
Description	Generates the ASN.1 EnableProfileRequest structure for MEP capable eUICCs configured with MEP-A1 according to the input parameters.
Parameter(s)	• paramIccidValue: The ICCID of the Profile to Enable (optional) • paramIsdpAidValue: The ISD-P AID of the Profile to Enable (optional) • paramRefreshFlag: Boolean, TRUE if refreshFlag SHALL be set, FALSE otherwise (optional) • paramTargetEsimPort: target eSIM port (mandatory) Either paramIccidValue or paramIsdpAidValue is passed as a parameter. paramRefreshFlag is required depending on the test scenario.

Details	If paramIccidValue is provided Then
	`req EnableProfileRequest ::= {`
	`  profileIdentifier iccid : paramIccidValue,`
	`  refreshFlag : paramRefreshFlag,`
	`  targetEsimPort : paramTargetEsimPort`
	`}`
	Else
	`req EnableProfileRequest ::= {`
	`  profileIdentifier isdpAid : paramIsdpAidValue,`
	`  refreshFlag : paramRefreshFlag,`
	`  targetEsimPort : paramTargetEsimPort`
	} End if

Method	MTD_GENERATE_BPP
Description	Generate a BPP according to the input parameters.
Parameter(s)	• paramInitSC: The InitialiseSecureChannel request • paramConfISDP: The ConfigureISDP request (plain) • paramStoreMetadata: The StoreMetadata request (plain) • paramReplaceSessionKeys: The ReplaceSessionKeys request (plain) – Optional parameter • paramUPP: The Unprotected Profile Package to download

Details	Split the paramStoreMetadata in several segments of maximum 1008 bytes. Each Metadata segment is named <METADATA_SEG> here after.
	Split the paramUPP in several segments of maximum 1007 bytes. Each UPP segment named <UPP_SEG> here after.
	Create the following structure of data:
	`req BoundProfilePackage ::= {`
	`  paramInitSC,`
	`  firstSequenceOf87 {`
	`  0x87 <L> paramConfISDP`
	`  },`
	`  sequenceOf88 {`
	`  0x88 <L> <METADATA_SEG>,`
	`  ...`
	`  0x88 <L> <METADATA_SEG>`

	},
	-- secondSequenceOf87 SHALL be set only if paramReplaceSessionKeys is provided
	secondSequenceOf87 {
	0x87 <L> paramReplaceSessionKeys
	},
	sequenceOf86 {
	0x86 <L> <UPP_SEG>,
	...
	0x86 <L> <UPP_SEG>
	}
	}
	Use <OT_SK_S_SM_DP+_ECKA> and <OTPK_EUICC_ECKA> in order to generate the <SHS>.
	Concatenate #KEY_TYPE, #KEY_LENGTH, <L> #HOST_ID and <L> #EID1 as SharedInfo.
	Retrieve <S_ENC>, <S_MAC> and <S_INIT_MAC> across SHA-256 calculated from <SHS> and SharedInfo.
	
	Encrypt paramConfISDP with <S_ENC>.
	Calculate and add a MAC to the tag 0x87 of firstSequenceOf87 by using <S_MAC>.
	
	Calculate and add a MAC to all tags 0x88 of sequenceOf88 by using <S_MAC>.
	
	If paramReplaceSessionKeys is provided Then
	Encrypt paramReplaceSessionKeys with <S_ENC>
	Calculate and add a MAC to the tag 0x87 of secondSequenceOf87 by using <S_MAC>.
	End If
	
	Encrypt all <UPP_SEG> with <S_ENC>, or <PPK_ENC> if paramReplaceSessionKeys is provided.
	Calculate and add a MAC to all tags 0x86 of sequenceOf86 by using <S_MAC>, or <PPK_MAC> (and <PPK_INIT_MAC> for the first tag) if paramReplaceSessionKeys is provided.

Method	MTD_GENERATE_HASHED_CC
Description	Generate an Hashed Confirmation Code based on the Confirmation Code and the Transaction ID given in parameter.
Parameter(s)	• paramConfirmationCode: The Confirmation Code (plain) • paramTransactionId: The Transaction ID (plain)
Details	Generate a SHA-256 of the paramConfirmationCode. Concatenate the obtained hash value with the paramTransactionId. Generate and return a SHA-256 of these two concatenated elements.

Method	MTD_GET_PROFILE_INFO
Description	Generate the ASN.1 ProfileInfoListRequest according to the input parameters.
Parameter(s)	• paramIccidValue: The ICCID of the Profile • paramIsdpAidValue: The ISD-P AID of the Profile

	Either paramIccidValue or paramIsdpAidValue is passed as a parameter.

Details	If paramIccidValue is provided Then
	`req ProfileInfoListRequest::= {`
	`searchCriteria iccid: paramIccidValue`
	`}`
	`Else if paramIsdpAidValue is provided then`
	`req ProfileInfoListRequest::= {`
	`searchCriteria isdpAid: paramIsdpAidValue`
	`}`
	`Else`
	`req ProfileInfoListRequest::= {`
	`}`
	`End If`

Method	MTD_GET_PROFILE_INFO_TAGLIST
Description	Generates the ASN.1 ProfileInfoListRequest to retrieve specified tags for all installed Profiles.
Parameter(s)	• paramTagList: list of target Tags to get Profile metadata of all installed profiles (mandatory)

Details	req ProfileInfoListRequest::= {
	tagList: paramTagList
	}

Method	MTD_LOAD_RPM_PKG_REQ_FOR_LIST_PROFILE_INFO
Description	This Method can be used for creating LoadRpmPackageRequest for rpmPackage with single RPM command.
Parameter(s)	• paramTransactionId: The Transaction Id (Mandatory) • paramIccidValue: The ICCID within RPM Command • paramSmdpSignature3: The SM-DP+ Signature3 (Mandatory) • paramProfileOwnerOidValue: The ProfileOwnerOid for seachCriteria in ListProfileInfo if required • paramTagListValue: The TagList required for ListProfileInfo if required Parameters paramTagListValue and paramProfileOwnerOidValue shall be passed and can be empty.

Details	If paramTagListValue is not present Then
	If paramProfileOwnerOidValue is not present Then
	`loadRpmPkgReq LoadRpmPackageRequest ::= {`
	`smdpSigned3 {`
	`transactionId paramTransactionId,`
	`rpmPackage {`
	`{`
	`continueOnFailure NULL,`
	`rpmCommandDetails listProfileInfo : {`
	`searchCriteria iccid : {paramIccidValue}`
	`}`
	`}`
	`}`
	`},`
	`smdpSignature3 paramSmdpSignature3`

```
        }
    Else
    loadRpmPkgReq LoadRpmPackageRequest ::= {
      smdpSigned3 {
        transactionId paramTransactionId,
        rpmPackage {
         {
          continueOnFailure NULL,
          rpmCommandDetails listProfileInfo : {
            searchCriteria profileOwnerOid :
   {paramProfileOwnerOidValue}
          }
         }
        }
      },
      smdpSignature3 paramSmdpSignature3
          }
Else
   If paramProfileOwnerOidValue is not present Then
    loadRpmPkgReq LoadRpmPackageRequest ::= {
      smdpSigned3 {
        transactionId paramTransactionId,
        rpmPackage {
         {
          continueOnFailure NULL,
          rpmCommandDetails listProfileInfo : {
             searchCriteria iccid : {paramIccidValue},
             tagList { paramTagListValue}
         }
        }
        }
      },
      smdpSignature3 paramSmdpSignature3
          }
  Else if
    loadRpmPkgReq LoadRpmPackageRequest ::= {
      smdpSigned3 {
        transactionId paramTransactionId,
        rpmPackage {
         {
           continueOnFailure NULL,
           rpmCommandDetails listProfileInfo : {
             searchCriteria profileOwnerOid :
   {paramProfileOwnerOidValue},
             tagList { paramTagListValue}
          }
         }
        }
      },
      smdpSignature3 paramSmdpSignature3
          }
End if
```

Method	MTD_LOAD_RPM_PKG_REQ_MULT_CMNDS
Description	This Method can be used to generate LoadRpmPackageRequest with rpmPackage containing multiple RPM commands.
Parameter(s)	• paramRpmComandList: The rpm commands list (Mandatory)

Details	loadRpmPkgReq LoadRpmPackageRequest ::= {
	smdpSigned3 {
	transactionId <S_TRANSACTION_ID>,
	rpmPackage {
	paramRpmComandList
	}
	}
	},
	smdpSignature3 <S_SM_DP+_SIGNATURE3>
	}

Method	MTD_LOAD_RPM_PKG_REQ_SINGLE_CMND
Description	This Method can be used for creating LoadRpmPackageRequest for rpmPackage with single RPM command.
Parameter(s)	• paramComandDetailsChoice: The rpmCommandDetails choice (Mandatory) • paramTransactionId: The Transaction Id (Mandatory) • paramIccidValue: The ICCID within RPM Command (Mandatory) • paramSmdpSignature3: The SM-DP+ Signature3 (Mandatory) • paramUpdateMetadataRequest: The UpdateMetadataRequest if required • paramDpiRpmValue: The dpiRpm within RPM Command if required Parameters paramUpdateMetadataRequest and paramDpiRpmValue shall be passed and can be empty depend on the paramComandDetailsChoice.

Details	If paramComandDetailsChoice = enable or disable or delete Then
	loadRpmPkgReq LoadRpmPackageRequest ::= {
	smdpSigned3 {
	transactionId paramTransactionId,
	rpmPackage {
	{
	continueOnFailure NULL,
	rpmCommandDetails paramComandDetailsChoice : {
	iccid paramIccidValue
	}
	}
	}
	},
	smdpSignature3 paramSmdpSignature3
	}
	Else if paramComandDetailsChoice = updateMetadata Then
	loadRpmPkgReq LoadRpmPackageRequest ::= {
	smdpSigned3 {
	transactionId paramTransactionId,
	rpmPackage {
	{
	continueOnFailure NULL,
	rpmCommandDetails updateMetadata : {

```
              iccid paramIccidValue
              updateMetadataRequest {paramUpdateMetadataRequest}
            }
          }
        }
      },
      smdpSignature3 paramSmdpSignature3
         }
```

Else if paramComandDetailsChoice = contactPcmp Then

If paramDpiRpmValue is not present Then

```
    loadRpmPkgReq LoadRpmPackageRequest ::= {
      smdpSigned3 {
        transactionId paramTransactionId,
        rpmPackage {
         {
          continueOnFailure NULL,
          rpmCommandDetails contactPcmp : {
            iccid paramIccidValue
          }
         }
        }
      },
      smdpSignature3 paramSmdpSignature3
          }
```

Else

```
    loadRpmPkgReq LoadRpmPackageRequest ::= {
      smdpSigned3 {
        transactionId paramTransactionId,
        rpmPackage {
         {
          continueOnFailure NULL,
          rpmCommandDetails contactPcmp : {
             iccid paramIccidValue,
             dpiRpm paramDpiRpmValue
          }
         }
        }
      },
      smdpSignature3 paramSmdpSignature3
          }
```

End if

Method	MTD_REMOVE_NOTIF
Description	Constructs the command data for RemoveNotificationFromList
Parameter(s)	• paramSeqNumber: the sequence number to be removed

Details	`request NotificationSentRequest ::= {`
	`  seqNumber paramSeqNumber`
	`}`

Method	MTD_REQ_RPM_CMND
Description	This Method can be used to generate a RpmCommand for given paramComandDetailsChoice and continueOnFailure flag. - paramRpmComandList mentioned in MTD_LOAD_RPM_PKG_REQ_MULT_CMNDS can be generated
Parameter(s)	• paramComandDetailsChoice: The rpmCommandDetails choice (Mandatory) • paramIccidValue: The ICCID (Optional) • paramContinueOnFailureFlag: The Continue On Failure (Optional) • paramAdditional: The Additional parameters needed (Optional) If the paramComandDetailsChoice is listProfileInfo paramIccidValue is optional. If ContinueOnFailure shall be present set it as TRUE else it is not required. paramAdditional: If the paramComandDetailsChoice is updateMetadata UpdateMetadataRequest shall be included in paramAdditional. If the paramComandDetailsChoice is listProfileInfo and need TagList to be specified it shall be included in paramAdditional

Details	If paramComandDetailsChoice = enable or disable or delete or contactPcmp Then
	If paramContinueOnFailureFlag is present Then
	{
	continueOnFailure NULL,
	rpmCommandDetails paramComandDetailsChoice : {
	iccid paramIccidValue
	}
	}
	Else
	{
	rpmCommandDetails paramComandDetailsChoice : {
	iccid paramIccidValue
	}
	}
	Else if paramComandDetailsChoice = listProfileInfo and `paramAdditional` is not present Then
	If paramContinueOnFailureFlag is present Then
	{
	continueOnFailure NULL,
	rpmCommandDetails listProfileInfo : {
	searchCriteria iccid : { paramIccidValue }
	}
	}
	Else
	{
	rpmCommandDetails listProfileInfo : {
	searchCriteria iccid : { paramIccidvalue }
	}
	}
	Else if paramComandDetailsChoice = listProfileInfo and `paramAdditional` is present Then
	If paramContinueOnFailureFlag is present Then
	{
	continueOnFailure NULL,
	rpmCommandDetails listProfileInfo : {
	searchCriteria iccid : { paramIccidValue },
	tagList paramAdditional
	}
	}
	Else
	{
	rpmCommandDetails listProfileInfo : {
	searchCriteria iccid : { paramIccidvalue },
	tagList paramAdditional
	}
	}
	Else – for updateMetadata
	If paramContinueOnFailureFlag is present Then
	{
	continueOnFailure NULL,
	rpmCommandDetails updateMetadata : {
	iccid paramIccidvalue
	updateMetadataRequest {paramAdditional}
	}
	}
	Else
	{
	rpmCommandDetails updateMetadata : {

	iccid paramIccidValue
	updateMetadataRequest {paramAdditional}
	}
	}
	End if

Method	MTD_RES_RPR_CMND_RESULT
Description	This Method can be used to generate the SuccessResult list or ErrorResult list resulted from multiple RPM commands. - paramRpmCommandResultList mentioned in MTD_RES_RPR_FOR_MULT_CMNDS can be generated.
Parameter(s)	• paramRprErrorMask: The RPR error mask as defined below. (Mandatory) • paramRpmCommandResultDataChoice: The rpmCommandDetails choice (Optional) • paramIccidValue: The ICCID within the RPM Command received (Optional) • paramProfileInfoValue: The Profile Info response (Optional) • paramRpmCommandResultError: The Rpm Command Result error (Optional) paramRprErrorMask is defined as, 0 – OK result (RPM command successful) 1 - RpmCommandResultDataError is present 2 - RpmProcessingTerminated is present paramRpmCommandResultDataChoice is not required for error code RpmProcessingTerminated. Parameter paramIccidValue is mandatory if paramRpmCommandResultDataChoice is not listProfileInfoResult. Parameter paramProfileInfoValue is mandatory if paramRprErrorMask = 0 (the successful responses) and if paramRpmCommandResultDataChoice is listProfileInfoResult. Parameter paramRpmCommandResultError is mandatory if paramRprErrorMask > 0 (all error cases) and can be empty for paramRprErrorMask = 0

Details	If paramRprErrorMask = 0 Then -- *OK response*
	If paramRpmCommandResultDataChoice = enableResult or disableResult or deleteResult or updateMetadataResult Then
	{
	iccid paramIccidValue,
	rpmCommandResultData paramRpmCommandResultDataChoice : {
	paramRpmCommandResultDataChoice ok
	}
	}
	Else if paramRpmCommandResultDataChoice = listProfileInfoResult Then
	{
	rpmCommandResultData listProfileInfoResult : {
	profileInfoListOk : {paramProfileInfoValue}
	}
	Else if paramRpmCommandResultDataChoice = contactPcmpResult Then
	{

	iccid paramIccidValue,
	rpmCommandResultData contactPcmpResult : {
	contactPcmpResponseOk : {
	pcmpAddress #TEST_PCMP_ADDRESS1 }
	}
	}
	Else If paramRprErrorMask = 1 Then -- *rpmCommandResultDataError is present*
	{
	iccid paramIccidValue,
	rpmCommandResultData paramRpmCommandResultDataChoice : {
	paramRpmCommandResultDataChoice
	paramRpmCommandResultError
	}
	}
	Else If paramRprErrorMask = 2 Then -- *rpmProcessingTerminated is present*
	{
	rpmCommandResultData rpmProcessingTerminated : {
	paramRpmCommandResultError
	}
	}
	End if

Method	MTD_RES_RPR_FOR_MULT_CMNDS
Description	This Method can be used for verifying the rpmPackageResult resulted from multiple RPM commands.
Parameter(s)	• paramRpmCommandResultList The Rpm Command Result list for multiple RPM commands (Mandatory)

Details	response LoadRpmPackageResult ::=
	loadRpmPackageResultSigned : {
	loadRpmPackageResultDataSigned {
	transactionId <S_TRANSACTION_ID>,
	notificationMetadata #NOTIF_METADATA_RPR,
	smdpOid #S_SM_DP+_OID,
	finalResult rpmPackageExecutionResult : {
	paramRpmCommandResultList }
	}
	},
	euiccSignRPR <EUICC_SIGN_RPR>
	}

Method	MTD_RES_RPR_FOR_SINGLE_CMND
Description	This Method can be used for verifying the rpmPackageResult resulted from a single RPM command.
Parameter(s)	• paramRpmCommandResultDataChoice: The rpmCommandDetails choice • paramTransactionId: The Transaction Id (Mandatory) • paramIccidValue: The ICCID within the RPM Command received • paramRprErrorMask: The RPR error mask as defined below. (Mandatory) • paramNotificationMetadata: The notification Metadata.

	• paramSmdpOid: The SM-DP+ OID. • paramProfileInfoValue: The Profile Info response • paramPcmpAddressValue: The pcmp Address. • paramRpmCommandResultError: The Rpm Command Result error paramRprErrorMask is defined as, 0 – OK result (RPM command successful) 1 – RpmCommandResultDataError is present 2 – RpmProcessingTerminated is present 3 – LoadRpmPackageErrorCode is present 4 – LoadRpmPackageErrorCodeNotSigned is present Parameters paramRpmCommandResultDataChoice and paramIccidValue are mandatory for paramRprErrorMask 0 and 1 only. Parameters paramNotificationMetadata and paramSmdpOid are mandatory for all the cases except for paramRprErrorMask is 4 (loadRpmPackageErrorCodeNotSigned). Parameters paramProfileInfoValue and paramPcmpAddressValue shall be passed and can be empty depend on the paramRpmCommandResultDataChoice for paramRprErrorMask is 0 (the successful responses). Parameter paramRpmCommandResultError is mandatory if paramRprErrorMask > 0 (all error cases) and can be empty for paramRprErrorMask = 0.

Details	If paramRprErrorMask = 0 Then -- *OK response*
	If paramRpmCommandResultDataChoice = enableResult or disableResult or deleteResult Then
	`response LoadRpmPackageResult ::=`
	`loadRpmPackageResultSigned : {`
	`loadRpmPackageResultDataSigned {`
	`transactionId paramTransactionId,`
	`notificationMetadata paramNotificationMetadata,`
	`smdpOid paramSmdpOid,`
	`finalResult rpmPackageExecutionResult : {`
	`{`
	`iccid paramIccidValue,`
	`rpmCommandResultData paramRpmCommandResultDataChoice : {`
	`paramRpmCommandResultDataChoice ok`
	`}`
	`}`
	`}`
	`},`
	`euiccSignRPR <EUICC_SIGN_RPR>`
	`}`
	Else if paramRpmCommandResultDataChoice = updateMetadataResult Then
	`response LoadRpmPackageResult ::=`
	`loadRpmPackageResultSigned : {`
	`loadRpmPackageResultDataSigned {`
	`transactionId paramTransactionId,`
	`notificationMetadata paramNotificationMetadata,`
	`smdpOid paramSmdpOid,`
	`finalResult rpmPackageExecutionResult : {`
	`{`
	`iccid paramIccidValue,`

```
            rpmCommandResultData updateMetadataResult : ok
          }
        }
      },
      euiccSignRPR <EUICC_SIGN_RPR>
    }
```

Else if paramRpmCommandResultDataChoice = listProfileInfoResult Then

```
    response LoadRpmPackageResult ::=
    loadRpmPackageResultSigned : {
      loadRpmPackageResultDataSigned {
        transactionId paramTransactionId,
        notificationMetadata paramNotificationMetadata,
        smdpOid paramSmdpOid,
        finalResult rpmPackageExecutionResult : {
          {
            rpmCommandResultData listProfileInfoResult : {
              profileInfoListOk : {paramProfileInfoValue}
            }
          }
        }
      },
      euiccSignRPR <EUICC_SIGN_RPR>
    }
```

Else if paramRpmCommandResultDataChoice = contactPcmpResult Then

```
    response LoadRpmPackageResult ::=
    loadRpmPackageResultSigned : {
      loadRpmPackageResultDataSigned {
        transactionId paramTransactionId,
        notificationMetadata paramNotificationMetadata,
        smdpOid paramSmdpOid,
        finalResult rpmPackageExecutionResult : {
          {
            iccid paramIccidValue,
            rpmCommandResultData contactPcmpResult : {
              contactPcmpResponseOk : {
                pcmpAddress paramPcmpAddressValue }
            }
          }
        }
      },
      euiccSignRPR <EUICC_SIGN_RPR>
    }
```

Else If paramRprErrorMask = 1 Then -- *rpmCommandResultDataError is present*

If paramRpmCommandResultDataChoice = listProfileInfoResult Then

```
    response LoadRpmPackageResult ::=
    loadRpmPackageResultSigned : {
      loadRpmPackageResultDataSigned {
        transactionId paramTransactionId,
        notificationMetadata paramNotificationMetadata,
        smdpOid paramSmdpOid,
        finalResult rpmPackageExecutionResult : {
          {
            rpmCommandResultData listProfileInfoResult : {
              profileInfoListError : { paramRpmCommandResultError}
            }
          }
        }
```

```
    },
    euiccSignRPR <EUICC_SIGN_RPR>
  }
```

Else if paramRpmCommandResultDataChoice = contactPcmpResult Then

```
 response LoadRpmPackageResult ::=
 loadRpmPackageResultSigned : {
   loadRpmPackageResultDataSigned {
     transactionId paramTransactionId,
     notificationMetadata paramNotificationMetadata,
     smdpOid paramSmdpOid,
     finalResult rpmPackageExecutionResult : {
       {
          iccid paramIccidValue,
          rpmCommandResultData contactPcmpResult: {
             contactPcmpResponseError:
 { paramRpmCommandResultError}
          }
       }
     }
   },
   euiccSignRPR <EUICC_SIGN_RPR>
 }
```

Else if paramRpmCommandResultDataChoice = updateMetadataResult Then

```
 response LoadRpmPackageResult ::=
 loadRpmPackageResultSigned : {
   loadRpmPackageResultDataSigned {
     transactionId paramTransactionId,
     notificationMetadata paramNotificationMetadata,
     smdpOid paramSmdpOid,
     finalResult rpmPackageExecutionResult : {
       {
          iccid paramIccidValue,
          rpmCommandResultData updateMetadataResult:
                 paramRpmCommandResultError
       }
     }
   },
   euiccSignRPR <EUICC_SIGN_RPR>
 }
```

Else

```
response LoadRpmPackageResult ::=
loadRpmPackageResultSigned : {
  loadRpmPackageResultDataSigned {
    transactionId paramTransactionId,
    notificationMetadata paramNotificationMetadata,
    smdpOid paramSmdpOid,
    finalResult rpmPackageExecutionResult : {
      {
       iccid paramIccidValue,
       rpmCommandResultData paramRpmCommandResultDataChoice : {
          paramRpmCommandResultDataChoice
paramRpmCommandResultError
       }
    }
 }
           },
           euiccSignRPR <EUICC_SIGN_RPR>
       }
```

	Else If paramRprErrorMask = 2 Then -- *rpmProcessingTerminated is present*
	`response LoadRpmPackageResult ::=`
	`loadRpmPackageResultSigned : {`
	`loadRpmPackageResultDataSigned {`
	`transactionId paramTransactionId,`
	`notificationMetadata paramNotificationMetadata,`
	`smdpOid paramSmdpOid,`
	`finalResult rpmPackageExecutionResult : {`
	`{`
	`rpmCommandResultData rpmProcessingTerminated : {`
	`paramRpmCommandResultError`
	`}`
	`}`
	`}`
	`},`
	`euiccSignRPR <EUICC_SIGN_RPR>`
	`}`
	Else If paramRprErrorMask = 3 Then -- *loadRpmPackageErrorCode is present*
	`response LoadRpmPackageResult ::= {`
	`loadRpmPackageResultSigned : {`
	`loadRpmPackageResultDataSigned {`
	`transactionId paramTransactionId,`
	`notificationMetadata paramNotificationMetadata,`
	`smdpOid paramSmdpOid,`
	`finalResult loadRpmPackageErrorCode : {`
	`paramRpmCommandResultError }`
	`},`
	`euiccSignRPR <EUICC_SIGN_RPR>`
	`}`
	Else If paramRprErrorMask = 4 Then -- loadRpmPackageErrorCodeNotSigned is present
	`response LoadRpmPackageResult ::=`
	`loadRpmPackageResultNotSigned : {`
	`transactionId paramTransactionId,`
	`loadRpmPackageErrorCodeNotSigned paramRpmCommandResultError`
	`}`
	End if

Method	MTD_RETRIEVE_NOTIF_SEQ_NUM
Description	Constructs the command data for RetrieveNotificationsList filtered by sequence number
Parameter(s)	• paramSeqNumber: the sequence number to be retrieved

Details	`request RetrieveNotificationsListRequest ::= {`
	`searchCriteria seqNumber paramSeqNumber`
	`}`

Method	MTD_SELECT
Description	Generates the SELECT command as defined in GlobalPlatform Card Specification [6].
Parameter(s)	• paramAID: the AID to select

Details	`- CLA = 0x or 4x (x = <CHANNEL_NUMBER>)`
	`- INS = A4`
	`- P1 = 04`
	`- P2 = 00`

	- LC = <L> - paramAID - LE = 00

Method	MTD_SEND_SMS_PP
Description	Generate and send an envelope SMS-PP download to the MNO-SD
Parameter(s)	• paramApdusList: the list of APDUs (plain) to send
Details	Generate and send the following envelope: 80 C2 00 00 <L> D1 <L> 02 02 83 81 -- Device identity Tag 06 07 91 33 86 09 40 00 F0 -- Address Tag (TON/NPI/..) 0B <L> -- SMS TPDU 44 -- SMS-DELIVER 05 85 02 13 F2 -- TP-Originating-Address 7F -- TP-Protocol-Identifier F6 -- TP-Data-Coding-Scheme 71 30 12 41 55 74 40 -- TP-Service-Centre-Time-Stamp <L> -- TP-User-Data-Length 02 -- User-Data-Header-Length 70 -- IEIa 00 -- IEIDLa <L> -- Command Packet Length (2 bytes) <L> -- Command Header Length (1 byte) 12 21 -- SPI 00 -- KIC 15 -- KID (SCP80 Keyset version 0x01 in Triple DES) B2 01 00 -- MNO-SD TAR <MNO_SCP80_COUNTER> 00 -- Padding Counter <CC> -- Cryptographic checksum <C_APDUS_SCRIPT> -- Command APDUs script <C_APDUS_SCRIPT> SHALL contain the paramApdusList (i.e. each APDU is named <APDU1>; <APDU2>; …; <APDUn> here after) formatted as an expanded structure with definite length as defined in ETSI TS 102 226 [14]: AA <L> 22 <L> <APDU1> 22 <L> <APDU2> ... 22 <L> <APDUn> The Cryptographic checksum <CC> SHALL be generated in Triple DES (outer-CBC mode using two different keys) with the #MNO_SCP80_AUTH_KEY as defined in ETSI TS 102 225 [13]. If the command packet length is higher than 140 bytes, it SHALL be sent over several envelopes: SMS concatenation as defined in 3GPP TS 23.040 [22] SHALL be used.

Method	MTD_STORE_DATA
Description	Generates the STORE DATA command (Case 4) as defined in GlobalPlatform Card Specification [6].

Parameter(s)	• paramCommandData: the command data
Details	- CLA = 8x or Cx (x = <CHANNEL_NUMBER>) - INS = E2 - P1 = 91 - P2 = 00 - LC = <L> - paramCommandData - LE = 00

Method	MTD_STORE_DATA_Case3
Description	Generates the STORE DATA command (Case3) as defined in GlobalPlatform Card Specification [6].
Parameter(s)	• paramCommandData: the command data
Details	- CLA = 8x or Cx (x = <CHANNEL_NUMBER>) - INS = E2 - P1 = 90 - P2 = 00 - LC = <L> - paramCommandData

Method	MTD_STORE_DATA_SCRIPT
Description	Generate (multiple) STORE DATA command(s) by breaking the data into smaller components (if needed) for transmission.
Parameter(s)	• paramTLVDataToTransmit: TLVs array or single TLV to transfer to the eUICC • paramCase4Command (optional parameter, default value = TRUE): TRUE if the APDU is a Case 4 command, FALSE if the APDU is a Case 3 command

Details	For each element of paramTLVDataToTransmit
	If the size of the element is greater than 255 bytes, split the element in several blocks of 255 bytes. The last block MAY be shorter. Each block is named <DATA_SUB_PART> here after.
	
	If the element is up to 255 bytes, <DATA_SUB_PART> contains the value of the element.
	
	The bit b1 of P1 in the STORE DATA commands is named <B1_P1> here after and is defined as below:
	If paramCase4Command = TRUE Then
	<B1_P1> = 1
	Else
	<B1_P1> = 0
	End If
	
	Set <STORE_DATA_BLOCK_NUM> to 0
	For each <DATA_SUB_PART>
	If <DATA_SUB_PART> is an intermediate part, generate the following STORE DATA:
	- CLA = 8x or Cx (x = <CHANNEL_NUMBER>)
	- INS = E2

	- P1 = 1x (x = <B1_P1>) - P2 = <STORE_DATA_BLOCK_NUM> - LC = <L> - <DATA_SUB_PART> - LE = 00 -- present only if paramCase4Command = TRUE If <DATA_SUB_PART> is the last part, generate the following STORE DATA: - CLA = 8x or Cx (x = <CHANNEL_NUMBER>) - INS = E2 - P1 = 9x (x = <B1_P1>) - P2 = <STORE_DATA_BLOCK_NUM> - LC = <L> - <DATA_SUB_PART> - LE = 00 -- present only if paramCase4Command = TRUE Increase the <STORE_DATA_BLOCK_NUM> by 1 End End

C.2 Procedures

	Procedure	PROC_EUICC_CONFIGURE_LSIS_FOR_MEP
	Description	Configures LSIs to support MEP with the MEP mode and number of LSIs as needed by the test sequence.
	Parameters	• paramHighestLsi: The highest LSI proposed by the scenario. • paramLsiOptions: LSI options supported by the scenario. • paramMepModePriority: MEP mode priority order supported by the scenario. • paramMaxLsiNum: The maximum number of LSIs for Enabled Profiles paramMepModePriority: up to 3 bytes to represent the priority order. e.g. '010203' indicates priority order as 'MEP-A1, MEP-A2, MEP-B'. '0301' indicates priority order as 'MEP-B, MEP-A1', and not supporting MEP-A2. '01' indicates S_Device supports only MEP-A1, not supporting MEP-A2 or MEP-B (Forcing eUICC to select MEP-A1 in case it supports any other MEP mode as well in addition to MEP-A1).

Step	Direction	Sequence / Description	Expected result
1	S_Device → eUICC	send 'LSI support' in PPS2	
2	S_Device → eUICC	send MANAGE_LSI(Configure LSI) with the following tags: • 'Highest LSI proposed by the terminal' set to paramHighestLsi. • 'LSI options supported by the terminal' set to paramLsiOptions. • 'MEP mode(s) supported by the	SW=9000 Parse response and : • If 'LSI Options' is present Then ○ extract 'LSI Options' as <MEP_LSI_OPTIONS> • If 'LSI Options' is not present Then

		S_Device in the order of priority' set to paramMepModePriority • 'Maximum number of LSIs for Enabled Profiles of the S_Device' set to paramMaxLsiNum;	○ <MEP_LSI_OPTIONS> is 0x00 and • extract 'Jointly supported MEP mode' as <MEP_MODE> • extract 'Jointly supported maximum number of LSIs for Enabled Profiles' as <MEP_MAX_LSIS>

	Procedure	PROC_EUICC_INITIALIZATION_SEQUENCE	
	Description	Initialize communication between the S_Device and the eUICC.	
Step	**Direction**	**Sequence / Description**	**Expected result**
1	S_Device → eUICC	RESET	ATR present
2	S_Device → eUICC	[SELECT_MF]	FCP Template present SW=0x9000
3	S_Device → eUICC	[TERMINAL_CAPABILITY_LPAd]	SW=0x9000
4	S_Device → eUICC	[TERMINAL_PROFILE]	Toolkit initialization THEN SW=0x9000

	Procedure	PROC_EUICC_INITIALIZATION_SEQUENCE_Enterprise	
	Description	Initialize communication between the S_Device and the eUICC.	
Step	**Direction**	**Sequence / Description**	**Expected result**
1	S_Device → eUICC	RESET	ATR present
2	S_Device → eUICC	[SELECT_MF]	FCP Template present SW=0x9000
3	S_Device → eUICC	[TERMINAL_CAPABILITY_Enterprise]	SW=0x9000
4	S_Device → eUICC	[TERMINAL_PROFILE]	Toolkit initialization THEN SW=0x9000

	Procedure	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged	
	Description	Initialize communication between the S_Device and the eUICC.	
Step	**Direction**	**Sequence / Description**	**Expected result**
1	S_Device → eUICC	RESET	ATR returned by eUICC
2	S_Device → eUICC	[SELECT_MF]	FCP Template present SW=0x9000
3	S_Device → eUICC	[TERMINAL_CAPABILITY_LPAd]	SW=0x9000
4	S_Device → eUICC	[TERMINAL_PROFILE_eUICCProfileStateChanged]	Toolkit initialization THEN SW=0x9000

	Procedure	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP	
	Description	Initialize communication between the S_Device and the MEP capable eUICC after configuring 3 LSIs using the MANGE LSI (Configure LSI) command. It performs MEP eUICC initialization procedure for 3 LSIs configured. Expects TCs to have the steps for RESET and the PROC_EUICC_CONFIGURE_LSIS_FOR_MEP before calling this PROC.	
Step	**Direction**	**Sequence / Description**	**Expected result**
1	S_Device	PROC_MEP_LSI_MULTIPLEXING (0)	
2	S_Device → eUICC	[SELECT_MF]	FCP Template present SW=0x9000
3	S_Device → eUICC	[TERMINAL_CAPABILITY_LPAd]	SW=0x9000
4	S_Device → eUICC	[TERMINAL_PROFILE_LSI_COMMAND_eUICCProfileStateChanged]	Toolkit initialization THEN SW=0x9000
5	S_Device	PROC_MEP_LSI_MULTIPLEXING (1)	
Repeat Step 2 to 4			
6	S_Device	PROC_MEP_LSI_MULTIPLEXING (0)	
IF (NOT(<MEP_MODE>=MEP-B))			
7	S_Device	PROC_MEP_LSI_MULTIPLEXING (2)	
8	Repeat Step 4 to 6		
9	S_Device	PROC_MEP_LSI_MULTIPLEXING (0)	

	Procedure	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP_EN_DS_FIRST_PROFILE	
	Description	Initialize communication between the S_Device and the MEP capable eUICC after enable or disable the profile operation for the 1st profile. NOTE: 1st profile means the profile assigned to LSI-1 for MEP-A1 or MEP-A2, or the profile assigned to LSI-0 for MEP-B	
Step	**Direction**	**Sequence / Description**	**Expected result**
IF (NOT(<MEP_MODE> = MEP-B))			
1	S_Device	PROC_MEP_LSI_MULTIPLEXING(1)	
ENDIF			
IF (<MEP_MODE> = MEP-B)			
2	S_Device	PROC_MEP_LSI_MULTIPLEXING (0)	

ENDIF			
3	S_Device → eUICC	[SELECT_MF]	FCP Template present SW=0x9000
4	S_Device → eUICC	[TERMINAL_CAPABILITY_LPAd]	SW=0x9000
5	S_Device → eUICC	[TERMINAL_PROFILE_LSI_COMMAND_eUICCProfileStateChanged]	Toolkit initialization THEN SW=0x9000

	Procedure	PROC_EUICC_INITIALIZATION_SEQUENCE_eUICCProfileStateChanged_MEP_EN_DS_SECOND_PROFILE		
	Description	Initialize communication between the S_Device and the eUICC after enable or disable the profile operation for the 2nd profile. NOTE: 2nd profile means the profile assigned to LSI-2 for MEP-A1 or MEP-A2, or the profile assigned to LSI-1 for MEP-B		
Step	**Direction**	**Sequence / Description**	**Expected result**	
IF (NOT(<MEP_MODE> = MEP-B))				
1	S_Device	PROC_MEP_LSI_MULTIPLEXING(2)		
ENDIF				
IF (<MEP_MODE> = MEP-B)				
2	S_Device	PROC_MEP_LSI_MULTIPLEXING(1)		
ENDIF				
3	S_Device → eUICC	[SELECT_MF]	FCP Template present SW=0x9000	
4	S_Device → eUICC	[TERMINAL_CAPABILITY_LPAd]	SW=0x9000	
5	S_Device → eUICC	[TERMINAL_PROFILE_LSI_COMMAND_eUICCProfileStateChanged]	Toolkit initialization THEN SW=0x9000	

	Procedure	PROC_EUICC_INITIALIZATION_SEQUENCE_LPA_Alerting	
	Description	Initialize communication between the S_Device and the eUICC.	
Step	**Direction**	**Sequence / Description**	**Expected result**
1	S_Device → eUICC	RESET	ATR present
2	S_Device → eUICC	[SELECT_MF]	FCP Template present SW=0x9000
3	S_Device → eUICC	[TERMINAL_CAPABILITY_LPA_Alerting]	SW=0x9000
4	S_Device → eUICC	[TERMINAL_PROFILE]	Toolkit initialization THEN SW=0x9000

	Procedure	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP	
	Description	Initialize communication between the S_Device and the MEP capable eUICC after using the MANAGE LSI(Configure LSI) command to configure enough LSIs to host two simultaneously Enabled Profiles. It performs MEP eUICC initialization procedure for all LSIs configured. Expects TCs to have the steps for RESET and the PROC_EUICC_CONFIGURE_LSIS_FOR_MEP before calling this PROC.	
Step	**Direction**	**Sequence / Description**	**Expected result**
1	S_Device	PROC_MEP_LSI_MULTIPLEXING(0)	
2	S_Device → eUICC	[SELECT_MF]	FCP Template present SW=0x9000
3	S_Device → eUICC	[TERMINAL_CAPABILITY_LPAd]	SW=0x9000
4	S_Device → eUICC	[TERMINAL_PROFILE_LSI_COMMAND]	Toolkit initialization THEN SW=0x9000
5	S_Device	PROC_MEP_LSI_MULTIPLEXING(1)	
6	Repeat steps 2-4		
IF (NOT(<MEP_MODE> = MEP-B))			
7	S_Device	PROC_MEP_LSI_MULTIPLEXING(2)	
8	Repeat steps 2-4		
ENDIF			
9	S_Device	PROC_MEP_LSI_MULTIPLEXING(0)	

	Procedure	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_FIRST_PROFILE	
	Description	Initialize communication between the S_Device and the MEP capable eUICC after enable or disable the profile operation for the 1st profile. NOTE: 1st profile means the profile assigned to LSI-1 for MEP-A1 or MEP-A2, or the profile assigned to LSI-0 for MEP-B	
Step	**Direction**	**Sequence / Description**	**Expected result**
IF (NOT(<MEP_MODE> = MEP-B))			
1	S_Device	PROC_MEP_LSI_MULTIPLEXING(1)	
ENDIF			
IF (<MEP_MODE> = MEP-B)			
2	S_Device	PROC_MEP_LSI_MULTIPLEXING(0)	
ENDIF			
3	S_Device → eUICC	[SELECT_MF]	FCP Template present SW=0x9000
4	S_Device → eUICC	[TERMINAL_CAPABILITY_LPAd]	SW=0x9000
5	S_Device → eUICC	[TERMINAL_PROFILE_LSI_COMMAND]	Toolkit initialization THEN SW=0x9000

	Procedure	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_EN_DS_SECOND_PROFILE	
	Description	Initialize communication between the S_Device and the eUICC after enable or disable the profile operation for the 2nd profile. NOTE: 2nd profile means the profile assigned to LSI-2 for MEP-A1 or MEP-A2, or the profile assigned to LSI-1 for MEP-B	
Step	**Direction**	**Sequence / Description**	**Expected result**
IF (NOT(<MEP_MODE> = MEP-B))			
1	S_Device	PROC_MEP_LSI_MULTIPLEXING(2)	
ENDIF			
IF (<MEP_MODE> = MEP-B)			
2	S_Device	PROC_MEP_LSI_MULTIPLEXING(1)	
ENDIF			
3	S_Device → eUICC	[SELECT_MF]	FCP Template present SW=0x9000
4	S_Device → eUICC	[TERMINAL_CAPABILITY_LPAd]	SW=0x9000
5	S_Device → eUICC	[TERMINAL_PROFILE_LSI_COMMAND]	Toolkit initialization THEN SW=0x9000

	Procedure	PROC_EUICC_INITIALIZATION_SEQUENCE_MEP_ONE_LSI_FOR_ENABLED_PROFILE	
	Description	Initialises communication between the S_Device and the MEP capable eUICC with only one LSI out of the two or more LSIs configured for Enabled Profiles by a previous MANAGE LSI(Configure LSI) command. Expects TCs to have the steps for RESET and the PROC_EUICC_CONFIGURE_LSIS_FOR_MEP before calling this PROC.	
Step	**Direction**	**Sequence / Description**	**Expected result**
1	S_Device	PROC_MEP_LSI_MULTIPLEXING(0)	
2	S_Device → eUICC	[SELECT_MF]	FCP Template present SW=0x9000
3	S_Device → eUICC	[TERMINAL_CAPABILITY_LPAd]	SW=0x9000
4	S_Device → eUICC	[TERMINAL_PROFILE_LSI_COMMAND]	Toolkit initialization THEN SW=0x9000
IF (NOT(<MEP_MODE> = MEP-B))			
5	S_Device	PROC_MEP_LSI_MULTIPLEXING(1)	
Repeat steps 2-4			
ENDIF			
6	S_Device	PROC_MEP_LSI_MULTIPLEXING(0)	

	Procedure	PROC_MEP_LSI_MULTIPLEXING	
	Description	Selects the LSI for the given eSIM Port number to be used for the next APDUs on an eUICC configured for MEP MEP-capable eUICC. The test tool SHALL use the LSI selection method indicated by the EUM in #IUT_EUICC_MULTIPLEXING_LSI_INDICATION. All subsequent APDUs are sent on the indicated paramPortForNextCommand until the next call to this procedure. This procedure does nothing if the current command Port is already the one specified in the parameter paramPortForNextCommand.	
	Parameters	paramCommandPort: LSI number on which to send the subsequent APDUs	
Step	**Direction**	**Sequence / Description**	**Expected result**
IF (#IUT_EUICC_MULTIPLEXING_LSI_INDICATION = 'MANAGE LSI(Select LSI)')			
1	S_Device → eUICC	Send 'MANAGE LSI(Select LSI) with the LSI Number set to paramPortForNextCommand	SW=9000
2	S_Device → eUICC	Send the subsequent commands specified by the Test Sequence unchanged, until the next call to this procedure.	
ENDIF			
IF (#IUT_EUICC_MULTIPLEXING_LSI_INDICATION = ' T=1 + NAD byte')			
3	S_Device → eUICC	Construct the transport for the subsequent commands specified by the Test Sequence by setting the LSI number paramPortForNextCommand in the NAD byte, until the next call to this procedure.	
ENDIF			

	Procedure	PROC_MEP_REFRESH_EN_DS	
	Description	Handles the REFRESH on an LSI and simulates the device processing it after enabling or disabling profiles.	
	Parameters	• paramTargetEsimPort: target eSIM port for ES10 calls (mandatory for MEP-A1) • paramRefreshMode: Refresh mode “eUICC Profile State Change" or “UICC Reset”(mandatory)	
Step	**Direction**	**Sequence / Description**	**Expected result**
IF (<MEP_MODE> = ‘MEP-A1’)			
NOTE: in this case the enable or disable command was sent on port 0, and the resulting SW was 91YY			
1	S_Device → eUICC	FETCH ‘YY’	LSI COMMAND Proactive Command Extract ‘Action’ as <LSI_COMMAND_ACTION> from the command qualifier Verify that <LSI_COMMAND_ACTION> is equal to “Proactive session request” Verify that ‘LSI numbers’ contains a single number and extract it as <LSI_NUMBER> Verify that <LSI_NUMBER> is equal to paramTargetEsimPort

2	S_Device → eUICC	[TERMINAL RESPONSE]	
3	PROC_MEP_LSI_MULTIPLEXING(paramTargetEsimPort)		
4	S_Device → eUICC	FETCH 'Le=00'	REFRESH Proactive Command Verify that the Refresh Mode in the Proactive Command is equal to paramRefreshMode
ENDIF			
NOTE: if <MEP_MODE> = 'MEP-B', the enable or disable command was sent on the target port, and the resulting SW was 91XX			
5	S_Device → eUICC	FETCH 'XX'	REFRESH Proactive Command Verify that the Refresh Mode in the Proactive Command is equal to paramRefreshMode
ENDIF			
IF (paramRefreshMode = "eUICC Profile State Change")			
6	S_Device → eUICC	[TERMINAL RESPONSE]	
ENDIF			
IF (paramRefreshMode = "UICC Reset")			
7	S_Device → eUICC	MANAGE_LSI(Reset LSE, paramTargetEsimPort)	
ENDIF			

	Procedure	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR
	Description	The LPAd opens a logical channel and selects the ISD-R. Since the S_LPAd may have to do that on more than one LSI, the S_LPAd SHALL record one value of the <CHANNEL NUMBER> per LSI.

Step	Direction	Sequence / Description	Expected result
1	S_LPAd → eUICC	[MANAGE_CHANNEL_OPEN]	Extract the <CHANNEL_NUMBER> from response data SW=0x9000
2	S_LPAd → eUICC	MTD_SELECT(#ISD_R_AID)	SW=0x9000

	Procedure	PROC_VERIFY_SESSION_IS_CANCELLED
	Description	Verify that the RSP session identified by the TransactionID <S_TRANSACTION_ID> has been cancelled by the eUICC (i.e. Common Mutual Authentication and Profile Download procedures SHALL be rejected as long as no GetEUICCChallenge has been requested).

Step	Direction	Sequence / Description	Expected result

1	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(#PREP_DOWNLOAD_NO_CC)	#R_PREP_DOWN_NO_SESSION
			SW=0x9000
			The transactionId returned in the response SHALL not be checked (any value SHALL be accepted).

3	S_LPAd → eUICC	MTD_STORE_DATA_SCRIPT(
		MTD_AUTHENTICATE_SMDP(
		#TEST_DP_ADDRESS1,
		<S_SMDP_CHALLENGE>,
		#CTX_PARAMS1,
		<S_SMDP_SIGNATURE1>,
		#CERT_S_SM_DPauth_SIG,
		NO_PARAM,
		#CRL_LIST,
		FALSE
		)
		)
			#R_AUTH_SERVER_NO_SESSION
			SW = 0x9000
			The transactionId returned in the response SHALL not be checked (any value SHALL be accepted).

Annex D Commands And Responses

D.1 ES8+ Requests And Responses

D.1.1 ES8+ Requests

Name	Content
CONF_ISDP_EMPTY	req ConfigureISDPRequest ::={}

CONF_ISDP_MAX_LENGTH	req ConfigureISDPRequest ::={
	dpProprietaryData { -- *size=128 bytes*
	dpOid #S_SM_DP+_OID,
	additionalSmdpData
	#ADDITIONAL_SMDP_DATA_MAX_LENGTH
	}
	}
	-- NOTE: Instead of
	DpProprietaryData ::= SEQUENCE {
	dpOid OBJECT IDENTIFIER
	-- additional data objects defined by the
	-- SM-DP+ MAY follow
	}
	
	-- the following structure is used to test the
	-- DpProprietaryData size:
	DpProprietaryData ::= SEQUENCE {
	dpOid OBJECT IDENTIFIER,
	additionalSmdpData OCTET STRING OPTIONAL
	}

CONF_ISDP_PROF1	req ConfigureISDPRequest ::={
	dpProprietaryData {
	dpOid #S_SM_DP+_OID
	}
	}

CONF_ISDP_PROF2	req ConfigureISDPRequest ::={
	dpProprietaryData {
	dpOid #S_SM_DP+_OID
	}
	}

CONF_ISDP_SIZE_EXCEEDED	req ConfigureISDPRequest ::={
	dpProprietaryData { -- *size=129 bytes*
	dpOid #S_SM_DP+_OID,
	additionalSmdpData
	#ADDITIONAL_SMDP_DATA_EXCEEDED_MAX
	}
	}
	
	-- NOTE: Instead of
	DpProprietaryData ::= SEQUENCE {
	dpOid OBJECT IDENTIFIER
	-- additional data objects defined by the
	-- SM-DP+ MAY follow
	}
	
	-- the following structure is used to test the
	-- DpProprietaryData size:
	DpProprietaryData ::= SEQUENCE {
	dpOid OBJECT IDENTIFIER,

Name	Content
	additionalSmdpData OCTET STRING OPTIONAL }

FULL_METADATA	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	notificationConfigurationInfo {
	{ profileManagementOperation {
	notificationInstall,
	notificationLocalEnable,
	notificationLocalDisable,
	notificationLocalDelete
	},
	notificationAddress #TEST_DP_ADDRESS1
	}
	},
	profileOwner {
	mccMnc #MCC_MNC1
	},
	profilePolicyRules {ppr1}
	}

INIT_SC_INVALID_CRT	req InitialiseSecureChannelRequest ::={
	remoteOpId #REMOTE_OP_ID_INSTALL,
	transactionId <S_TRANSACTION_ID>,
	controlRefTemplate {
	keyType #INVALID_KEY_TYPE,
	keyLen #KEY_LENGTH,
	hostId #HOST_ID
	},
	smdpOtpk <OTPK_S_SM_DP+_ECKA>,
	smdpSign <S_SM_DP+_SIGN>
	}

INIT_SC_INVALID_OP_ID	req InitialiseSecureChannelRequest ::={
	remoteOpId #INVALID_REMOTE_OP_ID,
	transactionId <S_TRANSACTION_ID>,
	controlRefTemplate {
	keyType #KEY_TYPE,
	keyLen #KEY_LENGTH,
	hostId #HOST_ID
	},
	smdpOtpk <OTPK_S_SM_DP+_ECKA>,
	smdpSign <S_SM_DP+_SIGN>
	}

INIT_SC_INVALID_SIGN	req InitialiseSecureChannelRequest ::={
	remoteOpId #REMOTE_OP_ID_INSTALL,
	transactionId <S_TRANSACTION_ID>,
	controlRefTemplate {
	keyType #KEY_TYPE,
	keyLen #KEY_LENGTH,
	hostId #HOST_ID
	},
	smdpOtpk <OTPK_S_SM_DP+_ECKA>,
	smdpSign <S_SM_DP+_SIGN>
	}

Name	Content
	The <S_SM_DP+_SIGN> SHALL NOT be computed using the #SK_S_SM_DPpb_SIG but SHALL have the same length as for a valid signature.

INIT_SC_INVALID_TRANS_ID	req InitialiseSecureChannelRequest ::={
	remoteOpId #REMOTE_OP_ID_INSTALL,
	transactionId <INVALID_TRANSACTION_ID>,
	controlRefTemplate {
	keyType #KEY_TYPE,
	keyLen #KEY_LENGTH,
	hostId #HOST_ID
	},
	smdpOtpk <OTPK_S_SM_DP+_ECKA>,
	smdpSign <S_SM_DP+_SIGN>
	}

METADATA_ICCID_MISMATCH	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF2,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1
	}

METADATA_MCCMNC_MISMATCH	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	profileOwner {
	mccMnc #MCC_MNC2
	},
	profilePolicyRules {ppr2}
	}

METADATA_NO_CLASS	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	notificationConfigurationInfo {
	{ profileManagementOperation {
	notificationInstall,
	notificationLocalEnable,
	notificationLocalDisable,
	notificationLocalDelete
	},
	notificationAddress #TEST_DP_ADDRESS1
	}
	}
	}

METADATA_OP_PROF1	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	notificationConfigurationInfo {
	{ profileManagementOperation {
	notificationInstall,
	notificationLocalEnable,
	notificationLocalDisable,
	notificationLocalDelete
	},

Name	Content

	notificationAddress #TEST_DP_ADDRESS1
	}
	},
	profileOwner {
	mccMnc #MCC_MNC1
	}
	}

METADATA_OP_PROF1_RPM_CONF_EN_OTHER_ENTERPRISE_PROF_UM_ENT_CONF	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable},
	tagList '99BD'H -- Tag for PPR,
	EnterpriseConfig
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	profileOwnerOid #S_PROFILE_OWNER_OID
	},
	enterpriseConfiguration {
	enterpriseOid #S_ENTERPRISE_OID,
	enterpriseName #ENTERPRISE_NAME1,
	}
	}
	}

Name	Content

METADATA_OP_PROF1_RPM_CONF_ENTERPRISE_REF_RULE3	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable},
	taglist 'BD'H
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	profileOwnerOid #S_PROFILE_OWNER_OID
	},
	enterpriseConfiguration {
	enterpriseOid #S_ENTERPRISE_OID,
	enterpriseName #ENTERPRISE_NAME1,
	enterpriseRules {
	enterpriseRuleBits {
	referenceEnterpriseRule,
	onlyEnterpriseProfilesCanBeInstalled
	},
	numberOfNonEnterpriseProfiles 0
	}
	}
	}

METADATA_OP_PROF1_RPM_CONF_NO_ENTERPRISE_CONF_TAGLIST_ENT_CONF	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable},
	tagList 'BD'H -- Tag for EnterpriseConfig
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	profileOwnerOid #S_PROFILE_OWNER_OID
	}
	}

Name	Content

METADATA_OP_PROF1_RPM_CONF_REF_PROF_ONLY_THIS_EN	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable}
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	profileOwnerOid #S_PROFILE_OWNER_OID
	},
	enterpriseConfiguration {
	enterpriseOid #S_ENTERPRISE_OID,
	enterpriseName #ENTERPRISE_NAME1,
	enterpriseRules {
	enterpriseRuleBits {
	referenceEnterpriseRule,
	priorityEnterpriseProfile,
	onlyEnterpriseProfilesCanBeInstalled
	},
	numberOfNonEnterpriseProfiles 0
	}
	}
	}

METADATA_OP_PROF2_RPM_CONF_ALL_ENTERP_RULES	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF2,
	serviceProviderName #SP_NAME2,
	profileName #NAME_OP_PROF2,
	iconType png,
	icon #ICON_OP_PROF2,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC2
	},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable, disable, delete, listProfileInfo, contactPcmp}
	tagList '99BA9BBCBD9F1F'H -- Tags for rpmConfiguration, hriServerAddress, lprConfiguration, EnterpriseConfiguration, ServiceDescription
	}
	},
	profileOwnerOid #S_PROFILE_OWNER_OID
	},
	enterpriseConfiguration {

Name	Content

	enterpriseOid #S_ENTERPRISE_OID,
	enterpriseName #ENTERPRISE_NAME1,
	enterpriseRules {
	enterpriseRuleBits {
	referenceEnterpriseRule,
	priorityEnterpriseProfile,
	onlyEnterpriseProfilesCanBeInstalled
	},
	numberOfNonEnterpriseProfiles 0
	}
	},
	serviceDescription {voice, data}
	}

METADATA_OP_PROF2_RPM_CONF_ALL_NO_ENTERP_CONF	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF2,
	serviceProviderName #SP_NAME2,
	profileName #NAME_OP_PROF2,
	iconType png,
	icon #ICON_OP_PROF2,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC2
	},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable, disable, delete, listProfileInfo, contactPcmp}
	tagList '99BA9BBC9F1F'H -- Tags for rpmConfiguration, hriServerAddress, lprConfiguration, ServiceDescription
	}
	},
	profileOwnerOid #S_PROFILE_OWNER_OID
	},
	serviceDescription {voice, data}
	}

Name	Content

METADATA_OP_PROF2_RPM_CONF_ALLOW_NON_ENTERPRISE_PROF_INS	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF2,
	serviceProviderName #SP_NAME2,
	profileName #NAME_OP_PROF2,
	iconType png,
	icon #ICON_OP_PROF2,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC2
	},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable}
	}
	},
	profileOwnerOid #S_PROFILE_OWNER_OID
	},
	enterpriseConfiguration {
	enterpriseOid #S_ENTERPRISE_OID,
	enterpriseName #ENTERPRISE_NAME1,
	enterpriseRules {
	enterpriseRuleBits {
	referenceEnterpriseRule,
	priorityEnterpriseProfile,
	},
	numberOfNonEnterpriseProfiles 0
	}
	}
	}

Name	Content

METADATA_OP_PROF2_RPM_CONF_EN_ONLY_THIS_REF_ENTERPRISE_PROF	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF2,
	serviceProviderName #SP_NAME2,
	profileName #NAME_OP_PROF2,
	iconType png,
	icon #ICON_OP_PROF2,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC2
	},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable}
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	profileOwnerOid #S_PROFILE_OWNER_OID
	},
	enterpriseConfiguration {
	enterpriseOid #S_ENTERPRISE_OID,
	enterpriseName #ENTERPRISE_NAME1,
	enterpriseRules {
	enterpriseRuleBits {
	referenceEnterpriseRule,
	priorityEnterpriseProfile,
	onlyEnterpriseProfilesCanBeInstalled,
	},
	numberOfNonEnterpriseProfiles 0
	}
	}
	}

METADATA_OP_PROF2_RPM_CONF_EN_PPR	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF2,
	serviceProviderName #SP_NAME2,
	profileName #NAME_OP_PROF2,
	iconType png,
	icon #ICON_OP_PROF2,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC2
	},
	profilePolicyRules {ppr1, ppr2},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable}
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	profileOwnerOid #S_PROFILE_OWNER_OID
	}
	}

METADATA_OP_PROF4	metadataReq StoreMetadataRequest ::= { iccid #ICCID_OP_PROF4,

Name	Content

	serviceProviderName #SP_NAME4,
	profileName #NAME_OP_PROF4,
	iconType png,
	icon #ICON_OP_PROF4,
	profileClass operational,
	notificationConfigurationInfo {
	{ profileManagementOperation {
	notificationInstall,
	notificationLocalEnable,
	notificationLocalDisable,
	notificationLocalDelete
	},
	notificationAddress #TEST_DP_ADDRESS4
	}
	},
	profileOwner {
	mccMnc #MCC_MNC4
	},
	profilePolicyRules {
	ppr1
	}
	}

METADATA_OP_PROF9	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF9,
	serviceProviderName #SP_NAME9,
	profileName #NAME_OP_PROF9,
	profileOwner {
	mccMnc #MCC_MNC9,
	gid1 #GID1,
	gid2 #GID2
	},
	profilePolicyRules {
	ppr2
	}
	}

METADATA_OP_PROF10	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF10,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF5,
	iconType png,
	icon #ICON_OP_PROF5,
	profileClass operational,
	notificationConfigurationInfo {
	{ profileManagementOperation {
	notificationInstall,
	notificationLocalEnable,
	notificationLocalDisable,
	notificationLocalDelete
	},
	notificationAddress #TEST_DP_ADDRESS1
	}
	},
	profileOwner {
	mccMnc #MCC_MNC2
	}
	}

Name	Content

METADATA_OP_PROF10_NO_PROFILE_OWNER	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF10,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF5,
	iconType png,
	icon #ICON_OP_PROF5,
	profileClass operational,
	notificationConfigurationInfo {
	{ profileManagementOperation {
	notificationInstall,
	notificationLocalEnable,
	notificationLocalDisable,
	notificationLocalDelete
	},
	notificationAddress #TEST_DP_ADDRESS1
	}
	}
	}

METADATA_OP1_GID1GID2_PRESENT	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	profileOwner {
	mccMnc #MCC_MNC1,
	gid1 #GID1,
	gid2 #GID2
	},
	profilePolicyRules {ppr2}
	}

METADATA_OP9_GID1GID2_MISSING	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF9,
	serviceProviderName #SP_NAME9,
	profileName #NAME_OP_PROF9,
	profileOwner {
	mccMnc #MCC_MNC9
	}
	}

METADATA_PPR_NO_OWNER	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	profilePolicyRules {ppr2}
	}

Name	Content

METADATA_SERVICE_SPECIFIC_STORED	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	notificationConfigurationInfo {
	{ profileManagementOperation {
	notificationInstall,
	notificationLocalEnable,
	notificationLocalDisable,
	notificationLocalDelete
	},
	notificationAddress #TEST_DP_ADDRESS1
	}
	},
	profileOwner {
	mccMnc #MCC_MNC1
	},
	serviceSpecificDataStoredInEuicc
	#VENDOR_SPECIFIC_EXTENSION1
	}

METADATA_SERVICE_SPECIFIC_NOT_STORED	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	notificationConfigurationInfo {
	{ profileManagementOperation {
	notificationInstall,
	notificationLocalEnable,
	notificationLocalDisable,
	notificationLocalDelete
	},
	notificationAddress #TEST_DP_ADDRESS1
	}
	},
	profileOwner {
	mccMnc #MCC_MNC1
	},
	serviceSpecificDataNotStoredInEuicc
	#VENDOR_SPECIFIC_EXTENSION2
	}

Name	Content

METADATA_SERVICE_SPECIFIC_STORED_AND_NOT_STORED	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	notificationConfigurationInfo {
	{ profileManagementOperation {
	notificationInstall,
	notificationLocalEnable,
	notificationLocalDisable,
	notificationLocalDelete
	},
	notificationAddress #TEST_DP_ADDRESS1
	}
	},
	profileOwner {
	mccMnc #MCC_MNC1
	},
	serviceSpecificDataStoredInEuicc #VENDOR_SPECIFIC_EXTENSION1,
	serviceSpecificDataNotStoredInEuicc #VENDOR_SPECIFIC_EXTENSION2
	}
	}

METADATA_WILDCARD	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	profileOwner {
	mccMnc #MCC_MNC_WILDCARD
	},
	profilePolicyRules {ppr2}
	}

METADATA_WITH_EC	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	enterpriseConfiguration {
	enterpriseOid #ENTERPRISE_OID
	enterpriseName #ENTERPRISE_NAME1
	}
	}

METADATA_WITH_EC_OID2	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF2,
	serviceProviderName #SP_NAME2,
	profileName #NAME_OP_PROF2,
	profileOwner {
	mccMnc #MCC_MNC2
	},

Name	Content

	enterpriseConfiguration {
	enterpriseOid #S_ENTERPRISE_OID2
	enterpriseName #ENTERPRISE_NAME1
	}
	}

METADATA_WITH_EC_PROF2	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF2,
	serviceProviderName #SP_NAME2,
	profileName #NAME_OP_PROF2,
	profileOwner {
	mccMnc #MCC_MNC2
	},
	enterpriseConfiguration {
	enterpriseOid #S_ENTERPRISE_OID
	enterpriseName #ENTERPRISE_NAME1
	}
	}

METADATA_WITH_ER	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	enterpriseConfiguration {
	enterpriseOid #S_ENTERPRISE_OID
	enterpriseName #ENTERPRISE_NAME1
	enterpriseRules {
	enterpriseRuleBits {
	priorityEnterpriseProfile,
	onlyEnterpriseProfilesCanBeInstalled
	},
	numberOfNonEnterpriseProfiles 0
	}
	}
	}

METADATA_WITH_JPG	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType jpg,
	icon #ICON_JPG
	}

METADATA_WITH_NOTIFS	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	notificationConfigurationInfo {
	{ profileManagementOperation {
	notificationInstall
	},
	notificationAddress #TEST_DP_ADDRESS3
	},
	{ profileManagementOperation {

Name	Content

	notificationInstall
	},
	notificationAddress #TEST_DP_ADDRESS2
	},
	{ profileManagementOperation {
	notificationLocalEnable
	},
	notificationAddress #TEST_DP_ADDRESS2
	},
	{ profileManagementOperation {
	notificationLocalEnable
	},
	notificationAddress #TEST_DP_ADDRESS3
	},
	{ profileManagementOperation {
	notificationLocalDisable
	},
	notificationAddress #TEST_DP_ADDRESS3
	},
	{ profileManagementOperation {
	notificationLocalDisable
	},
	notificationAddress #TEST_DP_ADDRESS4
	},
	{ profileManagementOperation {
	notificationLocalDelete
	},
	notificationAddress #TEST_DP_ADDRESS1
	},
	{ profileManagementOperation {
	notificationLocalDelete
	},
	notificationAddress #TEST_DP_ADDRESS3
	}
	}
	}

METADATA_WITH_PPR1_PPR2	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	profilePolicyRules {ppr1,ppr2}
	}

METADATA_WITH_PPR2	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	profilePolicyRules {ppr2}
	}

METADATA_WITH_RER	metadataReq StoreMetadataRequest ::= { iccid #ICCID_OP_PROF1, serviceProviderName #SP_NAME1,

Name	Content

	profileName #NAME_OP_PROF1,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	enterpriseConfiguration {
	enterpriseOid #S_ENTERPRISE_OID
	enterpriseName #ENTERPRISE_NAME1
	enterpriseRules {
	enterpriseRuleBits {
	referenceEnterpriseRule,
	priorityEnterpriseProfile,
	onlyEnterpriseProfilesCanBeInstalled
	},
	numberOfNonEnterpriseProfiles 0
	}
	}
	}

METADATA_WITH_RER_PROF2	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF2,
	serviceProviderName #SP_NAME2,
	profileName #NAME_OP_PROF2,
	profileOwner {
	mccMnc #MCC_MNC2
	},
	enterpriseConfiguration {
	enterpriseOid #ENTERPRISE_OID
	enterpriseName #ENTERPRISE_NAME1
	enterpriseRules {
	enterpriseRuleBits {
	referenceEnterpriseRule,
	onlyEnterpriseProfilesCanBeInstalled
	},
	numberOfNonEnterpriseProfiles 0
	}
	}
	}

METADATA_WITHOUT_ICON	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType jpg
	}

REPLACE_S_KEYS_REQ	req ReplaceSessionKeysRequest ::={
	initialMacChainingValue <PPK_INIT_MAC>,
	ppkEnc <PPK_ENC>,
	ppkCmac <PPK_MAC>
	}

REPLACE_S_KEYS_REQ_INV_SIZE	req ReplaceSessionKeysRequest ::={
	initialMacChainingValue
	#PPK_INIT_MAC_INV_SIZE,
	ppkEnc #PPK_ENC_INV_SIZE,
	ppkCmac #PPK_MAC_INV_SIZE
	}

S_INIT_SC_PROF1	req InitialiseSecureChannelRequest ::={ remoteOpId #REMOTE_OP_ID_INSTALL,

Name	Content

	transactionId <S_TRANSACTION_ID>,
	controlRefTemplate {
	keyType #KEY_TYPE,
	keyLen #KEY_LENGTH,
	hostId #HOST_ID
	},
	smdpOtpk <OTPK_S_SM_DP+_ECKA>,
	smdpSign <S_SM_DP+_SIGN>
	}

S_INIT_SC_PROF2	req InitialiseSecureChannelRequest ::={
	remoteOpId #REMOTE_OP_ID_INSTALL,
	transactionId <S_TRANSACTION_ID>,
	controlRefTemplate {
	keyType #KEY_TYPE,
	keyLen #KEY_LENGTH,
	hostId #HOST_ID
	},
	smdpOtpk <OTPK_S_SM_DP+_ECKA>,
	smdpSign <S_SM_DP+_SIGN>
	}

D.2 VOID

D.3 ES10x Requests And Responses

D.3.1 ES10x Requests

Name	Content

AUTHENTICATE_SMDP_WITH_DEVICE_INFO_NAI	req AuthenticateServerRequest ::= {
	serverSigned1 {
	transactionId <S_TRANSACTION_ID>,
	euiccChallenge <EUICC_CHALLENGE>,
	serverAddress #TEST_DP_ADDRESS1,
	serverChallenge <S_SMDP_CHALLENGE>
	},
	serverSignature1 <S_SMDP_SIGNATURE1>,
	euiccCiPKIdToBeUsed
	<EUICC_CI_PK_ID_TO_BE_USED>,
	serverCertificate #CERT_S_SM_DPauth_ECDSA,
	ctxParams1 #CTX_PARAMS1_DEVICE_INFO_NAI
	}

CANCEL_SESSION_INV_TRANS_ID	req CancelSessionRequest ::={
	transactionId <INVALID_TRANSACTION_ID>,
	reason endUserRejection
	}

CANCEL_SESSION_LOAD_BPP	req CancelSessionRequest ::={
	transactionId <S_TRANSACTION_ID>,
	reason loadBppExecutionError
	}

CANCEL_SESSION_METADATA	req CancelSessionRequest ::={
	transactionId <S_TRANSACTION_ID>,
	reason metadataMismatch
	}

CANCEL_SESSION_POSTPONED	req CancelSessionRequest ::={ transactionId <S_TRANSACTION_ID>, reason postponed

Name	Content
	}

CANCEL_SESSION_PPR	req CancelSessionRequest ::={
	  transactionId <S_TRANSACTION_ID>,
	  reason pprNotAllowed
	}

CANCEL_SESSION_REJECT	req CancelSessionRequest ::={
	  transactionId <S_TRANSACTION_ID>,
	  reason endUserRejection
	}

CANCEL_SESSION_TIMEOUT	req CancelSessionRequest ::={
	  transactionId <S_TRANSACTION_ID>,
	  reason timeout
	}

CANCEL_SESSION_UNDEF	req CancelSessionRequest ::={
	  transactionId <S_TRANSACTION_ID>,
	  reason undefinedReason
	}

ENTERPRISE_CONFIG1	{
	  enterpriseOid #S_ENTERPRISE_OID,
	  enterpriseName #ENTERPRISE_NAME1,
	  enterpriseRules {
	  enterpriseRuleBits {
	  referenceEnterpriseRule,
	  priorityEnterpriseProfile,
	  onlyEnterpriseProfilesCanBeInstalled
	  },
	  numberOfNonEnterpriseProfiles 0
	  }
	}

ENTERPRISE_CONFIG1_EC	{
	  enterpriseOid #S_ENTERPRISE_OID,
	  enterpriseName #ENTERPRISE_NAME1,
	}

ENTERPRISE_CONFIG1_ER	{
	  enterpriseOid #S_ENTERPRISE_OID,
	  enterpriseName #ENTERPRISE_NAME1,
	  enterpriseRules {
	  enterpriseRuleBits {
	  priorityEnterpriseProfile,
	  onlyEnterpriseProfilesCanBeInstalled
	  },
	  numberOfNonEnterpriseProfiles 0
	  }
	}

Name	Content

ENTERPRISE_CONFIG3	{
	enterpriseOid #ENTERPRISE_OID
	enterpriseName #ENTERPRISE_NAME1
	enterpriseRules {
	enterpriseRuleBits {
	referenceEnterpriseRule,
	priorityEnterpriseProfile,
	onlyEnterpriseProfilesCanBeInstalled
	},
	numberOfNonEnterpriseProfiles 0
	}
	}

ENTERPRISE_CONFIG4	{
	enterpriseOid #ENTERPRISE_OID
	enterpriseName #ENTERPRISE_NAME1
	enterpriseRules {
	enterpriseRuleBits {
	onlyEnterpriseProfilesCanBeInstalled
	},
	numberOfNonEnterpriseProfiles 0
	}
	}

ENTERPRISE_CONFIG5	{
	enterpriseOid #S_ENTERPRISE_OID,
	enterpriseName #ENTERPRISE_NAME1,
	}

EUICC_MEMORY_RESET	req EuiccMemoryResetRequest ::= {
	resetOptions {
	deleteOperationalProfiles,
	resetDefaultSmdpAddress
	}
	}

EUICC_MEMORY_RESET_DEF_SMDPADDRESS	req EuiccMemoryResetRequest ::= {
	resetOptions { resetDefaultSmdpAddress }
	}

EUICC_MEMORY_RESET_OP_PRO	req EuiccMemoryResetRequest ::= {
	resetOptions { deleteOperationalProfiles }
	}

GET_CONF_OP_PROF1	opConfProf1Req ProfileInfoListRequest ::= {
	searchCriteria iccid: #ICCID_OP_PROF1,
	tagList '4FB8'H
	}

GET_EID	getEIDReq GetEuiccDataRequest ::= {
	tagList '5A'H
	}

GET_EID_INVALID	getEIDReq GetEuiccDataRequest ::= {
	tagList '6B'H
	}

GET_ENTERPRISE_CONFIG_OP_PROF1	opConfProf1Req ProfileInfoListRequest ::= {
	searchCriteria iccid: #ICCID_OP_PROF1,
	tagList '5ABD'H
	}

GET_ENTERPRISE_CONFIG_OP_PROF2	opConfProf1Req ProfileInfoListRequest ::= { searchCriteria iccid: #ICCID_OP_PROF2,

Name	Content
	tagList '5ABD'H }
GET_EUICC_CHALLENGE	request GetEuiccChallengeRequest ::= {}
GET_EUICC_CONFIGURED_DATA	request EuiccConfiguredDataRequest ::={}
GET_EUICC_INFO1	request GetEuiccInfo1Request::= { }
GET_EUICC_INFO2	request GetEuiccInfo2Request::= { }

GET_HRI_SRV_ADDRESS_OP_PROF1	opConfProf1Req ProfileInfoListRequest ::= {
	searchCriteria iccid: #ICCID_OP_PROF1,
	tagList '5A9B'H
	}

GET_LPR_CONFIG_OP_PROF1	opConfProf1Req ProfileInfoListRequest ::= {
	searchCriteria iccid: #ICCID_OP_PROF1,
	tagList '5ABC'H
	}

GET_METADATA_OP_PROF1	opConfProf1Req ProfileInfoListRequest ::= {
	searchCriteria iccid: #ICCID_OP_PROF1,
	tagList '5A9192939495B6B799'H
	}

GET_METADATA_OP_PROF1_SERVICE_SPECIFIC	opConfProf1Req ProfileInfoListRequest ::= {
	searchCriteria iccid: #ICCID_OP_PROF1,
	tagList '5A9192939495B6B799BF22'H
	}

GET_MULTIPLE_TAGS_OP_PROF1	opConfProf1Req ProfileInfoListRequest ::= {
	searchCriteria iccid: #ICCID_OP_PROF1,
	tagList 'BA9BBC'H
	}

GET_NEW_METADATA_V3	getupdate1Req ProfileInfoListRequest ::= {
	searchCriteria iccid: #ICCID_OP_PROF1,
	tagList '9192939499B6BF22BA9BBCBF20'H -- *names, icon, PPRs, notif config, specific data, RPM configuration, HRI Server Address, LPR configuration and DC configuration*
	}

GET_NEW_METADATA_NOTIF	getupdate1Req ProfileInfoListRequest ::= {
	searchCriteria iccid: #ICCID_OP_PROF1,
	tagList '91929394B699'H -- *names, icon, notif config and PPRs*
	}

GET_NEW_METADATA_SPEC_DATA	getupdate1Req ProfileInfoListRequest ::= {
	searchCriteria iccid: #ICCID_OP_PROF1,
	tagList '91929394BF2299'H -- *names, icon, specific data and PPRs*
	}

GET_NEW_METADATA_RPM_CONFIG	getupdate1Req ProfileInfoListRequest ::= {
	searchCriteria iccid: #ICCID_OP_PROF1,
	tagList '91929394BA99'H -- *names, icon, RPM configuration and PPRs*
	}

GET_NEW_METADATA_HRI_ADDR	getupdate1Req ProfileInfoListRequest ::= {
	searchCriteria iccid: #ICCID_OP_PROF1,
	tagList '919293949B99'H -- *names, icon, HRI Server Address and PPRs*
	}

GET_NEW_METADATA_LPR_CONFIG	getupdate1Req ProfileInfoListRequest ::= {

Name	Content
	searchCriteria iccid: #ICCID_OP_PROF1, tagList '91929394BC99'H -- *names, icon, LPR configuration and PPRs* }

GET_NEW_METADATA_DC_CONFIG	getupdate1Req ProfileInfoListRequest ::= {
	searchCriteria iccid: #ICCID_OP_PROF1,
	tagList '91929394BF2099'H -- *names, icon, DC configuration and PPRs*
	}

GET_NOTIF_CONF_OP_PROF1	opConfProf1Req ProfileInfoListRequest ::= {
	searchCriteria iccid: #ICCID_OP_PROF1,
	tagList '5AB6'H
	}

GET_PPR_OP_PROF1	opConfProf1Req ProfileInfoListRequest ::= {
	searchCriteria iccid: #ICCID_OP_PROF1,
	tagList '5A99'H
	}

GET_PROFILES_INFO_ALL	request ProfileInfoListRequest::= { }

GET_PROFILES_INFO_ICCID_TAGLIST 1	request ProfileInfoListRequest::= {
	searchCriteria iccid: #ICCID_OP_PROF1,
	tagList '9F70'H --*state*
	}

GET_PROFILES_INFO_ICCID_TAGLIST 2	request ProfileInfoListRequest::= {
	searchCriteria iccid: #ICCID_OP_PROF1,
	tagList '93'H --*icon type*
	}

GET_PROFILES_INFO_ICCID_TAGLIST 3	request ProfileInfoListRequest::= {
	searchCriteria iccid: #ICCID_OP_PROF1,
	tagList '95'H --*Profile Class*
	}

GET_PROFILES_INFO_ICCID_TAGLIST 4	request ProfileInfoListRequest::= {
	searchCriteria iccid: #ICCID_OP_PROF1,
	tagList 'B6'H --*Notification configuration*
	}

GET_PROFILES_INFO_ICCID_TAGLIST 5	request ProfileInfoListRequest::= {
	searchCriteria iccid: #ICCID_OP_PROF3,
	tagList '99'H --*ppr*
	}

GET_PROFILES_INFO_OPTAGLIST1	request ProfileInfoListRequest::= {
	searchCriteria profileClass: operational,
	tagList '5A9F70'H -- *ICCID and State*
	}

GET_PROFILES_INFO_OPTAGLIST2	request ProfileInfoListRequest::= {
	searchCriteria profileClass: operational,
	tagList '909F70'H --*Nickname and State*
	}

GET_PROFILES_INFO_OPTAGLIST3	request ProfileInfoListRequest::= {
	searchCriteria profileClass: operational,
	tagList '9493'H --*Icon, Icon type*
	}

GET_PROFILES_INFO_OPTAGLIST4	request ProfileInfoListRequest::= {
	searchCriteria profileClass: operational,
	tagList '949F70'H --*Icon, state*
	}

GET_PROFILES_INFO_PROFCLASS	request ProfileInfoListRequest::= {

Name	Content
	searchCriteria profileClass: operational }

GET_PROFILES_INFO_TAGLIST_ICCID	request ProfileInfoListRequest::= {
	tagList '5A'H
	}

GET_PROFILES_INFO_TAGLIST_ICON	request ProfileInfoListRequest::= {
	tagList '94'H
	}

GET_PROFILES_INFO_TAGLIST_ISDPAID	request ProfileInfoListRequest::= {
	tagList '4F'H
	}

GET_PROFILES_INFO_TAGLIST_PROFILE_NAME	request ProfileInfoListRequest::= {
	tagList '92'H
	}

GET_PROFILES_INFO_TAGLIST_PROFILE_NICKNAME	request ProfileInfoListRequest::= {
	tagList '90'H
	}

GET_PROFILES_INFO_TAGLIST_PROFILE_OWNER	request ProfileInfoListRequest::= {
	tagList 'B7'H
	}

GET_PROFILES_INFO_TAGLIST_SMDP_PROP_DATA	request ProfileInfoListRequest::= {
	tagList 'B8'H
	}

GET_PROFILES_INFO_TAGLIST_SP_NAME	request ProfileInfoListRequest::= {
	tagList '91'H
	}

GET_PROFILES_INFO_TAGLIST1	request ProfileInfoListRequest::= {
	tagList '5A9F70'H -- *ICCID and State*
	}

GET_PROFILES_INFO_TAGLIST2	request ProfileInfoListRequest::= {
	tagList '909F70'H --*Nickname and State*
	}

GET_PROFILES_INFO_TAGLIST3	request ProfileInfoListRequest::= {
	tagList '9493'H --*Icon, Icon type*
	}

GET_PROFILES_INFO_TAGLIST4	request ProfileInfoListRequest::= {
	tagList '949F70'H --*Icon, state*
	}

GET_PROFILES_OWNERS	request ProfileInfoListRequest::= {
	tagList 'B7'H
	}

GET_RAT	request GetRatRequest ::={}

GET_RPM_CONFIG_OP_PROF1	opConfProf1Req ProfileInfoListRequest ::= {
	searchCriteria iccid: #ICCID_OP_PROF1,
	tagList '5ABA'H
	}

LIST_NOTIF_ALL	request ListNotificationRequest ::= {
	profileManagementOperation {
	notificationInstall,
	notificationLocalEnable,
	notificationLocalDisable,
	notificationLocalDelete
	}
	}

LIST_NOTIF_ALL_RPM	request ListNotificationRequest ::= { profileManagementOperation {

Name	Content
	notificationRpmEnable, notificationRpmDisable, notificationRpmDelete, loadRpmPackageResult } }

LIST_NOTIF_DELETE	request ListNotificationRequest ::= {
	profileManagementOperation {
	notificationLocalDelete
	}
	}

LIST_NOTIF_DELETE_ENABLE_RPM	request ListNotificationRequest ::= {
	profileManagementOperation {
	notificationRpmDelete,
	notificationRpmEnable
	}
	}

LIST_NOTIF_DELETE_RPM	request ListNotificationRequest ::= {
	profileManagementOperation {
	notificationRpmDelete
	}
	}

LIST_NOTIF_DISABLE	request ListNotificationRequest ::= {
	profileManagementOperation {
	notificationLocalDelete
	}
	}

LIST_NOTIF_DISABLE_DELETE	request ListNotificationRequest ::= {
	profileManagementOperation {
	notificationLocalDisable,
	notificationLocalDelete
	}
	}

LIST_NOTIF_DISABLE_DELETE_RPM	request ListNotificationRequest ::= {
	profileManagementOperation {
	notificationRpmDisable,
	notificationRpmDelete
	}
	}

LIST_NOTIF_DISABLE_ENABLE	request ListNotificationRequest ::= {
	profileManagementOperation {
	notificationLocalDisable,
	notificationLocalEnable
	}
	}

LIST_NOTIF_DISABLE_ENABLE_RPM	request ListNotificationRequest ::= {
	profileManagementOperation {
	notificationRpmDisable,
	notificationRpmEnable
	}
	}

LIST_NOTIF_DISABLE_RPM	request ListNotificationRequest ::= {
	profileManagementOperation {
	notificationRpmDisable
	}
	}

Name	Content

LIST_NOTIF_ENABLE	request ListNotificationRequest ::= {
	profileManagementOperation {
	notificationLocalEnable
	}
	}

LIST_NOTIF_ENABLE_DISABLE_DELETE	request ListNotificationRequest ::= {
	profileManagementOperation {
	notificationLocalEnable,
	notificationLocalDisable,
	notificationLocalDelete
	}
	}

LIST_NOTIF_ENABLE_RPM	request ListNotificationRequest ::= {
	profileManagementOperation {
	notificationRpmEnable
	}
	}

LIST_NOTIF_INSTALL	request ListNotificationRequest ::= {
	profileManagementOperation {
	notificationInstall
	}
	}

LIST_NOTIF_INSTALL_ENABLE	request ListNotificationRequest ::= {
	profileManagementOperation {
	notificationInstall,
	notificationLocalEnable
	}
	}

LIST_NOTIF_INSTALL_ENABLE_DISABLE	request ListNotificationRequest ::= {
	profileManagementOperation {
	notificationInstall,
	notificationLocalEnable,
	notificationLocalDisable
	}
	}

LIST_NOTIF_NONE	request ListNotificationRequest ::= {
	profileManagementOperation {}
	}

LIST_NOTIF_OMITTED	request ListNotificationRequest ::= {}

LIST_NOTIF_RPR_DELETE_ENABLE_RPM	request ListNotificationRequest ::= {
	profileManagementOperation {
	loadRpmPackageResult,
	notificationRpmDelete,
	notificationRpmEnable
	}
	}

LIST_NOTIF_RPR_DELETE_RPM	request ListNotificationRequest ::= {
	profileManagementOperation {
	loadRpmPackageResult,
	notificationRpmDelete
	}
	}

LIST_NOTIF_RPR_DISABLE_DELETE_RPM	request ListNotificationRequest ::= {
	profileManagementOperation {
	loadRpmPackageResult,
	notificationRpmDisable,

Name	Content
	notificationRpmDelete } }

LIST_NOTIF_RPR_DISABLE_ENABLE_RPM	request ListNotificationRequest ::= {
	profileManagementOperation {
	loadRpmPackageResult,
	notificationRpmDisable,
	notificationRpmEnable
	}
	}

LIST_NOTIF_RPR_DISABLE_RPM	request ListNotificationRequest ::= {
	profileManagementOperation {
	loadRpmPackageResult,
	notificationRpmDisable
	}
	}

LIST_NOTIF_RPR_ENABLE_RPM	request ListNotificationRequest ::= {
	profileManagementOperation {
	loadRpmPackageResult,
	notificationRpmEnable
	}
	}

METADATA_EN_DI_DE_NOTIFS	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	notificationConfigurationInfo {
	{ profileManagementOperation {
	notificationLocalEnable,
	notificationLocalDisable,
	notificationLocalDelete
	},
	notificationAddress #TEST_DP_ADDRESS1
	},
	{ profileManagementOperation {
	notificationLocalEnable,
	notificationLocalDisablee,
	notificationLocalDelete
	},
	notificationAddress #TEST_DP_ADDRESS2
	}
	}
	}

METADATA_OP_PROF1_INST_DIFF	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	notificationConfigurationInfo {
	{ profileManagementOperation {
	notificationInstall
	},
	notificationAddress #TEST_DP_ADDRESS2
	}
	},

Name	Content

	profileOwner {
	mccMnc #MCC_MNC1
	}
	}

METADATA_OP_PROF1_NO_INSTALL	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	notificationConfigurationInfo {
	{
	profileManagementOperation {
	notificationLocalEnable,
	notificationLocalDisable,
	notificationLocalDelete
	},
	notificationAddress #TEST_DP_ADDRESS1
	}
	},
	profileOwner {
	mccMnc #MCC_MNC1
	}
	}

METADATA_OP_PROF1_RPM_CONF	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	profileClass operational,
	notificationConfigurationInfo {
	{
	profileManagementOperation {
	notificationRpmEnable,
	notificationRpmDisable,
	notificationRpmDelete
	},
	notificationAddress #TEST_DP_ADDRESS1
	}
	},
	profileOwner {
	mccMnc #MCC_MNC1
	},
	rpmConfiguration {
	managingDpList {
	{
	managingDP #S_SM_DP+_OID,
	rpmType { enable, disable, delete,
	listProfileInfo, contactPcmp
	}
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	profileOwnerOid #S_PROFILE_OWNER_OID
	}
	}

Name	Content

METADATA_OP_PROF1_RPM_CONF_ALL	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable, disable, delete, listProfileInfo, contactPcmp}
	tagList '99BA9BBC'H -- Tag for PPR, rpmConfiguration, hriServerAddress, lprConfiguration
	}
	},
	profileOwnerOid #S_PROFILE_OWNER_OID
	},
	lprConfiguration{ # TEST_PCMP_ADDRESS1 }
	}

METADATA_OP_PROF1_RPM_CONF_ALL_CI_PKI_RAND	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	rpmConfiguration {
	managingDpList {
	{
	managingDP #S_SM_DP+_OID,
	rpmType { enable, disable, delete, listProfileInfo, contactPcmp}
	tagList '99BA9BBC'H -- Tags for PPR, rpmConfiguration, hriServerAddress, lprConfiguration
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	allowedCiPKId <CI_PKI_RANDOM>,
	profileOwnerOid #S_PROFILE_OWNER_OID
	}
	}

METADATA_OP_PROF1_RPM_CONF_ALL_DP_OID2	metadataReq StoreMetadataRequest ::= { iccid #ICCID_OP_PROF1, serviceProviderName #SP_NAME1, profileName #NAME_OP_PROF1, iconType png, icon #ICON_OP_PROF1,

Name	Content

	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	profilePolicyRules {ppr1},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID2,
	rpmType {enable, disable, delete, listProfileInfo, contactPcmp}
	tagList '99BA9BBC'H -- Tags for PPR, rpmConfiguration, hriServerAddress, lprConfiguration
	}
	},
	profileOwnerOid #S_PROFILE_OWNER_OID
	},
	hriServerAddress { #TEST_HRI_ADDRESS1 }
	}

METADATA_OP_PROF1_RPM_CONF_ALL_PPR1	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	profilePolicyRules {ppr1},
	hriServerAddress { #TEST_HRI_ADDRESS1 },
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable, disable, delete, listProfileInfo, contactPcmp}
	tagList '99BA9BBC'H -- Tag for *PPR*, rpmConfiguration, hriServerAddress, lprConfiguration
	}
	},
	profileOwnerOid #S_PROFILE_OWNER_OID
	}
	}

Name	Content

METADATA_OP_PROF1_RPM_CONF_ALL_LPR_CONF_NOT_ALLOWED	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	profilePolicyRules {ppr1},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable, disable, delete, listProfileInfo, contactPcmp}
	tagList '99BA9B'H -- Tag for PPR, rpmConfiguration and hriServerAddress
	}
	},
	profileOwnerOid #S_PROFILE_OWNER_OID
	},
	hriServerAddress { #TEST_HRI_ADDRESS1 }
	}

METADATA_OP_PROF1_RPM_CONF_CP_CI_PKI_RAND	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	rpmConfiguration {
	managingDpList {
	{
	managingDP #S_SM_DP+_OID,
	rpmType {contactPcmp},
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	allowedCiPKId <CI_PKI_RANDOM>,
	profileOwnerOid #S_PROFILE_OWNER_OID
	},
	lprConfiguration {
	pcmpAddress #TEST_PCMP_ADDRESS1
	}
	}

METADATA_OP_PROF1_RPM_CONF_DE	metadataReq StoreMetadataRequest ::= { iccid #ICCID_OP_PROF1, serviceProviderName #SP_NAME1, profileName #NAME_OP_PROF1, iconType png, icon #ICON_OP_PROF1, profileClass operational,

Name	Content

	profileOwner {
	mccMnc #MCC_MNC1
	},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {delete}
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	profileOwnerOid #S_PROFILE_OWNER_OID
	}
	}

METADATA_OP_PROF1_RPM_CONF_DE_CI_PKI_RAND	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	rpmConfiguration {
	managingDpList {
	{
	managingDP #S_SM_DP+_OID,
	rpmType {delete},
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	allowedCiPKId <CI_PKI_RANDOM>,
	profileOwnerOid #S_PROFILE_OWNER_OID
	}
	}

METADATA_OP_PROF1_RPM_CONF_DE_DP_OID2	metadataReq StoreMetadataRequest:: = {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID2,
	rpmType {delete}
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	profileOwnerOid #S_PROFILE_OWNER_OID
	}
	}

Name	Content

METADATA_OP_PROF1_RPM_CONF_DE_PPR2	metadataReq StoreMetadataRequest:: = {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	profilePolicyRules {ppr2},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {delete}
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	profileOwnerOid #S_PROFILE_OWNER_OID
	}
	}

METADATA_OP_PROF1_RPM_CONF_DI	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {disable}
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	profileOwnerOid #S_PROFILE_OWNER_OID
	}
	}

METADATA_OP_PROF1_RPM_CONF_DI_CI_PKI_RAND	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	rpmConfiguration {
	managingDpList {
	{
	managingDP #S_SM_DP+_OID,
	rpmType {disable},
	}

Name	Content
	}, pollingAddress #TEST_DP_ADDRESS1, allowedCiPKId #<CI_PKI_RANDOM>, profileOwnerOid #S_PROFILE_OWNER_OID } }

METADATA_OP_PROF1_RPM_CONF_DI_DP_OID2	metadataReq StoreMetadataRequest:: = {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID2,
	rpmType {disable}
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	profileOwnerOid #S_PROFILE_OWNER_OID
	}
	}

METADATA_OP_PROF1_RPM_CONF_DI_PPR1	metadataReq StoreMetadataRequest:: = {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	profilePolicyRules {ppr1},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {disable}
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	profileOwnerOid #S_PROFILE_OWNER_OID
	}
	}

METADATA_OP_PROF1_RPM_CONF_EN	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC1
	},

Name	Content

	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable}
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	profileOwnerOid #S_PROFILE_OWNER_OID
	}
	}

METADATA_OP_PROF1_RPM_CONF_EN_CI_PKI_RAND	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	rpmConfiguration {
	managingDpList {
	{
	managingDP #S_SM_DP+_OID,
	rpmType {enable},
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	allowedCiPKId <CI_PKI_RANDOM>,
	profileOwnerOid #S_PROFILE_OWNER_OID
	}
	}

METADATA_OP_PROF1_RPM_CONF_EN_DP_OID2	metadataReq StoreMetadataRequest:: = {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID2,
	rpmType {enable}
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	profileOwnerOid #S_PROFILE_OWNER_OID
	}
	}

Name	Content

METADATA_OP_PROF1_RPM_CONF_EN_ONLY_THIS_REF_ENTERPRISE_PROF	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable}
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	profileOwnerOid #S_PROFILE_OWNER_OID
	},
	enterpriseConfiguration {
	enterpriseOid #S_ENTERPRISE_OID,
	enterpriseName #ENTERPRISE_NAME1,
	enterpriseRules {
	enterpriseRuleBits {
	referenceEnterpriseRule,
	priorityEnterpriseProfile,
	onlyEnterpriseProfilesCanBeInstalled
	},
	numberOfNonEnterpriseProfiles 0
	}
	}
	}

METADATA_OP_PROF1_RPM_CONF_EN_OTHER_ENTERPRISE_PROF	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable},
	tagList 'BD'H
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	profileOwnerOid #S_PROFILE_OWNER_OID
	},
	enterpriseConfiguration {
	enterpriseOid #S_ENTERPRISE_OID,
	enterpriseName #ENTERPRISE_NAME1,
	enterpriseRules {

Name	Content
	enterpriseRuleBits {}, numberOfNonEnterpriseProfiles 0 } } }

METADATA_OP_PROF1_RPM_CONF_EN_REF_ENTERPRISE_PROF	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable}
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	profileOwnerOid #S_PROFILE_OWNER_OID
	},
	enterpriseConfiguration {
	enterpriseOid #S_ENTERPRISE_OID,
	enterpriseName #ENTERPRISE_NAME1,
	enterpriseRules {
	enterpriseRuleBits {
	referenceEnterpriseRule,
	priorityEnterpriseProfile,
	onlyEnterpriseProfilesCanBeInstalled
	},
	numberOfNonEnterpriseProfiles 0
	}
	}
	}

METADATA_OP_PROF1_RPM_CONF_NO_PCMP_ADDR	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {contactPcmp}
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	profileOwnerOid #S_PROFILE_OWNER_OID
	}
	}

Name	Content

METADATA_OP_PROF1_RPM_CONF_OSN_DP2	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	profileClass operational,
	notificationConfigurationInfo {
	{
	profileManagementOperation {
	notificationRpmEnable,
	notificationRpmDisable,
	notificationRpmDelete
	},
	notificationAddress #TEST_DP_ADDRESS2
	}
	},
	profileOwner {
	mccMnc #MCC_MNC1
	},
	rpmConfiguration {
	managingDpList {
	{
	managingDP #S_SM_DP+_OID,
	rpmType { enable, disable, delete,
	listProfileInfo, contactPcmp
	}
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	profileOwnerOid #S_PROFILE_OWNER_OID
	}
	}

METADATA_OP_PROF1_RPM_CONF_PCMP	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {contactPcmp}
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	profileOwnerOid #S_PROFILE_OWNER_OID
	},
	lprConfiguration {
	pcmpAddress #TEST_PCMP_ADDRESS1
	}
	}

METADATA_OP_PROF1_RPM_CONF_UM_CI_PKI_RAND	metadataReq StoreMetadataRequest ::= { iccid #ICCID_OP_PROF1, serviceProviderName #SP_NAME1,

Name	Content

	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	rpmConfiguration {
	managingDpList {
	{
	managingDP #S_SM_DP+_OID,
	tagList '99BA9BBC'H -- Tag for PPR, rpmConfiguration, hriServerAddress, lprConfiguration
	}
	},
	allowedCiPKId #<CI_PKI_RANDOM>,
	profileOwnerOid #S_PROFILE_OWNER_OID
	}
	}

METADATA_OP_PROF1_RPM_CONF_UM_PPR_CTRL_BIT	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	profilePolicyRules {
	pprUpdateControl,ppr1,ppr2 },
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	tagList '99'H -- Tag for PPR
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	profileOwnerOid #S_PROFILE_OWNER_OID
	}
	}

METADATA_OP_PROF1_RPM_CONF_UPDATE_MD_PPR	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	profilePolicyRules {ppr1, ppr2},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	tagList '99'H -- Tag for PPR

Name	Content
	} }, pollingAddress #TEST_DP_ADDRESS1, profileOwnerOid #S_PROFILE_OWNER_OID } }

METADATA_OP_PROF2	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF2,
	serviceProviderName #SP_NAME2,
	profileName #NAME_OP_PROF2,
	iconType png,
	icon #ICON_OP_PROF2,
	profileClass operational,
	notificationConfigurationInfo {
	{ profileManagementOperation {
	notificationInstall,
	notificationLocalEnable,
	notificationLocalDisable,
	notificationLocalDelete
	},
	notificationAddress #TEST_DP_ADDRESS2
	}
	},
	profileOwner {
	mccMnc #MCC_MNC2
	}
	}

METADATA_OP_PROF2_NO_INSTALL	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF2,
	serviceProviderName #SP_NAME2,
	profileName #NAME_OP_PROF2,
	iconType png,
	icon #ICON_OP_PROF2,
	profileClass operational,
	notificationConfigurationInfo {
	{
	profileManagementOperation {
	notificationLocalEnable,
	notificationLocalDisable,
	notificationLocalDelete
	},
	notificationAddress #TEST_DP_ADDRESS2
	}
	},
	profileOwner {
	mccMnc #MCC_MNC2
	}
	}

METADATA_OP_PROF2_RPM_CONF_ALL	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF2,
	serviceProviderName #SP_NAME2,
	profileName #NAME_OP_PROF2,
	iconType png,
	icon #ICON_OP_PROF2,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC2

Name	Content

	},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable, disable, delete, listProfileInfo, contactPcmp}
	tagList '99BA9BBCBD9F1F'H -- Tags for rpmConfiguration, hriServerAddress, lprConfiguration, EnterpriseConfiguration, ServiceDescription
	}
	},
	profileOwnerOid #S_PROFILE_OWNER_OID
	},
	enterpriseConfiguration {
	enterpriseOid #S_ENTERPRISE_OID,
	enterpriseName #ENTERPRISE_NAME1,
	enterpriseRules {
	enterpriseRuleBits {
	referenceEnterpriseRule,
	priorityEnterpriseProfile,
	onlyEnterpriseProfilesCanBeInstalled
	},
	numberOfNonEnterpriseProfiles 0
	}
	},
	serviceDescription {voice, data}
	}

METADATA_OP_PROF2_RPM_CONF_ALL_OWNER2	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF2,
	serviceProviderName #SP_NAME2,
	profileName #NAME_OP_PROF2,
	iconType png,
	icon #ICON_OP_PROF2,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC2
	},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable, disable, delete, listProfileInfo, contactPcmp}
	tagList '99BA9BBC9F1F'H -- Tags for PPR, rpmConfiguration, hriServerAddress, lprConfiguration, ServiceDescription
	}
	},
	profileOwnerOid #S_PROFILE_OWNER_OID2
	},
	serviceDescription {voice, data}
	}

METADATA_OP_PROF2_RPM_CONF_EN	metadataReq StoreMetadataRequest ::= { iccid #ICCID_OP_PROF2, serviceProviderName #SP_NAME2, profileName #NAME_OP_PROF2, iconType png,

Name	Content

	icon #ICON_OP_PROF2,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC2
	},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable}
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	profileOwnerOid #S_PROFILE_OWNER_OID
	}
	}

METADATA_OP_PROF2_RPM_CONF_EN_OWNER_OID1	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF2,
	serviceProviderName #SP_NAME2,
	profileName #NAME_OP_PROF2,
	iconType png,
	icon #ICON_OP_PROF2,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC2
	},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable}
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	profileOwnerOid #S_PROFILE_OWNER_OID
	}
	}

METADATA_OP_PROF2_RPM_CONF_EN_OWNER_OID1_PPR1	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF2,
	serviceProviderName #SP_NAME2,
	profileName #NAME_OP_PROF2,
	iconType png,
	icon #ICON_OP_PROF2,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC2
	},
	profilePolicyRules {ppr1},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable}
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	profileOwnerOid #S_PROFILE_OWNER_OID
	}
	}

Name	Content

METADATA_OP_PROF2_RPM_CONF_EN_OWNER_OID2	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF2,
	serviceProviderName #SP_NAME2,
	profileName #NAME_OP_PROF2,
	iconType png,
	icon #ICON_OP_PROF2,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC2
	},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable}
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	profileOwnerOid #S_PROFILE_OWNER_OID2
	}
	}

METADATA_OP_PROF2_RPM_CONF_EN_REF_ENTERPRISE_PROF	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF2,
	serviceProviderName #SP_NAME2,
	profileName #NAME_OP_PROF2,
	iconType png,
	icon #ICON_OP_PROF2,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC2
	},
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable}
	}
	},
	pollingAddress #TEST_DP_ADDRESS1,
	profileOwnerOid #S_PROFILE_OWNER_OID
	},
	enterpriseConfiguration {
	enterpriseOid #S_ENTERPRISE_OID,
	enterpriseName #ENTERPRISE_NAME1,
	enterpriseRules {
	enterpriseRuleBits {
	referenceEnterpriseRule,
	priorityEnterpriseProfile,
	onlyEnterpriseProfilesCanBeInstalled
	},
	numberOfNonEnterpriseProfiles 0
	}
	}
	}

METADATA_OP_PROF3	metadataReq StoreMetadataRequest ::= { iccid #ICCID_OP_PROF3, serviceProviderName #SP_NAME3, profileName #NAME_OP_PROF3, iconType png,

Name	Content

	icon #ICON_OP_PROF3,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC2
	},
	profilePolicyRules { ppr2 }
	}

METADATA_OP_PROF3_RPM_CONF_ALL	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF3,
	serviceProviderName #SP_NAME3,
	profileName #NAME_OP_PROF3,
	iconType png,
	icon #ICON_OP_PROF3,
	profileClass operational,
	profileOwner {
	mccMnc #MCC_MNC3
	},
	hriServerAddress { #TEST_HRI_ADDRESS3 },
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable, disable, delete, listProfileInfo}
	tagList 'BA9BBC'H -- Tags for rpmConfiguration, hriServerAddress, lprConfiguration
	}
	},
	profileOwnerOid #S_PROFILE_OWNER_OID
	}
	lprConfiguration {
	pcmpAddress #TEST_PCMP_ADDRESS3
	}
	}

MULTIPLE_TAGS_OP_PROF1	rpmConfiguration #RPM_CONFIG_OP_PROF1, lprConfiguration { pcmpAddress #TEST_PCMP_ADDRESS1 }, hriServerAddress #TEST_HRI_ADDRESS3

PREP_DOWN_INV_CURVE	req PrepareDownloadRequest ::= {
	smdpSigned2 {
	transactionId <S_TRANSACTION_ID>,
	ccRequiredFlag FALSE
	},
	smdpSignature2 <RANDOM_SM_DP+_SIGN>,
	smdpCertificate #CERT_S_SM_DPpb_INV_CURVE
	}

PREP_DOWNLOAD_CERT_SMDP2	req PrepareDownloadRequest ::= {
	smdpSigned2 {
	transactionId <S_TRANSACTION_ID>,
	ccRequiredFlag FALSE
	},
	smdpSignature2 <S_SM_DP+_SIGNATURE2>,
	smdpCertificate #CERT_S_SM_DP2pb_SIG
	}

PREP_DOWNLOAD_INV_CERT	req PrepareDownloadRequest ::= {
	smdpSigned2 {
	transactionId <S_TRANSACTION_ID>,

Name	Content
	ccRequiredFlag FALSE }, smdpSignature2 <S_SM_DP+_SIGNATURE2>, smdpCertificate #CERT_S_SM_DPpb_INV_SIGN }

PREP_DOWNLOAD_INV_OID	req PrepareDownloadRequest ::= {
	smdpSigned2 {
	transactionId <S_TRANSACTION_ID>,
	ccRequiredFlag FALSE
	},
	smdpSignature2 <S_SM_DP+_SIGNATURE2>,
	smdpCertificate #CERT_S_SM_DPauth_SIG
	}

PREP_DOWNLOAD_INV_SIGN	req PrepareDownloadRequest ::= {
	smdpSigned2 {
	transactionId <S_TRANSACTION_ID>,
	ccRequiredFlag FALSE
	},
	smdpSignature2 <S_SM_DP+_SIGNATURE2>,
	smdpCertificate #CERT_S_SM_DPpb_SIG
	}
	
	NOTE: The <S_SM_DP+_SIGNATURE2> SHALL NOT be computed using the #SK_S_SM_DPpb_SIG but SHALL have the same length as for a valid signature.

PREP_DOWNLOAD_INV_TRANS_ID	req PrepareDownloadRequest ::= {
	smdpSigned2 {
	transactionId <INVALID_TRANSACTION_ID>,
	ccRequiredFlag FALSE
	},
	smdpSignature2 <S_SM_DP+_SIGNATURE2>,
	smdpCertificate #CERT_S_SM_DPpb_SIG
	}

PREP_DOWNLOAD_INVALID_CC	req PrepareDownloadRequest ::= {
	smdpSigned2 {
	transactionId <S_TRANSACTION_ID>,
	ccRequiredFlag TRUE
	},
	smdpSignature2 <S_SM_DP+_SIGNATURE2>,
	smdpCertificate #CERT_S_SM_DPpb_SIG
	}

PREP_DOWNLOAD_NO_AUTH	req PrepareDownloadRequest ::= {
	smdpSigned2 {
	transactionId <S_TRANSACTION_ID>,
	ccRequiredFlag FALSE
	},
	smdpSignature2 <RANDOM_SM_DP+_SIGN>,
	smdpCertificate #CERT_S_SM_DPpb_SIG
	}

PREP_DOWNLOAD_NO_CC	req PrepareDownloadRequest ::= {
	smdpSigned2 {
	transactionId <S_TRANSACTION_ID>,
	ccRequiredFlag FALSE
	},
	smdpSignature2 <S_SM_DP+_SIGNATURE2>,
	smdpCertificate #CERT_S_SM_DPpb_SIG

Name	Content
	}

PREP_DOWNLOAD_RETRY_CC	req PrepareDownloadRequest ::= {
	  smdpSigned2 {
	  transactionId <S_TRANSACTION_ID>,
	  ccRequiredFlag TRUE,
	  bppEuiccOtpk <OTPK_EUICC_ECKA>
	  },
	  smdpSignature2 <S_SM_DP+_SIGNATURE2>,
	  hashCc <S_HASHED_CC>,
	  smdpCertificate #CERT_S_SM_DPpb_SIG
	}

PREP_DOWNLOAD_WITH_CC	req PrepareDownloadRequest ::= {
	  smdpSigned2 {
	  transactionId <S_TRANSACTION_ID>,
	  ccRequiredFlag TRUE
	  },
	  smdpSignature2 <S_SM_DP+_SIGNATURE2>,
	  hashCc <S_HASHED_CC>,
	  smdpCertificate #CERT_S_SM_DPpb_SIG
	}

RETRIEVE_NOTIF_ALL	request RetrieveNotificationsListRequest ::= {
	  searchCriteria profileManagementOperation {
	  notificationInstall,
	  notificationLocalEnable,
	  notificationLocalDisable,
	  notificationLocalDelete
	  }
	}

RETRIEVE_NOTIF_DELETE	request RetrieveNotificationsListRequest::= {
	  searchCriteria profileManagementOperation {
	  notificationLocalDelete
	  }
	}

RETRIEVE_NOTIF_DISABLE	request RetrieveNotificationsListRequest::= {
	  searchCriteria profileManagementOperation {
	  notificationLocalDisable
	  }
	}

RETRIEVE_NOTIF_DISABLE_DELETE	request RetrieveNotificationsListRequest ::= {
	  searchCriteria profileManagementOperation {
	  notificationLocalDisable,
	  notificationLocalDelete
	  }
	}

RETRIEVE_NOTIF_DISABLE_ENABLE	request RetrieveNotificationsListRequest ::= {
	  searchCriteria profileManagementOperation {
	  notificationLocalDisable,
	  notificationLocalEnable
	  }
	}

RETRIEVE_NOTIF_ENABLE	request RetrieveNotificationsListRequest::= {
	  searchCriteria profileManagementOperation {
	  notificationLocalEnable
	  }
	}

RETRIEVE_NOTIF_INSTALL	request RetrieveNotificationsListRequest::= {

Name	Content

	searchCriteria profileManagementOperation {
	notificationInstall
	}
	}

RETRIEVE_NOTIF_INSTALL_ENABLE	request RetrieveNotificationsListRequest ::= {
	searchCriteria profileManagementOperation {
	notificationInstall,
	notificationLocalEnable
	}
	}

RETRIEVE_NOTIF_INSTALL_ENABLE_DISABLE	request RetrieveNotificationsListRequest ::= {
	searchCriteria profileManagementOperation {
	notificationInstall,
	notificationLocalEnable,
	notificationLocalDisable
	}
	}

RETRIEVE_NOTIF_NONE	request RetrieveNotificationsListRequest ::= {
	searchCriteria profileManagementOperation {}
	}

RETRIEVE_NOTIF_OMITTED	request RetrieveNotificationsListRequest ::= { }

RPM_CONFIG_OID2	{
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType { enable, disable, delete, listProfileInfo }
	tagList '99BA9BBC'H
	}
	},
	profileOwnerOid #S_PROFILE_OWNER_OID2
	}

RPM_CONFIG_OP_PROF1	{
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable, disable, delete, listProfileInfo}
	tagList '99BA9BBC'H
	}
	},
	profileOwnerOid #S_PROFILE_OWNER_OID
	}

SET_EUICC_CONFIGURED_ADDRESS_1	request SetDefaultDpAddressRequest::={
	defaultDpAddress #TEST_DP_ADDRESS1
	}

SET_EUICC_CONFIGURED_ADDRESS_2	request SetDefaultDpAddressRequest::={
	defaultDpAddress #TEST_DP_ADDRESS2
	}

SET_EUICC_CONFIGURED_ADDRESS_EMPTY	request SetDefaultDpAddressRequest::={
	defaultDpAddress ""
	}

SET_NICKNAME_EMPTY_OP_PROF1	setNicknameReq SetNicknameRequest ::= {
	iccid #ICCID_OP_PROF1,
	profileNickname ""
	}

SET_NICKNAME_ICCID_UNKNOWN	setNicknameReq SetNicknameRequest ::= { iccid #ICCID_UNKNOWN,

Name	Content
	profileNickname #NICKNAME2 }

SET_NICKNAME_OP_PROF1	setNicknameReq SetNicknameRequest ::= {
	iccid #ICCID_OP_PROF1,
	profileNickname #NICKNAME2
	}

D.3.2 ES10x Responses

Name	Content

ENTERPRISE_CONFIG_NO_ENT_RULES	{
	enterpriseOid #S_ENTERPRISE_OID,
	enterpriseName #ENTERPRISE_NAME1,
	}

ENTERPRISE_CONFIG2	{
	enterpriseOid #S_ENTERPRISE_OID,
	enterpriseName #ENTERPRISE_NAME1,
	enterpriseRules {
	enterpriseRuleBits {
	referenceEnterpriseRule,
	priorityEnterpriseProfile,
	onlyEnterpriseProfilesCanBeInstalled
	},
	numberOfNonEnterpriseProfiles 0
	}
	}

NOTIF_METADATA_DELETE1 (NotificationMetadata)	{
	seqNumber <NOTIF_SEQ_NO_DE1>,
	profileManagementOperation {
	notificationLocalDelete},
	notificationAddress #TEST_DP_ADDRESS1,
	iccid #ICCID_OP_PROF1
	}

NOTIF_METADATA_DELETE2 (NotificationMetadata)	{
	seqNumber <NOTIF_SEQ_NO_DE2>,
	profileManagementOperation {
	notificationLocalDelete},
	notificationAddress #TEST_DP_ADDRESS2,
	iccid #ICCID_OP_PROF2
	}

NOTIF_METADATA_DISABLE1 (NotificationMetadata)	{
	seqNumber <NOTIF_SEQ_NO_DI1>,
	profileManagementOperation {
	notificationLocalDisable},
	notificationAddress #TEST_DP_ADDRESS1,
	iccid #ICCID_OP_PROF1
	}

NOTIF_METADATA_DP2_EN1_RPM (NotificationMetadata)	{
	seqNumber <NOTIF_SEQ_NO_EN1_RPM>,
	profileManagementOperation {
	notificationRpmEnable},

Name	Content
	notificationAddress #TEST_DP_ADDRESS2, iccid #ICCID_OP_PROF1 }

NOTIF_METADATA_EN1_RPM
(NotificationMetadata)
	{
	seqNumber <NOTIF_SEQ_NO_EN1_RPM>,
	profileManagementOperation {
	notificationRpmEnable },
	notificationAddress #TEST_DP_ADDRESS1,
	iccid #ICCID_OP_PROF1
	}

NOTIF_METADATA_ENABLE1
(NotificationMetadata)
	{
	seqNumber <NOTIF_SEQ_NO_EN1>,
	profileManagementOperation {
	notificationLocalEnable},
	notificationAddress #TEST_DP_ADDRESS1,
	iccid #ICCID_OP_PROF1
	}

NOTIF_METADATA_ENABLE2
(NotificationMetadata)
	{
	seqNumber <NOTIF_SEQ_NO_EN2>,
	profileManagementOperation {
	notificationLocalEnable},
	notificationAddress #TEST_DP_ADDRESS2,
	iccid #ICCID_OP_PROF2
	}

NOTIF_METADATA_INSTALL1
(NotificationMetadata)
	{
	seqNumber <NOTIF_SEQ_NO_IN1>,
	profileManagementOperation {
	notificationInstall },
	notificationAddress #TEST_DP_ADDRESS1,
	iccid #ICCID_OP_PROF1
	}

NOTIF_METADATA_INSTALL1_DP1_PIR
(NotificationMetadata)
	{
	seqNumber <NOTIF_SEQ_NO_IN1_DP1>,
	profileManagementOperation {
	notificationInstall },
	notificationAddress #TEST_DP_ADDRESS1,
	iccid #ICCID_OP_PROF1
	}

NOTIF_METADATA_INSTALL1_DP2_OSN
(NotificationMetadata)
	{
	seqNumber <NOTIF_SEQ_NO_IN1_DP2>,
	profileManagementOperation {
	notificationInstall },
	notificationAddress #TEST_DP_ADDRESS2,
	iccid #ICCID_OP_PROF1
	}

NOTIF_METADATA_INSTALL1_PIR
(NotificationMetadata)
	{
	seqNumber <NOTIF_SEQ_NO_IN1_PIR>,
	profileManagementOperation {
	notificationInstall
	},
	notificationAddress #TEST_DP_ADDRESS1,
	iccid #ICCID_OP_PROF1
	}

NOTIF_METADATA_INSTALL2
(NotificationMetadata)
	{
	seqNumber <NOTIF_SEQ_NO_IN2>,
	profileManagementOperation {
	notificationInstall

Name	Content
	}, notificationAddress #TEST_DP_ADDRESS2, iccid #ICCID_OP_PROF2 }

NOTIF_METADATA_INSTALL2_PIR
(NotificationMetadata)
	{
	seqNumber <NOTIF_SEQ_NO_IN2_PIR>,
	profileManagementOperation {
	notificationInstall
	},
	notificationAddress #TEST_DP_ADDRESS2,
	iccid #ICCID_OP_PROF2
	}

NOTIF_METADATA_PROF1_DP1_RPR
(NotificationMetadata)
	{
	seqNumber <NOTIF_SEQ_NO_PROF1_RPR>,
	profileManagementOperation {
	loadRpmPackageResult },
	notificationAddress #TEST_DP_ADDRESS1
	}

NOTIF_METADATA_RPR
(NotificationMetadata)
	{
	seqNumber <NOTIF_SEQ_NO_PROF1_RPR>,
	profileManagementOperation {
	loadRpmPackageResult },
	notificationAddress #TEST_DP_ADDRESS1,
	}

NOTIF_METADATA2_DELETE1
(NotificationMetadata)
	{
	seqNumber <NOTIF_SEQ_NO2_DE1>,
	profileManagementOperation {
	notificationLocalDelete},
	notificationAddress #TEST_DP_ADDRESS2,
	iccid #ICCID_OP_PROF1
	}

NOTIF_METADATA2_DISABLE1
(NotificationMetadata)
	{
	seqNumber <NOTIF_SEQ_NO2_DI1>,
	profileManagementOperation {
	notificationLocalDisable},
	notificationAddress #TEST_DP_ADDRESS2,
	iccid #ICCID_OP_PROF1
	}

NOTIF_METADATA2_ENABLE1
(NotificationMetadata)
	{
	seqNumber <NOTIF_SEQ_NO2_EN1>,
	profileManagementOperation {
	notificationLocalEnable},
	notificationAddress #TEST_DP_ADDRESS2,
	iccid #ICCID_OP_PROF1
	}

PPR1_WITH_OWNER_GID
(ProfilePolicyAuthorisationRule)
	{
	pprIds { ppr1 },
	allowedOperators {
	{ mccMnc #MCC_MNC2,
	gid1 #GID1,
	gid2 #GID2
	}
	},
	pprFlags {consentRequired}
	}

Name	Content

PPR1_WITHOUT_GID
(ProfilePolicyAuthorisationRule)
	{
	pprIds { ppr1 },
	allowedOperators {
	{ mccMnc #MCC_MNC4 }
	},
	pprFlags {consentRequired}
	}

PPR2_WITHOUT_CONSENT
(ProfilePolicyAuthorisationRule)
	{
	pprIds { ppr2 },
	allowedOperators {
	{ mccMnc '92EEEE'H, gid1 ''H, gid2 ''H}
	},
	pprFlags { }
	}

PPRS_ALLOWED
(ProfilePolicyAuthorisationRule)
	{
	pprIds { ppr1, ppr2 },
	allowedOperators {
	{ mccMnc 'EEEEEE'H, gid1 ''H, gid2 ''H}
	},
	pprFlags {consentRequired}
	}

PPR2_ALLOWED
(ProfilePolicyAuthorisationRule)
	{
	pprIds { ppr2 },
	allowedOperators {
	{ mccMnc 'EEEEEE'H, gid1 ''H, gid2 ''H}
	},
	pprFlags {consentRequired}
	}

PROFILE_INFO1
(ProfileInfo)
	{
	iccid #ICCID_OP_PROF1,
	isdpAid <ISD_P_AID1>,
	profileState enabled,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational
	}

PROFILE_INFO1_DISABLED
(ProfileInfo)
	{
	iccid #ICCID_OP_PROF1,
	isdpAid <ISD_P_AID1>,
	profileState disabled,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational
	}

PROFILE_INFO1_MEP (ProfileInfo)	{ iccid #ICCID_OP_PROF1, isdpAid <ISD_P_AID1>, profileState enabled or disabled, serviceProviderName #SP_NAME1,

Name	Content
	profileName #NAME_OP_PROF1, iconType png, icon #ICON_OP_PROF1, profileClass operational, enabledOnEsimPort <ANY_PORT_VALUE> }

PROFILE_INFO1_MEPB
(ProfileInfo)
	{
	iccid #ICCID_OP_PROF1,
	isdpAid <ISD_P_AID1>,
	profileState enabled OR disabled,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational
	}

PROFILE_INFO2
(ProfileInfo)
	{
	iccid #ICCID_OP_PROF2,
	isdpAid <ISD_P_AID2>,
	profileState disabled,
	serviceProviderName #SP_NAME2,
	profileName #NAME_OP_PROF2,
	iconType png,
	icon #ICON_OP_PROF2,
	profileClass operational
	}

PROFILE_INFO2_ENABLED
(ProfileInfo)
	{
	iccid #ICCID_OP_PROF2,
	isdpAid <ISD_P_AID2>,
	profileState enabled,
	serviceProviderName #SP_NAME2,
	profileName #NAME_OP_PROF2,
	iconType png,
	icon #ICON_OP_PROF2,
	profileClass operational
	}

PROFILE_INFO2_ENABLED_MEP
(ProfileInfo)
	{
	iccid #ICCID_OP_PROF2,
	isdpAid <ISD_P_AID2>,
	profileState enabled,
	serviceProviderName #SP_NAME2,
	profileName #NAME_OP_PROF2,
	iconType png,
	icon #ICON_OP_PROF2,
	profileClass operational,
	enabledOnEsimPort <ANY_PORT_VALUE>
	}

Name	Content

PROFILE_INFO2_ENABLED_ON_OTHER_PORT	{
	iccid #ICCID_OP_PROF2,
	isdpAid <ISD_P_AID2>,
	profileState enabled or disabled,
	serviceProviderName #SP_NAME2,
	profileName #NAME_OP_PROF2,
	iconType png,
	icon #ICON_OP_PROF2,
	profileClass operational,
	enabledOnEsimPort <ANY_PORT_VALUE>
	}

PROFILE_INFO2_MEPB
(ProfileInfo)
	{
	iccid #ICCID_OP_PROF2,
	isdpAid <ISD_P_AID2>,
	profileState enabled OR disabled,
	serviceProviderName #SP_NAME2,
	profileName #NAME_OP_PROF2,
	iconType png,
	icon #ICON_OP_PROF2,
	profileClass operational
	}

PROFILE_INFO3
(ProfileInfo)
	{
	iccid #ICCID_OP_PROF3,
	isdpAid <ISD_P_AID3>,
	profileState disabled,
	profileNickname #NICKNAME3,
	serviceProviderName #SP_NAME3,
	profileName #NAME_OP_PROF3,
	iconType png,
	icon #ICON_OP_PROF3,
	profileClass operational
	}

Name	Content

PROFILE_INFO3_ENABLED
(ProfileInfo)
	{
	iccid #ICCID_OP_PROF3,
	isdpAid <ISD_P_AID3>,
	profileState enabled,
	profileNickname #NICKNAME3,
	serviceProviderName #SP_NAME3,
	profileName #NAME_OP_PROF3,
	iconType png,
	icon #ICON_OP_PROF3,
	profileClass operational
	}

PROFILE_INFO3_ENABLED_ON_OTHER_PORT
(ProfileInfo)
	{
	iccid #ICCID_OP_PROF3,
	isdpAid <ISD_P_AID3>,
	profileState enabled or disabled,
	profileNickname #NICKNAME3,
	serviceProviderName #SP_NAME3,
	profileName #NAME_OP_PROF3,
	iconType png,
	icon #ICON_OP_PROF3,
	profileClass operational,
	enabledOnEsimPort <ANY_PORT_VALUE>
	}

PROFILE_INFO3_MEPB
(ProfileInfo)
	{
	iccid #ICCID_OP_PROF3,
	isdpAid <ISD_P_AID3>,
	profileState enabled OR disabled,
	profileNickname #NICKNAME3,
	serviceProviderName #SP_NAME3,
	profileName #NAME_OP_PROF3,
	iconType png,
	icon #ICON_OP_PROF3,
	profileClass operational
	}

PROFILE_INFO4_ENABLED
(ProfileInfo)
	{
	iccid #ICCID_OP_PROF4,
	isdpAid <ISD_P_AID4>,
	profileState enabled,
	serviceProviderName #SP_NAME4,
	profileName #NAME_OP_PROF4,
	iconType png,
	icon #ICON_OP_PROF4,
	profileClass operational
	}

PROFILES_INFO_ICCID_TAGLIST1 (ProfileInfo)	{profileState enabled}
PROFILES_INFO_ICCID_TAGLIST2 (ProfileInfo)	{iconType png}
PROFILES_INFO_ICCID_TAGLIST3 (ProfileInfo)	{profileClass operational }
PROFILES_INFO_ICCID_TAGLIST4	notificationConfigurationInfo from #METADATA_OP_PROF1

Name	Content
(ProfileInfo)	
PROFILES_INFO_ICCID_TAGLIST5 (ProfileInfo)	profilePolicyRules from #METADATA_OP_PROF3
PROFILES_INFO_ICCID_TAGLIST6 (ProfileInfo)	profilePolicyRules from METADATA_OP_PROF1_RPM_CONF_ALL_PPR1

PROFILES_INFO_RPM_TAGLIST1
(ProfileInfo)
	{
	iccid #ICCID_OP_PROF1,
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable, disable, delete,
	listProfileInfo, contactPcmp}
	tagList '99BA9BBC'H -- Tags for
	PPR,rpmConfig,hriConfig,lprConfig
	}
	},
	profileOwnerOid #S_PROFILE_OWNER_OID
	}
	}

PROFILES_INFO_RPM_TAGLIST2
(ProfileInfo)
	{
	iccid #ICCID_OP_PROF3,
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable, disable, delete,
	listProfileInfo}
	tagList 'BA9BBC'H -- Tags for
	PPR,rpmConfig,hriConfig,lprConfig
	}
	},
	profileOwnerOid #S_PROFILE_OWNER_OID
	}
	}

PROFILES_INFO_RPM_TAGLIST3
(ProfileInfo)
	{
	iccid #ICCID_OP_PROF1,
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable, disable, delete,
	listProfileInfo, contactPcmp}
	tagList '99BA9BBC'H -- Tags for
	PPR,rpmConfig,hriConfig,lprConfig
	}
	},
	profileOwnerOid #S_PROFILE_OWNER_OID
	},
	hriServerAddress { #TEST_HRI_ADDRESS1 }
	}

PROFILES_INFO_RPM_TAGLIST4
(ProfileInfo)
	{
	iccid #ICCID_OP_PROF3,
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable, disable, delete,
	listProfileInfo}

Name	Content

	tagList 'BA9BBC'H -- Tags for rpmConfig,hriConfig,lprConfig
	}
	},
	profileOwnerOid #S_PROFILE_OWNER_OID
	},
	hriServerAddress { #TEST_HRI_ADDRESS3 },
	lprConfiguration {
	pcmpAddress #TEST_PCMP_ADDRESS3
	}
	}

PROFILES_INFO_RPM_TAGLIST5
(ProfileInfo)
	{
	iccid #ICCID_OP_PROF2,
	rpmConfiguration {
	managingDpList {
	{ managingDP #S_SM_DP+_OID,
	rpmType {enable, disable, delete, listProfileInfo, contactPcmp}
	tagList '99BA9BBC9F1F'H -- *Tags for PPR*, rpmConfiguration, hriServerAddress, lprConfiguration, ICCID, ServiceDescription
	}
	},
	profileOwnerOid #S_PROFILE_OWNER_OID2
	},
	serviceDescription {voice, data}
	}

PROFILES_INFO_RPM_TAGLIST6
(ProfileInfo)
	{
	iccid #ICCID_OP_PROF2,
	enterpriseConfiguration {
	enterpriseOid #S_ENTERPRISE_OID,
	enterpriseName #ENTERPRISE_NAME1,
	enterpriseRules {
	enterpriseRuleBits {
	referenceEnterpriseRule,
	priorityEnterpriseProfile,
	onlyEnterpriseProfilesCanBeInstalled
	},
	numberOfNonEnterpriseProfiles 0
	}
	},
	serviceDescription {voice, data}
	}

PROFILES_INFO_TAGLIST_ICCID (ProfileInfo)	{iccid #ICCID_OP_PROF1}, {iccid #ICCID_OP_PROF2}, {iccid #ICCID_OP_PROF3}
PROFILES_INFO_TAGLIST_ICON (ProfileInfo)	{icon #ICON_OP_PROF1}, {icon #ICON_OP_PROF2}, {icon #ICON_OP_PROF3}
PROFILES_INFO_TAGLIST_ISDPAID (ProfileInfo)	{isdpAid <ISD_P_AID1>}, {isdpAid <ISD_P_AID2>}, {isdpAid <ISD_P_AID3>}

Name	Content
PROFILES_INFO_TAGLIST_PROFILE_NAME (ProfileInfo)	{profileName #NAME_OP_PROF1}, {profileName #NAME_OP_PROF2}, {profileName #NAME_OP_PROF3}
PROFILES_INFO_TAGLIST_PROFILE_NICKNAME (ProfileInfo)	{profileNickname #NICKNAME3}
PROFILES_INFO_TAGLIST_PROFILE_OWNER (ProfileInfo)	{profileOwner #OWNER_OP_PROF1}, {profileOwner #OWNER_OP_PROF2}, {profileOwner #OWNER_OP_PROF2}
PROFILES_INFO_TAGLIST_SMDP_PROP_DATA (ProfileInfo)	{dpProprietaryData #SMDP_PROP_DATA1}
PROFILES_INFO_TAGLIST_SP_NAME (ProfileInfo)	{serviceProviderName #SP_NAME1}, {serviceProviderName #SP_NAME2}, {serviceProviderName #SP_NAME3}

PROFILES_INFO_TAGLIST1
(ProfileInfo)
	{
	iccid #ICCID_OP_PROF1,
	profileState enabled
	},
	{
	iccid #ICCID_OP_PROF2,
	profileState disabled
	},
	{
	iccid #ICCID_OP_PROF3,
	profileState disabled
	}

PROFILES_INFO_TAGLIST2
(ProfileInfo)
	{
	profileState enabled
	},
	{
	profileState disabled
	},
	{
	profileState disabled,
	profileNickname #NICKNAME3
	}

PROFILES_INFO_TAGLIST3
(ProfileInfo)
	{
	iconType png,
	icon #ICON_OP_PROF1
	},
	{
	iconType png,
	icon #ICON_OP_PROF2
	},
	{
	iconType png,
	icon #ICON_OP_PROF3
	}

PROFILES_INFO_TAGLIST4 (ProfileInfo)	{ profileState enabled, icon #ICON_OP_PROF1 }, { profileState disabled,

Name	Content

	icon #ICON_OP_PROF2
	},
	{
	profileState disabled,
	icon #ICON_OP_PROF3
	}

PROFILES_INFO_TAGLIST5_MEPA1
(ProfileInfo)
	{
	iccid #ICCID_OP_PROF1,
	profileState enabled,
	enabledOnEsimPort 0x01
	
	},
	{
	iccid #ICCID_OP_PROF2,
	profileState enabled,
	enabledOnEsimPort 0x02
	},
	{
	iccid #ICCID_OP_PROF3,
	profileState disabled
	
	}

PROFILES_INFO_TAGLIST5_MEPA2_12
(ProfileInfo)
	{
	iccid #ICCID_OP_PROF1,
	profileState enabled,
	enabledOnEsimPort 0x01
	
	},
	{
	iccid #ICCID_OP_PROF2,
	profileState enabled,
	enabledOnEsimPort 0x02
	},
	{
	iccid #ICCID_OP_PROF3,
	profileState disabled
	}

PROFILES_INFO_TAGLIST5_MEPA2_21
(ProfileInfo)
	{
	iccid #ICCID_OP_PROF1,
	profileState enabled,
	enabledOnEsimPort 0x02
	
	},
	{
	iccid #ICCID_OP_PROF2,
	profileState enabled,
	enabledOnEsimPort 0x01
	},
	{
	iccid #ICCID_OP_PROF3,
	profileState disabled
	}

Name	Content

PROFILES_INFO_TAGLIST5_MEPB (ProfileInfo)	{
	iccid #ICCID_OP_PROF1,
	profileState disabled or enabled,
	enabledOnEsimPort 0x00
	
	},
	{
	iccid #ICCID_OP_PROF2,
	profileState disabled or enabled,
	enabledOnEsimPort 0x01
	},
	{
	iccid #ICCID_OP_PROF3,
	profileState disabled
	
	}

R_AUTH_SERVER_INV_CERT	resp AuthenticateServerResponse ::=
	authenticateResponseError : {
	transactionId <S_TRANSACTION_ID>,
	authenticateErrorCode invalidCertificate
	}

R_AUTH_SERVER_INV_CHALLENGE	resp AuthenticateServerResponse ::=
	authenticateResponseError : {
	transactionId <S_TRANSACTION_ID>,
	authenticateErrorCode euiccChallengeMismatch
	}

R_AUTH_SERVER_INV_CI	resp AuthenticateServerResponse ::=
	authenticateResponseError : {
	transactionId <S_TRANSACTION_ID>,
	authenticateErrorCode ciPKUnknown
	}

R_AUTH_SERVER_INV_CURV	resp AuthenticateServerResponse ::=
	authenticateResponseError : {
	transactionId <S_TRANSACTION_ID>,
	authenticateErrorCode unsupportedCurve
	}

R_AUTH_SERVER_INV_OID	resp AuthenticateServerResponse ::=
	authenticateResponseError : {
	transactionId <S_TRANSACTION_ID>,
	authenticateErrorCode invalidOid
	}

R_AUTH_SERVER_INV_SIGN	resp AuthenticateServerResponse ::=
	authenticateResponseError : {
	transactionId <S_TRANSACTION_ID>,
	authenticateErrorCode invalidSignature
	}

R_AUTH_SERVER_NO_SESSION	resp AuthenticateServerResponse ::=
	authenticateResponseError : {
	transactionId <S_TRANSACTION_ID>,
	authenticateErrorCode noSessionContext
	}

R_CANCEL_SESSION_INV_TRANS_ID	resp CancelSessionResponse ::= cancelSessionResponseError : invalidTransactionId
R_CANCEL_SESSION_LOAD_BPP	resp CancelSessionResponse ::= cancelSessionResponseOk : {

Name	Content

	euiccCancelSessionSigned {
	transactionId <S_TRANSACTION_ID>,
	smdpOid #S_SM_DP+_OID,
	reason loadBppExecutionError
	},
	euiccCancelSessionSignature
	<EUICC_CS_SIGNATURE>
	}

R_CANCEL_SESSION_METADATA	resp CancelSessionResponse ::=
	cancelSessionResponseOk : {
	euiccCancelSessionSigned {
	transactionId <S_TRANSACTION_ID>,
	smdpOid #S_SM_DP+_OID,
	reason metadataMismatch
	},
	euiccCancelSessionSignature
	<EUICC_CS_SIGNATURE>
	}

R_CANCEL_SESSION_POSTPONED	resp CancelSessionResponse ::=
	cancelSessionResponseOk : {
	euiccCancelSessionSigned {
	transactionId <S_TRANSACTION_ID>,
	smdpOid #S_SM_DP+_OID,
	reason postponed
	},
	euiccCancelSessionSignature
	<EUICC_CS_SIGNATURE>
	}

R_CANCEL_SESSION_PPR	resp CancelSessionResponse ::=
	cancelSessionResponseOk : {
	euiccCancelSessionSigned {
	transactionId <S_TRANSACTION_ID>,
	smdpOid #S_SM_DP+_OID,
	reason pprNotAllowed
	},
	euiccCancelSessionSignature
	<EUICC_CS_SIGNATURE>
	}

R_CANCEL_SESSION_REJ	resp CancelSessionResponse ::=
	cancelSessionResponseOk : {
	euiccCancelSessionSigned {
	transactionId <S_TRANSACTION_ID>,
	smdpOid #S_SM_DP+_OID,
	reason endUserRejection
	},
	euiccCancelSessionSignature
	<EUICC_CS_SIGNATURE>
	}

R_CANCEL_SESSION_TIMEOUT	resp CancelSessionResponse ::=
	cancelSessionResponseOk {
	euiccCancelSessionSigned {
	transactionId <S_TRANSACTION_ID>,
	smdpOid #S_SM_DP+_OID,
	reason timeout
	},
	euiccCancelSessionSignature
	<EUICC_CS_SIGNATURE>

Name	Content
	}

R_CANCEL_SESSION_UNDEF	resp CancelSessionResponse ::=
	cancelSessionResponseOk : {
	  euiccCancelSessionSigned {
	  transactionId <S_TRANSACTION_ID>,
	  smdpOid #S_SM_DP+_OID,
	  reason undefinedReason
	  },
	  euiccCancelSessionSignature
	<EUICC_CS_SIGNATURE>
	}

R_CHALLENGE	response GetEuiccChallengeResponse ::=
	{
	  euiccChallenge <EUICC_CHALLENGE>
	}

R_CONF_OP_PROF1	resp ProfileInfoListResponse ::=
	  profileInfoListOk :{
	  {
	  isdpAid <ISD_P_AID>,
	  dpProprietaryData {
	  dpOid #S_SM_DP+_OID,
	  additionalSmdpData
	  #ADDITIONAL_SMDP_DATA_MAX_LENGTH
	  }
	  }
	}
	
	-- NOTE: Instead of
	DpProprietaryData ::= SEQUENCE {
	  dpOid OBJECT IDENTIFIER
	  -- additional data objects defined by the
	  -- SM-DP+ MAY follow
	}
	
	-- the following structure is used to test the
	-- DpProprietaryData size:
	DpProprietaryData ::= SEQUENCE {
	  dpOid OBJECT IDENTIFIER,
	  additionalSmdpData OCTET STRING OPTIONAL
	}

R_DEFAULT_RAT	response GetRatResponse ::= {
	  rat {
	  #PPRS_ALLOWED
	  }
	}

R_DELETE_PROFILE_DISALLOWEDBY POLICY	respDelProf DeleteProfileResponse ::= {
	  deleteResult disallowedByPolicy
	}

R_DELETE_PROFILE_ICCID_ISDP_NOT FOUND	resp DeleteProfileResponse ::= {
	  deleteResult iccidOrAidNotFound
	}

R_DELETE_PROFILE_NOTDISABLESTA TE	respDelProf DeleteProfileResponse ::= {
	  deleteResult profileNotInDisabledState
	}

R_DELETE_PROFILE_OK	respDelProf DeleteProfileResponse ::= { deleteResult ok

Name	Content
	}

R_DISABLE_PROFILE_DISALLOWEDbyPOLICY	resp DisableProfileResponse ::= {
	  disableResult disallowedByPolicy
	}

R_DISABLE_PROFILE_ICCID_ISDP_NOTFOUND	resp DisableProfileResponse ::= {
	  disableResult iccidOrAidNotFound
	}

R_DISABLE_PROFILE_NOT_ENABLE_STATE	resp DisableProfileResponse ::= {
	  disableResult profileNotInEnabledState
	}

R_DISABLE_PROFILE_OK	resp DisableProfileResponse ::= {
	  disableResult ok
	}

R_ENABLE_PROFILE_DISALLOWEDbyPOLICY	respEnaPro EnableProfileResponse ::= {
	  enableResult disallowedByPolicy
	}

R_ENABLE_PROFILE_ICCID_ISDP_NOTFOUND	respEnaPro EnableProfileResponse ::= {
	  enableResult iccidOrAidNotFound
	}

R_ENABLE_PROFILE_MEP_A2_OK	resp EnableProfileResponse ::= {
	  enableResult ok,
	  targetEsimPort <MEP-A2_TARGET_ESIM_PORT>
	}

R_ENABLE_PROFILE_NOT_DISABLE_STATE	respEnaPro EnableProfileResponse ::= {
	  enableResult profileNotInDisabledState
	}

R_ENABLE_PROFILE_OK	resp EnableProfileResponse ::= {
	  enableResult ok
	}

R_ES10a_GECA_DS	response EuiccConfiguredDataResponse ::= {
	  …
	  -- defaultDpAddress SHALL not be present
	  rootDsAddress #TEST_ROOT_DS_ADDRESS
	}

R_ES10a_GECA_DS_DP_1	response EuiccConfiguredDataResponse ::= {
	  …
	  defaultDpAddress #TEST_DP_ADDRESS1,
	  rootDsAddress #TEST_ROOT_DS_ADDRESS,
	  …
	}

R_ES10a_GECA_DS_DP_2	response EuiccConfiguredDataResponse ::= {
	  …
	  defaultDpAddress #TEST_DP_ADDRESS2,
	  rootDsAddress #TEST_ROOT_DS_ADDRESS,
	  …
	}

R_ES10a_SD_DP_A_OK	response SetDefaultDpAddressResponse::= {
	  setDefaultDpAddressResult ok
	}

R_EUICC_INFO1	response EUICCInfo1::=
	
	{
	  lowestSvn #RSP_VERSION_LOWEST_H,
	
	  euiccCiPKIdListForVerification
	  <EUICC_CI_PK_ID_LIST_FOR_VERIFICATION>,
	  euiccCiPKIdListForSigning

Name	Content
	<EUICC_CI_PK_ID_LIST_FOR_SIGNING>, euiccCiPKIdListForSigningV3 <EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3>, -- OPTIONAL euiccRspCapability <EUICC_RSP_CAPABILITY>, highestSvn #IUT_RSP_VERSION_HIGHEST}

R_EUICC_INFO2	response EUICCInfo2::=
	{
	baseProfilePackageVersion #IUT_SIMA_VERSION,
	lowestSvn #RSP_VERSION_LOWEST_H,
	euiccFirmwareVersion #IUT_EUICC_FIRMWARE_VER,
	extCardResource <EXT_CARD_RESOURCE>,
	uiccCapability #IUT_UICC_CAPABILITY,
	ts102241Version #IUT_TS102241_VERSION, --
	OPTIONAL
	globalplatformVersion
	#IUT_GLOBALPLATFORM_VERSION,
	euiccRspCapability <EUICC_RSP_CAPABILITY>,
	euiccCiPKIdListForVerification
	<EUICC_CI_PK_ID_LIST_FOR_VERIFICATION>,
	euiccCiPKIdListForSigning
	<EUICC_CI_PK_ID_LIST_FOR_SIGNING>,
	euiccCategory #IUT_EUICC_CATEGORY,
	forbiddenProfilePolicyRules <PPR_IDS>, --
	OPTIONAL
	ppVersion #IUT_PP_VERSION,
	sasAcreditationNumber
	#IUT_SAS_ACCREDITATION_NUMBER,
	certificationDataObject {
	platformLabel #IUT_PLATFORM_LABEL,
	discoveryBaseURL #IUT_DLOA_URL
	},
	treProperties #IUT_TRE_PROPERTIES,
	treProductReference #IUT_TRE_REFERENCE, --
	OPTIONAL
	additionalProfilePackageVersions
	#IUT_EUICC_ADD_PP_VERSIONS, -- OPTIONAL
	lpaMode {lpad},
	euiccCiPKIdListForSigningV3
	<EUICC_CI_PK_ID_LIST_FOR_SIGNING_V3>,
	additionalEuiccInfo
	#IUT_ADDITIONAL_EUICC_INFO, --OPTIONAL
	highestSvn #IUT_RSP_VERSION_HIGHEST}

R_EUICC_MEMORY_RESET_OK	resp EuiccMemoryResetResponse ::= {
	resetResult ok
	}

R_GET_METADATA_OP_PROF1	resp ProfileInfoListResponse ::=
	profileInfoListOk :{
	{
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,

Name	Content

	profileClass operational,
	notificationConfigurationInfo {
	{ profileManagementOperation {
	notificationInstall,
	notificationLocalEnable,
	notificationLocalDisable,
	notificationLocalDelete
	},
	notificationAddress
	#TEST_DP_ADDRESS1
	}
	},
	profileOwner {
	mccMnc #MCC_MNC1
	},
	profilePolicyRules {ppr1}
	}
	}

R_GET_METADATA_OP_PROF1_NO_SERVICE_SPECIFIC	resp ProfileInfoListResponse ::=
	profileInfoListOk :{
	{
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	notificationConfigurationInfo {
	{ profileManagementOperation {
	notificationInstall,
	notificationLocalEnable,
	notificationLocalDisable,
	notificationLocalDelete
	},
	notificationAddress
	#TEST_DP_ADDRESS1
	}
	},
	profileOwner {
	mccMnc #MCC_MNC1
	}
	}
	}

Name	Content

R_GET_METADATA_OP_PROF1_SERVICE_SPECIFIC	resp ProfileInfoListResponse ::=
	profileInfoListOk :{
	{
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileClass operational,
	notificationConfigurationInfo {
	{ profileManagementOperation {
	notificationInstall,
	notificationLocalEnable,
	notificationLocalDisable,
	notificationLocalDelete
	},
	notificationAddress #TEST_DP_ADDRESS1
	}
	},
	profileOwner {
	mccMnc #MCC_MNC1
	},
	serviceSpecificDataStoredInEuicc #VENDOR_SPECIFIC_EXTENSION1
	}
	}

R_GET_PROF_NOTIF_CONF	resp ProfileInfoListResponse ::=
	profileInfoListOk :{
	{
	iccid #ICCID_OP_PROF1,
	notificationConfigurationInfo {
	{ profileManagementOperation {
	notificationInstall
	},
	notificationAddress #TEST_DP_ADDRESS3
	},
	{ profileManagementOperation {
	notificationInstall
	},
	notificationAddress #TEST_DP_ADDRESS2
	},
	{ profileManagementOperation {
	notificationLocalEnable
	},
	notificationAddress #TEST_DP_ADDRESS2
	},
	{ profileManagementOperation {
	notificationLocalEnable
	},
	notificationAddress #TEST_DP_ADDRESS3
	},
	{ profileManagementOperation {
	notificationLocalDisable
	},

Name	Content

	notificationAddress #TEST_DP_ADDRESS3
	},
	{ profileManagementOperation {
	notificationLocalDisable
	},
	notificationAddress #TEST_DP_ADDRESS4
	},
	{ profileManagementOperation {
	notificationLocalDelete
	},
	notificationAddress #TEST_DP_ADDRESS1
	},
	{ profileManagementOperation {
	notificationLocalDelete
	},
	notificationAddress #TEST_DP_ADDRESS3
	}
	}
	}
	}

R_GET_UPDATE_N1	resp ProfileInfoListResponse ::=
	profileInfoListOk :{
	{
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profilePolicyRules { ppr2 }
	}
	}

R_GET_UPDATE_N2	resp ProfileInfoListResponse ::=
	profileInfoListOk :{
	{
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType jpg,
	icon #ICON_JPG,
	profilePolicyRules { ppr1 }
	}
	}

R_GET_UPDATE_N3	resp ProfileInfoListResponse ::=
	profileInfoListOk :{
	{
	serviceProviderName #SP_NAME2,
	profileName #NAME_OP_PROF2,
	iconType png,
	icon #ICON_OP_PROF1
	-- profilePolicyRules SHALL not be present
	}
	}

R_GET_UPDATE_N6	resp ProfileInfoListResponse ::=
	profileInfoListOk :{
	{
	serviceProviderName #SP_NAME2,
	profileName #NAME_OP_PROF2,
	iconType png,

Name	Content
	icon #ICON_OP_PROF1 -- profilePolicyRules SHALL not be present } }

R_GET_UPDATE_N7	resp ProfileInfoListResponse ::=
	profileInfoListOk :{
	{
	serviceProviderName #SP_NAME2,
	profileName #NAME_OP_PROF2
	-- iconType SHALL not be present
	-- icon SHALL not be present
	-- profilePolicyRules SHALL not be present
	}
	}

R_GET_UPDATE_N8	resp ProfileInfoListResponse ::=
	profileInfoListOk :{
	{
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	notificationConfigurationInfo {
	{ profileManagementOperation {
	notificationLocalDelete
	},
	notificationAddress #TEST_DP_ADDRESS2
	}
	},
	profilePolicyRules { ppr1, ppr2 }
	}
	}

R_GET_UPDATE_N9	resp ProfileInfoListResponse ::=
	profileInfoListOk :{
	{
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profilePolicyRules { ppr1, ppr2 },
	}
	}

R_GET_UPDATE_N10	resp ProfileInfoListResponse ::=
	profileInfoListOk :{
	{
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profilePolicyRules { ppr1, ppr2 },

Name	Content
	serviceSpecificDataStoredInEuicc #VENDOR_SPECIFIC_EXTENSION2, } }

R_GET_UPDATE_N11	resp ProfileInfoListResponse ::=
	profileInfoListOk :{
	{
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profilePolicyRules { ppr1, ppr2 }
	}
	}

R_GET_UPDATE_N12	resp ProfileInfoListResponse ::=
	profileInfoListOk :{
	{
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profilePolicyRules { ppr1, ppr2 },
	rpmConfiguration {
	managingDpList {},
	profileOwnerOid #S_PROFILE_OWNER_OID
	}
	}
	}

R_GET_UPDATE_N13	resp ProfileInfoListResponse ::=
	profileInfoListOk :{
	{
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profilePolicyRules { ppr1, ppr2 },
	}
	}

R_GET_UPDATE_N14	resp ProfileInfoListResponse ::=
	profileInfoListOk :{
	{
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profilePolicyRules { ppr1, ppr2 },
	hriServerAddress #TEST_HRI_ADDRESS3
	}
	}

R_GET_UPDATE_N15	resp ProfileInfoListResponse ::= profileInfoListOk :{

Name	Content

	{
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profilePolicyRules { ppr1, ppr2 },
	lprConfiguration {
	pcmpAddress #TEST_PCMP_ADDRESS3
	}
	}
	}

R_GET_UPDATE_N16	resp ProfileInfoListResponse ::=
	profileInfoListOk :{
	{
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profilePolicyRules { ppr1, ppr2
	}
	}

R_GET_UPDATE_N17	resp ProfileInfoListResponse ::=
	profileInfoListOk :{
	{
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profilePolicyRules { ppr1, ppr2 },
	DeviceChangeConfiguration {
	requestToDp {
	smdpAddressToBeUsedForDc
	#TEST_DP_ADDRESS2
	}
	}
	}
	}

R_GET_UPDATE_N18	resp ProfileInfoListResponse ::=
	profileInfoListOk :{
	{
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profilePolicyRules { ppr1, ppr2 }
	}
	}

R_GET_UPDATE_N19	resp ProfileInfoListResponse ::=
	profileInfoListOk :{
	{
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,

Name	Content

	iconType png,
	icon #ICON_OP_PROF1,
	profilePolicyRules { ppr1, ppr2 },
	DeviceChangeConfiguration {
	usingStoredAc {
	activationCodeForDc
	ACTIVATION_CODE_2
	}
	}
	}
	
	}

R_GET_UPDATE_N20	resp ProfileInfoListResponse ::=
	profileInfoListOk :{
	{
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profilePolicyRules { ppr1, ppr2 }
	}
	
	}

R_ISDR_SELECTION	resp ISDRProprietaryApplicationTemplate::= {
	lowestSvn #RSP_VERSION_LOWEST_H,
	euiccConfiguration {}
	}

R_ISDR_SELECTION_EN_PROF	resp ISDRProprietaryApplicationTemplate::= {
	lowestSvn #RSP_VERSION_LOWEST_H,
	euiccConfiguration {
	...,
	enabledProfile
	}
	}

R_ISDR_SELECTION_LPAE	resp ISDRProprietaryApplicationTemplate::= {
	lowestSvn #RSP_VERSION_LOWEST_H,
	euiccConfiguration {
	lpaeUsingCatSupported
	OR
	lpaeUsingScwsSupported
	}
	}

R_LIST_NOTIF_DE1	response ListNotificationResponse ::=
	notificationMetadataList : {
	#NOTIF_METADATA_DELETE1
	}

R_LIST_NOTIF_DE1_DE1	response ListNotificationResponse ::=
	notificationMetadataList : {
	#NOTIF_METADATA_DELETE1,
	#NOTIF_METADATA2_DELETE1
	}

R_LIST_NOTIF_DE1_DE2	response ListNotificationResponse ::= notificationMetadataList : { #NOTIF_METADATA_DELETE1,

Name	Content
	#NOTIF_METADATA_DELETE2 }

R_LIST_NOTIF_DI1	response ListNotificationResponse ::=
	notificationMetadataList : {
	#NOTIF_METADATA_DISABLE1
	}

R_LIST_NOTIF_DI1_DE1	response ListNotificationResponse ::=
	notificationMetadataList : {
	#NOTIF_METADATA_DISABLE1,
	#NOTIF_METADATA_DELETE1
	}

R_LIST_NOTIF_DI1_DI1	response ListNotificationResponse ::=
	notificationMetadataList : {
	#NOTIF_METADATA_DISABLE1,
	#NOTIF_METADATA2_DISABLE1
	}

R_LIST_NOTIF_DI1_EN2	response ListNotificationResponse ::=
	notificationMetadataList : {
	#NOTIF_METADATA_DISABLE1,
	#NOTIF_METADATA_ENABLE2
	}

R_LIST_NOTIF_EN1	response ListNotificationResponse ::=
	notificationMetadataList: {
	#NOTIF_METADATA_ENABLE1
	}

R_LIST_NOTIF_EN1_EN1	response ListNotificationResponse ::=
	notificationMetadataList : {
	#NOTIF_METADATA_ENABLE1,
	#NOTIF_METADATA2_ENABLE1
	}

R_LIST_NOTIF_EN1_IN2_PIR	response ListNotificationResponse ::=
	notificationMetadataList: {
	#NOTIF_METADATA_ENABLE1,
	#NOTIF_METADATA_INSTALL2_PIR
	}

R_LIST_NOTIF_EN1_RPM	response ListNotificationResponse ::=
	notificationMetadataList: {
	#NOTIF_METADATA_EN1_RPM
	}

R_LIST_NOTIF_EN1_RPM_DP2	response ListNotificationResponse ::=
	notificationMetadataList: {
	#NOTIF_METADATA_DP2_EN1_RPM
	}

R_LIST_NOTIF_IN1	response ListNotificationResponse ::=
	notificationMetadataList: {
	#NOTIF_METADATA_INSTALL1
	}

R_LIST_NOTIF_IN1_DP1_PIR_IN1_DP2_OSN	response ListNotificationResponse ::=
	notificationMetadataList: {
	#NOTIF_METADATA_INSTALL1_DP1_PIR,
	#NOTIF_METADATA_INSTALL1_DP2_OSN
	}

R_LIST_NOTIF_IN1_EN1	response ListNotificationResponse ::= notificationMetadataList: { #NOTIF_METADATA_INSTALL1, #NOTIF_METADATA_ENABLE1

Name	Content
	}

R_LIST_NOTIF_IN1_IN1_PIR	response ListNotificationResponse ::=
	notificationMetadataList: {
	#NOTIF_METADATA_INSTALL1,
	#NOTIF_METADATA_INSTALL1_PIR
	}

R_LIST_NOTIF_IN1_PIR	response ListNotificationResponse ::=
	notificationMetadataList: {
	#NOTIF_METADATA_INSTALL1_PIR
	}

R_LIST_NOTIF_IN1_PIR_EN1	response ListNotificationResponse ::=
	notificationMetadataList: {
	#NOTIF_METADATA_INSTALL1_PIR,
	#NOTIF_METADATA_ENABLE1
	}

R_LIST_NOTIF_IN1_PIR_IN2_PIR	response ListNotificationResponse ::=
	notificationMetadataList: {
	#NOTIF_METADATA_INSTALL1_PIR,
	#NOTIF_METADATA_INSTALL2_PIR
	}

R_LIST_NOTIF_IN2_PIR	response ListNotificationResponse ::=
	notificationMetadataList: {
	#NOTIF_METADATA_INSTALL2_PIR
	}

R_LIST_NOTIF_NONE	response ListNotificationResponse ::= notificationMetadataList: {}

R_LIST_NOTIF_RPR	response ListNotificationResponse ::=
	notificationMetadataList: {
	#NOTIF_METADATA_PROF1_DP1_RPR
	}

R_LIST_NOTIF_RPR_EN1_RPM_DP2	response ListNotificationResponse ::=
	notificationMetadataList: {
	#NOTIF_METADATA_PROF1_DP1_RPR,
	#NOTIF_METADATA_DP2_EN1_RPM
	}

R_LIST_NOTIF_UNDEFINED_ERROR	response ListNotificationResponse ::= listNotificationsResultError : undefinedError

R_METADATA_UNCHANGED	resp ProfileInfoListResponse ::=
	profileInfoListOk :{
	{
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profilePolicyRules {ppr1,ppr2}
	}
	}

R_PIR_DATA_MISMATCH	resp ProfileInstallationResult ::= {
	profileInstallationResultData {
	transactionId <S_TRANSACTION_ID>,
	notificationMetadata {
	seqNumber <SEQ_NUMBER>,
	profileManagementOperation {
	notificationInstall
	},

Name	Content

	notificationAddress #TEST_DP_ADDRESS1,
	...
	},
	smdpOid #S_SM_DP+_OID,
	finalResult errorResult : {
	bppCommandId loadProfileElements,
	errorReason
	installFailedDueToDataMismatch,
	...
	}
	},
	euiccSignPIR <EUICC_SIGN_PIR>
	}

R_PIR_EOID_MISMATCH	resp ProfileInstallationResult ::= {
	profileInstallationResultData {
	transactionId <S_TRANSACTION_ID>,
	notificationMetadata {
	seqNumber <SEQ_NUMBER>,
	profileManagementOperation {
	notificationInstall
	},
	notificationAddress #TEST_DP_ADDRESS1
	},
	smdpOid #S_SM_DP+_OID,
	finalResult errorResult : {
	bppCommandId storeMetadata,
	errorReason
	enterpriseOidMismatch
	}
	},
	euiccSignPIR <EUICC_SIGN_PIR>
	}

R_PIR_EP_NOTALLOWED	resp ProfileInstallationResult ::= {
	profileInstallationResultData {
	transactionId <S_TRANSACTION_ID>,
	notificationMetadata {
	seqNumber <SEQ_NUMBER>,
	profileManagementOperation {
	notificationInstall
	},
	notificationAddress #TEST_DP_ADDRESS1
	},
	smdpOid #S_SM_DP+_OID,
	finalResult errorResult : {
	bppCommandId storeMetadata,
	errorReason
	enterpriseProfileNotAllowed
	}
	},
	euiccSignPIR <EUICC_SIGN_PIR>
	}

R_PIR_EP_ONLY	resp ProfileInstallationResult ::= {
	profileInstallationResultData {
	transactionId <S_TRANSACTION_ID>,
	notificationMetadata {
	seqNumber <SEQ_NUMBER>,
	profileManagementOperation {

Name	Content

	notificationInstall
	},
	notificationAddress #TEST_DP_ADDRESS1
	},
	smdpOid #S_SM_DP+_OID,
	finalResult errorResult : {
	bppCommandId storeMetadata,
	errorReason
	enterpriseProfilesOnly
	}
	},
	euiccSignPIR <EUICC_SIGN_PIR>
	}

R_PIR_ER_NOTALLOWED	resp ProfileInstallationResult ::= {
	profileInstallationResultData {
	transactionId <S_TRANSACTION_ID>,
	notificationMetadata {
	seqNumber <SEQ_NUMBER>,
	profileManagementOperation {
	notificationInstall
	},
	notificationAddress #TEST_DP_ADDRESS1
	},
	smdpOid #S_SM_DP+_OID,
	finalResult errorResult : {
	bppCommandId storeMetadata,
	errorReason
	enterpriseRulesNotAllowed
	}
	},
	euiccSignPIR <EUICC_SIGN_PIR>
	}

R_PIR_ICCID_ALREADY_EXIST	resp ProfileInstallationResult ::= {
	profileInstallationResultData {
	transactionId <S_TRANSACTION_ID>,
	notificationMetadata {
	seqNumber <SEQ_NUMBER>,
	profileManagementOperation {
	notificationInstall
	},
	notificationAddress #TEST_DP_ADDRESS1,
	iccid #ICCID_OP_PROF1
	},
	smdpOid #S_SM_DP+_OID,
	finalResult errorResult : {
	bppCommandId storeMetadata,
	errorReason
	
	installFailedDueToIccidAlreadyExistsOnEuicc
	}
	},
	euiccSignPIR <EUICC_SIGN_PIR>
	}

R_PIR_INVALID_CRT	resp ProfileInstallationResult ::= {
	profileInstallationResultData {
	transactionId <S_TRANSACTION_ID>,
	notificationMetadata {

Name	Content

	seqNumber <SEQ_NUMBER>,
	profileManagementOperation {
	notificationInstall
	},
	notificationAddress #TEST_DP_ADDRESS1
	},
	smdpOid #S_SM_DP+_OID,
	finalResult errorResult : {
	bppCommandId initialiseSecureChannel,
	errorReason unsupportedCrtValues
	}
	},
	euiccSignPIR <EUICC_SIGN_PIR>
	}

R_PIR_INVALID_DATA	resp ProfileInstallationResult ::= {
	profileInstallationResultData {
	transactionId <S_TRANSACTION_ID>,
	notificationMetadata {
	seqNumber <SEQ_NUMBER>,
	profileManagementOperation {
	notificationInstall
	},
	notificationAddress #TEST_DP_ADDRESS1
	},
	smdpOid #S_SM_DP+_OID,
	finalResult errorResult : {
	bppCommandId configureISDP,
	errorReason incorrectInputValues
	}
	},
	euiccSignPIR <EUICC_SIGN_PIR>
	}

R_PIR_INVALID_OP_ID	resp ProfileInstallationResult ::= {
	profileInstallationResultData {
	transactionId <S_TRANSACTION_ID>,
	notificationMetadata {
	seqNumber <SEQ_NUMBER>,
	profileManagementOperation {
	notificationInstall
	},
	notificationAddress #TEST_DP_ADDRESS1
	},
	smdpOid #S_SM_DP+_OID,
	finalResult errorResult : {
	bppCommandId initialiseSecureChannel,
	errorReason
	unsupportedRemoteOperationType
	}
	},
	euiccSignPIR <EUICC_SIGN_PIR>
	}

R_PIR_INVALID_SIGN	resp ProfileInstallationResult ::= {
	profileInstallationResultData {
	transactionId <S_TRANSACTION_ID>,
	notificationMetadata {
	seqNumber <SEQ_NUMBER>,
	profileManagementOperation {

Name	Content

	notificationInstall
	},
	notificationAddress #TEST_DP_ADDRESS1
	},
	smdpOid #S_SM_DP+_OID,
	finalResult errorResult : {
	bppCommandId initialiseSecureChannel,
	errorReason invalidSignature
	}
	},
	euiccSignPIR <EUICC_SIGN_PIR>
	}

R_PIR_INVALID_TRANS_ID	resp ProfileInstallationResult ::= {
	profileInstallationResultData {
	transactionId <INVALID_TRANSACTION_ID>,
	notificationMetadata {
	seqNumber <SEQ_NUMBER>,
	profileManagementOperation {
	notificationInstall
	},
	notificationAddress #TEST_DP_ADDRESS1
	},
	smdpOid #S_SM_DP+_OID,
	finalResult errorResult : {
	bppCommandId initialiseSecureChannel,
	errorReason invalidTransactionId
	}
	},
	euiccSignPIR <EUICC_SIGN_PIR>
	}

R_PIR_METADATA_INVALID	resp ProfileInstallationResult ::= {
	profileInstallationResultData {
	transactionId <S_TRANSACTION_ID>,
	notificationMetadata {
	seqNumber <SEQ_NUMBER>,
	profileManagementOperation {
	notificationInstall
	},
	notificationAddress #TEST_DP_ADDRESS1,
	...
	},
	smdpOid #S_SM_DP+_OID,
	finalResult errorResult : {
	bppCommandId storeMetadata,
	errorReason
	bspStructureError
	OR
	incorrectInputValues
	}
	},
	euiccSignPIR <EUICC_SIGN_PIR>
	}

R_PIR_OK	response ProfileInstallationResult ::= {
	profileInstallationResultData {
	transactionId <S_TRANSACTION_ID>,
	notificationMetadata {
	seqNumber <SEQ_NUMBER>,

Name	Content

	profileManagementOperation {
	notificationInstall
	},
	notificationAddress #TEST_DP_ADDRESS1,
	iccid #ICCID_OP_PROF1
	},
	smdpOid #S_SM_DP+_OID,
	finalResult successResult : {
	aid <ISD_P_AID>,
	simaResponse #SIMA_RESULT_OK
	}
	},
	euiccSignPIR <EUICC_SIGN_PIR>
	}

R_PIR_OK_PROF9	response ProfileInstallationResult ::= {
	profileInstallationResultData {
	transactionId <S_TRANSACTION_ID>,
	notificationMetadata {
	seqNumber <SEQ_NUMBER>,
	profileManagementOperation {
	notificationInstall
	},
	notificationAddress #TEST_DP_ADDRESS1,
	iccid #ICCID_OP_PROF9
	},
	smdpOid #S_SM_DP+_OID,
	finalResult successResult : {
	aid <ISD_P_AID>,
	simaResponse #SIMA_RESULT_OK
	}
	},
	euiccSignPIR <EUICC_SIGN_PIR>
	}

R_PIR_OK_PROF10	response ProfileInstallationResult ::= {
	profileInstallationResultData {
	transactionId <S_TRANSACTION_ID>,
	notificationMetadata {
	seqNumber <SEQ_NUMBER>,
	profileManagementOperation {
	notificationInstall
	},
	notificationAddress #TEST_DP_ADDRESS1,
	iccid #ICCID_OP_PROF10
	},
	smdpOid #S_SM_DP+_OID,
	finalResult successResult : {
	aid <ISD_P_AID>,
	simaResponse #SIMA_RESULT_OK
	}
	},
	euiccSignPIR <EUICC_SIGN_PIR>
	}

R_PIR_PPK_INV	resp ProfileInstallationResult ::= {
	profileInstallationResultData {
	...
	finalResult errorResult : {
	bppCommandId replaceSessionKeys,

Name	Content
	errorReason incorrectInputValues OR bspStructureError OR bspSecurityError } }, euiccSignPIR <EUICC_SIGN_PIR> }

R_PIR_PPR_NOT_ALLOWED	resp ProfileInstallationResult ::= {
	profileInstallationResultData {
	transactionId <S_TRANSACTION_ID>,
	notificationMetadata {
	seqNumber <SEQ_NUMBER>,
	profileManagementOperation {
	notificationInstall
	},
	notificationAddress #TEST_DP_ADDRESS1,
	...
	},
	smdpOid #S_SM_DP+_OID,
	finalResult errorResult : {
	bppCommandId storeMetadata,
	errorReason pprNotAllowed
	}
	},
	euiccSignPIR <EUICC_SIGN_PIR>
	}

R_PIR_RER_NOTALLOWED	resp ProfileInstallationResult ::= {
	profileInstallationResultData {
	transactionId <S_TRANSACTION_ID>,
	notificationMetadata {
	seqNumber <SEQ_NUMBER>,
	profileManagementOperation {
	notificationInstall
	},
	notificationAddress #TEST_DP_ADDRESS1
	},
	smdpOid #S_SM_DP+_OID,
	finalResult errorResult : {
	bppCommandId storeMetadata,
	errorReason
	enterpriseRulesError
	}
	},
	euiccSignPIR <EUICC_SIGN_PIR>
	}

R_PIR_SECU_INVALID	resp ProfileInstallationResult ::= {
	profileInstallationResultData {
	transactionId <S_TRANSACTION_ID>,
	...
	smdpOid #S_SM_DP+_OID,
	finalResult errorResult : {
	bppCommandId loadProfileElements,
	errorReason incorrectInputValues
	OR

Name	Content
	bspStructureError OR bspSecurityError ... } }, euiccSignPIR <EUICC_SIGN_PIR> }

R_PIR_UNKNOWN_TLV	resp ProfileInstallationResult ::= {
	profileInstallationResultData {
	transactionId <S_TRANSACTION_ID>,
	notificationMetadata {
	seqNumber <SEQ_NUMBER>,
	profileManagementOperation {
	notificationInstall
	},
	notificationAddress #TEST_DP_ADDRESS1
	},
	smdpOid #S_SM_DP+_OID,
	finalResult errorResult : {
	bppCommandId storeMetadata,
	errorReason
	unknownTlvInMetadata
	}
	},
	euiccSignPIR <EUICC_SIGN_PIR>
	}

R_PREP_DOWN_INV_TRANS_ID	resp PrepareDownloadResponse ::=
	downloadResponseError : {
	transactionId <INVALID_TRANSACTION_ID>,
	downloadErrorCode invalidTransactionId
	}

R_PREP_DOWN_NO_SESSION	resp PrepareDownloadResponse ::=
	downloadResponseError : {
	transactionId <S_TRANSACTION_ID>,
	downloadErrorCode noSession
	}

R_PREP_DOWNLOAD_INV_CERT	resp PrepareDownloadResponse ::=
	downloadResponseError : {
	transactionId <S_TRANSACTION_ID>,
	downloadErrorCode invalidCertificate
	}

R_PREP_DOWNLOAD_INV_SIGN	resp PrepareDownloadResponse ::=
	downloadResponseError : {
	transactionId <S_TRANSACTION_ID>,
	downloadErrorCode invalidSignature
	}

R_PREP_DOWNLOAD_NO_CC	resp PrepareDownloadResponse ::=
	downloadResponseOk : {
	euiccSigned2 {
	transactionId <S_TRANSACTION_ID>,
	euiccOtpk <OTPK_EUICC_ECKA>
	},
	euiccSignature2 <EUICC_SIGNATURE2>
	}

R_PREP_DOWNLOAD_WITH_CC	resp PrepareDownloadResponse ::= downloadResponseOk : {

Name	Content

	euiccSigned2 {
	transactionId <S_TRANSACTION_ID>,
	euiccOtpk <OTPK_EUICC_ECKA>,
	hashCc <S_HASHED_CC>
	},
	euiccSignature2 <EUICC_SIGNATURE2>
	}

R_RAT_WITH_OTHER_RULES	response GetRatResponse ::= {
	rat {
	#PPR1_WITH_OWNER_GID,
	#PPR1_WITHOUT_GID,
	#PPR2_WITHOUT_CONSENT,
	#PPRS_ALLOWED
	}
	}

R_REMOVE_NOTIF_NOTHING_TO_DELETE	response NotificationSentResponse ::= {
	deleteNotificationStatus nothingToDelete
	}

R_REMOVE_NOTIF_OK	response NotificationSentResponse ::= {
	deleteNotificationStatus ok
	}

R_RETRIEVE_NOTIF_DE1_V3	resp RetrieveNotificationsListResponse ::=
	notificationList : {
	otherSignedNotification : {
	tbsOtherNotification
	#NOTIF_METADATA_DELETE1,
	euiccNotificationSignature
	<TBS_EUICC_NOTIF_SIG>,
	euiccCertificate #CERT_EUICC_SIG,
	nextCertInChain #CERT_EUM_SIG
	}
	}

R_RETRIEVE_NOTIF_DI1_V3	resp RetrieveNotificationsListResponse ::=
	notificationList : {
	otherSignedNotification : {
	tbsOtherNotification
	#NOTIF_METADATA_DISABLE1,
	euiccNotificationSignature
	<TBS_EUICC_NOTIF_SIG>,
	euiccCertificate #CERT_EUICC_SIG,
	nextCertInChain #CERT_EUM_SIG
	}
	}

R_RETRIEVE_NOTIF_DI1_DE1_V3	resp RetrieveNotificationsListResponse ::=
	notificationList : {
	otherSignedNotification : {
	tbsOtherNotification
	#NOTIF_METADATA_DISABLE1,
	euiccNotificationSignature
	<TBS_EUICC_NOTIF_SIG>,
	euiccCertificate #CERT_EUICC_SIG,
	nextCertInChain #CERT_EUM_SIG
	},
	otherSignedNotification : {
	tbsOtherNotification
	#NOTIF_METADATA_DELETE1,
	euiccNotificationSignature

Name	Content
	<TBS_EUICC_NOTIF_SIG>, euiccCertificate #CERT_EUICC_SIG, nextCertInChain #CERT_EUM_SIG } }

R_RETRIEVE_NOTIF_EN1_V3	resp RetrieveNotificationsListResponse ::=
	notificationList : {
	otherSignedNotification : {
	tbsOtherNotification
	#NOTIF_METADATA_ENABLE1,
	euiccNotificationSignature
	<TBS_EUICC_NOTIF_SIG>,
	euiccCertificate #CERT_EUICC_SIG,
	nextCertInChain #CERT_EUM_SIG
	}
	}

R_RETRIEVE_NOTIF_EN1_IN2_PIR_V3	resp RetrieveNotificationsListResponse ::=
	notificationList : {
	profileInstallationResult : {
	profileInstallationResultData {
	transactionId <S_TRANSACTION_ID>,
	notificationMetadata
	#NOTIF_METADATA_INSTALL2_PIR,
	smdpOid #S_SM_DP+_OID2,
	finalResult successResult : {
	aid <ISD_P_AID>,
	simaResponse #SIMA_RESULT_OK
	}
	},
	euiccSignPIR <EUICC_SIGN_PIR>
	},
	otherSignedNotification : {
	
	tbsOtherNotification#NOTIF_METADATA_ENABLE1,
	euiccNotificationSignature
	<TBS_EUICC_NOTIF_SIG>,
	euiccCertificate #CERT_EUICC_SIG,
	nextCertInChain #CERT_EUM_SIG
	}
	}

R_RETRIEVE_NOTIF_IN1_V3	resp RetrieveNotificationsListResponse ::=
	notificationList : {
	otherSignedNotification : {
	tbsOtherNotification
	#NOTIF_METADATA_INSTALL1,
	euiccNotificationSignature
	<TBS_EUICC_NOTIF_SIG>,
	euiccCertificate #CERT_EUICC_SIG,
	nextCertInChain #CERT_EUM_SIG
	}
	}

R_RETRIEVE_NOTIF_IN1_DP1_PIR	resp RetrieveNotificationsListResponse ::=
	notificationList : {
	profileInstallationResult : {
	profileInstallationResultData {
	transactionId <S_TRANSACTION_ID>,

Name	Content

	notificationMetadata #NOTIF_METADATA_INSTALL1_DP1_PIR,
	smdpOid #S_SM_DP+_OID1,
	finalResult successResult : {
	aid <ISD_P_AID>,
	simaResponse #SIMA_RESULT_OK
	}
	},
	euiccSignPIR <EUICC_SIGN_PIR>
	}
	}

R_RETRIEVE_NOTIF_IN1_DP1_PIR_IN1_DP2_OSN_V3	resp RetrieveNotificationsListResponse ::=
	notificationList : {
	profileInstallationResult : {
	profileInstallationResultData {
	transactionId <S_TRANSACTION_ID>,
	notificationMetadata #NOTIF_METADATA_INSTALL1_DP1_PIR,
	smdpOid #S_SM_DP+_OID1,
	finalResult successResult : {
	aid <ISD_P_AID>,
	simaResponse #SIMA_RESULT_OK
	}
	},
	euiccSignPIR <EUICC_SIGN_PIR>
	},
	otherSignedNotification : {
	tbsOtherNotification #NOTIF_METADATA_INSTALL1_DP2_OSN,
	euiccNotificationSignature <TBS_EUICC_NOTIF_SIG>,
	euiccCertificate #CERT_EUICC_SIG,
	nextCertInChain #CERT_EUM_SIG
	}
	}

R_RETRIEVE_NOTIF_IN1_DP2_OSN_V3	resp RetrieveNotificationsListResponse ::=
	notificationList : {
	otherSignedNotification : {
	tbsOtherNotification #NOTIF_METADATA_INSTALL1_DP2_OSN,
	euiccNotificationSignature <TBS_EUICC_NOTIF_SIG>,
	euiccCertificate #CERT_EUICC_SIG,
	nextCertInChain #CERT_EUM_SIG
	}
	}

R_RETRIEVE_NOTIF_IN1_EN1_V3	resp RetrieveNotificationsListResponse ::=
	notificationList : {
	otherSignedNotification : {
	tbsOtherNotification #NOTIF_METADATA_INSTALL1,
	euiccNotificationSignature <TBS_EUICC_NOTIF_SIG>,
	euiccCertificate #CERT_EUICC_SIG,
	nextCertInChain #CERT_EUM_SIG
	},
	otherSignedNotification : {

Name	Content
	tbsOtherNotification #NOTIF_METADATA_ENABLE1, euiccNotificationSignature <TBS_EUICC_NOTIF_SIG>, euiccCertificate #CERT_EUICC_SIG, nextCertInChain #CERT_EUM_SIG } }

R_RETRIEVE_NOTIF_IN1_IN1_PIR	resp RetrieveNotificationsListResponse ::=
	notificationList : {
	profileInstallationResult : {
	profileInstallationResultData {
	transactionId <S_TRANSACTION_ID>,
	notificationMetadata #NOTIF_METADATA_INSTALL1_PIR,
	smdpOid #S_SM_DP+_OID,
	finalResult successResult : {
	aid <ISD_P_AID>,
	simaResponse #SIMA_RESULT_OK
	}
	},
	euiccSignPIR <EUICC_SIGN_PIR>
	},
	otherSignedNotification : {
	tbsOtherNotification #NOTIF_METADATA_INSTALL1,
	euiccNotificationSignature <TBS_EUICC_NOTIF_SIG>,
	euiccCertificate #CERT_EUICC_SIG,
	nextCertInChain #CERT_EUM_SIG
	}
	}

R_RETRIEVE_NOTIF_IN1_PIR	resp RetrieveNotificationsListResponse ::=
	notificationList : {
	profileInstallationResult : {
	profileInstallationResultData {
	transactionId <S_TRANSACTION_ID>,
	notificationMetadata #NOTIF_METADATA_INSTALL1_PIR,
	smdpOid #S_SM_DP+_OID,
	finalResult successResult : {
	aid <ISD_P_AID>,
	simaResponse #SIMA_RESULT_OK
	}
	},
	euiccSignPIR <EUICC_SIGN_PIR>
	}
	}

R_RETRIEVE_NOTIF_IN1_PIR_EN1_V3	resp RetrieveNotificationsListResponse ::=
	notificationList : {
	profileInstallationResult : {
	profileInstallationResultData {
	transactionId <S_TRANSACTION_ID>,
	notificationMetadata #NOTIF_METADATA_INSTALL1_PIR,
	smdpOid #S_SM_DP+_OID,
	finalResult successResult : {

Name	Content

	aid <ISD_P_AID>,
	simaResponse #SIMA_RESULT_OK
	}
	},
	euiccSignPIR <EUICC_SIGN_PIR>
	},
	otherSignedNotification : {
	tbsOtherNotification
	#NOTIF_METADATA_ENABLE1,
	euiccNotificationSignature
	<TBS_EUICC_NOTIF_SIG>,
	euiccCertificate #CERT_EUICC_SIG,
	nextCertInChain #CERT_EUM_SIG
	}
	}

R_RETRIEVE_NOTIF_IN1_PIR_IN2_PIR	resp RetrieveNotificationsListResponse ::=
	notificationList : {
	profileInstallationResult : {
	profileInstallationResultData {
	transactionId <S_TRANSACTION_ID>,
	notificationMetadata
	#NOTIF_METADATA_INSTALL1_PIR,
	smdpOid #S_SM_DP+_OID,
	finalResult successResult : {
	aid <ISD_P_AID>,
	simaResponse #SIMA_RESULT_OK
	}
	},
	euiccSignPIR <EUICC_SIGN_PIR>
	},
	profileInstallationResult : {
	profileInstallationResultData {
	transactionId <S_TRANSACTION_ID>,
	notificationMetadata
	#NOTIF_METADATA_INSTALL2_PIR,
	smdpOid #S_SM_DP+_OID2,
	finalResult successResult : {
	aid <ISD_P_AID>,
	simaResponse #SIMA_RESULT_OK
	}
	},
	euiccSignPIR <EUICC_SIGN_PIR>
	}
	}

R_RETRIEVE_NOTIF_IN2_PIR	resp RetrieveNotificationsListResponse ::=
	notificationList : {
	profileInstallationResult : {
	profileInstallationResultData {
	transactionId <S_TRANSACTION_ID>,
	notificationMetadata
	#NOTIF_METADATA_INSTALL2_PIR,
	smdpOid #S_SM_DP+_OID2,
	finalResult successResult : {
	aid <ISD_P_AID>,
	simaResponse #SIMA_RESULT_OK
	}

Name	Content
	}, euiccSignPIR <EUICC_SIGN_PIR> } }
R_RETRIEVE_NOTIF_NONE	resp RetrieveNotificationsListResponse ::= notificationList : {}

SMDP_PROP_DATA1
(DpProprietaryData)
	{
	dpOid #S_SM_DP+_OID
	}

D.4 APDU

D.4.1 APDU Commands

Name	Content
DELETE_SSD	- CLA = 80, INS = E4, P1 = 00, P2 = 80, LC = <L> - Data = 4F <L> #SSD_AID - LE = 00
GET_MNO_SD	- CLA = 80, INS = F2, P1 = 80, P2 = 02, LC = <L> - Data = 4F 00 - LE = 00
GET_RESPONSE	- CLA = 0x (x = <CHANNEL_NUMBER>), INS = C0, P1 = 00, P2 = 00, LE = <L>
INSTALL_PERSO_RES_ISDP	- CLA = 80, INS = E6, P1 = 20, P2 = 00, LC = 16 - Data = 00 00 10 A0 00 00 05 59 10 10 FF FF FF FF 89 00 00 0F 00 00 00 00 - LE = 00
MANAGE_CHANNEL_OPEN	- CLA = 00, INS = 70, P1 = 00, P2 = 00, LE = 01
MANAGE_LSI(Select LSI)	- CLA = 80, INS = 7C, P1 = 00, P2 = <LSI_NUMBER>, LE = 00
READ_BINARY	- CLA = 00, INS = B0, P1 = 00, P2 = 00, LE = <L>
SELECT_ICCID	- CLA = 00, INS = A4, P1 = 00, P2 = 0C, LC = 02 - Data = 2F E2
SELECT_MF	- CLA = 00, INS = A4, P1 = 00, P2 = 04, LC = <L> - Data = 3F 00 - LE = 00
SELECT_USIM	- CLA = 00, INS = A4, P1 = 04, P2 = 04, LC = <L> - Data = #USIM_AID - LE = 00
TERMINAL_CAPABILITY_Enterprise	- CLA = 80, INS = AA, P1 = 00, P2 = 00, LC = <L> - Data = A9 05 81 00 83 01 27
TERMINAL_CAPABILITY_LPA_Alerting	- CLA = 80, INS = AA, P1 = 00, P2 = 00, LC = <L> - Data = A9 05 81 00 83 01 17
TERMINAL_CAPABILITY_LPAd	- CLA = 80, INS = AA, P1 = 00, P2 = 00, LC = <L> - Data = A9 05 81 00 83 01 07
TERMINAL_PROFILE	- CLA = 80, INS = 10, P1 = 00, P2 = 00, LC = <L> - Data = FF FF FF FF 7F 9D 00 DF BF 00 00 1F E2 00 00 00 C7 EB 00 00 00 01 68 00 50 00 00 00 00 00 02 00
TERMINAL_PROFILE_eUICCProfileStateChanged	- CLA = 80, INS = 10, P1 = 00, P2 = 00, LC = <L> - Data = FF FF FF FF FF FF 1F FF FF 03 02 FF FF 9F FF EF DF FF 0F FF 0F FF FF 0F FF 03 00 3F 7F FF 03 FF FF 20
TERMINAL_PROFILE_LSI_COMMAND	- CLA = 80, INS = 10, P1 = 00, P2 = 00, LC = <L>

	- Data = FF FF FF FF FF FF 1F FF FF 03 02 FF FF 9F FF EF DF FF 0F FF 0F FF FF 0F FF 03 00 3F 7F FF 03 FF FF 00 30 00
TERMINAL_PROFILE_LSI_COMMAND_ eUICCProfileStateChanged	- CLA = 80, INS = 10, P1 = 00, P2 = 00, LC = <L> - Data = FF FF FF FF FF FF 1F FF FF 03 02 FF FF 9F FF EF DF FF 0F FF 0F FF FF 0F FF 03 00 3F 7F FF 03 FF FF 20 30 00

D.4.2 R-APDU Chaining

During the execution of all sequences related to the eUICC testing (i.e. section 4.2), for commands where the response exceeds 256 bytes, the chaining mechanism defined in ISO/IEC 7816-4 [7], using the 61XX status word and multiple GET RESPONSE commands, SHALL be used.

As an example, the following generic sequence, which describes this mechanism, SHALL apply.

Step	Direction	Sequence / Description	Result
1	OCE → eUICC	Send APDU command on logical channel x	<R_APDU_PART1> SW=0x61XX
2	OCE → eUICC	Send [GET_RESPONSE] on logical channel x with LE='XX'	<R_APDU_PART2> SW=0x61XX
3	OCE → eUICC	Send [GET_RESPONSE] on logical channel x with LE='XX'	<R_APDU_PART3> SW=0x61XX
4	OCE → eUICC	Send [GET_RESPONSE] on logical channel x with LE='XX'	<R_APDU_PART4> SW=0x9000 The complete response is the result of the concatenation of all R-APDUs from <R_APDU_PART1> to <R_APDU_PART4>

D.5 ES6 Requests And Responses

D.5.1 ES6 Requests

Name	Content

METADATA_WITH_PPRS_AND_ICON	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	profilePolicyRules {ppr1,ppr2}
	}

METADATA_WITH_PPRS_ICON_AND_NOTIF	metadataReq StoreMetadataRequest ::= { iccid #ICCID_OP_PROF1, serviceProviderName #SP_NAME1, profileName #NAME_OP_PROF1, iconType png,

Name	Content

	icon #ICON_OP_PROF1,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	notificationConfigurationInfo {
	{ profileManagementOperation {
	notificationInstall,
	notificationLocalEnable,
	notificationLocalDisable,
	notificationLocalDelete
	},
	notificationAddress #TEST_DP_ADDRESS1
	}
	},
	profilePolicyRules {ppr1,ppr2}
	}

METADATA_WITH_PPRS_ICON_AND_SPEC_DATA	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	serviceSpecificDataStoredInEuicc #VENDOR_SPECIFIC_EXTENSION1,
	profilePolicyRules {ppr1,ppr2}
	}

METADATA_WITH_PPRS_ICON_AND_RPM_CONFIG	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	rpmConfiguration #RPM_CONFIG_OP_PROF1,
	profilePolicyRules {ppr1,ppr2}
	}

METADATA_WITH_PPRS_ICON_AND_HRI_ADDR	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	profilePolicyRules {ppr1,ppr2},
	hriServerAddress #TEST_HRI_ADDRESS1
	}

METADATA_WITH_PPRS_ICON_AND_LPR_CONFIG	metadataReq StoreMetadataRequest ::= { iccid #ICCID_OP_PROF1, serviceProviderName #SP_NAME1, profileName #NAME_OP_PROF1,

Name	Content

	iconType png,
	icon #ICON_OP_PROF1,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	profilePolicyRules {ppr1,ppr2},
	lprConfiguration {
	pcmpAddress #TEST_PCMP_ADDRESS1
	triggerLprOnEnableProfile
	}
	}

METADATA_WITH_PPRS_ICON_AND_DC_CONFIG_DP	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	profilePolicyRules {ppr1,ppr2},
	deviceChangeConfiguration {
	requestToDp {
	smdpAddressToBeUsedForDc
	#TEST_DP_ADDRESS1
	eidRequired
	}
	}
	}

METADATA_WITH_PPRS_ICON_AND_DC_CONFIG_AC	metadataReq StoreMetadataRequest ::= {
	iccid #ICCID_OP_PROF1,
	serviceProviderName #SP_NAME1,
	profileName #NAME_OP_PROF1,
	iconType png,
	icon #ICON_OP_PROF1,
	profileOwner {
	mccMnc #MCC_MNC1
	},
	profilePolicyRules {ppr1,ppr2},
	deviceChangeConfiguration {
	usingStoredAc
	{
	activationCodeForDc ACTIVATION_CODE_1
	deleteOldProfile
	}
	}
	}

REMOVE_PPR1	metadataReq UpdateMetadataRequest ::= {
	profilePolicyRules {ppr2}
	}

UPD_ENT_CONFIG1	metadataReq UpdateMetadataRequest ::= {
	enterpriseConfiguration {
	enterpriseOid #S_ENTERPRISE_OID,
	enterpriseName #ENTERPRISE_NAME1,
	enterpriseRules {
	enterpriseRuleBits {
	referenceEnterpriseRule,

Name	Content
	priorityEnterpriseProfile, onlyEnterpriseProfilesCanBeInstalled }, numberOfNonEnterpriseProfiles 0 } }

UPD_ICON_REM_PPR2	metadataReq UpdateMetadataRequest ::= {
	iconType jpg,
	icon #ICON_JPG,
	profilePolicyRules {ppr1}
	}

UPD_NAMES_REM_PPRS_V3	metadataReq UpdateMetadataRequest ::= {
	serviceProviderName #SP_NAME2,
	profileName #NAME_OP_PROF2,
	tagsForDeletion '99'H
	
	}

UPD_NO_METADATA	metadataReq UpdateMetadataRequest ::= { }

UPD_PPR_CONTROL	metadataReq UpdateMetadataRequest ::= {
	serviceProviderName #SP_NAME2,
	profileName #NAME_OP_PROF2,
	iconType jpg,
	icon #ICON_JPG,
	profilePolicyRules {pprUpdateControl, ppr1}
	}

UPD_WITH_EC	metadataReq UpdateMetadataRequest ::= {
	serviceProviderName #SP_NAME2,
	
	enterpriseConfiguration {
	enterpriseOid #S_ENTERPRISE_OID
	enterpriseName #ENTERPRISE_NAME1
	
	}
	}

UPD_NAMES_REM_ICON_REM_PPRS_V3	metadataReq UpdateMetadataRequest ::= {
	serviceProviderName #SP_NAME2,
	profileName #NAME_OP_PROF2,
	tagsForDeletion { iconType, icon,
	{profilePolicyRules {ppr1, ppr2} }
	}

UPD_NOTIF_CONFIG_INFO	metadataReq UpdateMetadataRequest ::= {
	
	notificationConfigurationInfo {
	{
	profileManagementOperation {
	notificationLocalDelete
	},
	notificationAddress #TEST_DP_ADDRESS2
	}
	}
	}

REM_NOTIF_CONFIG_INFO	metadataReq UpdateMetadataRequest ::= {
	
	tagsForDeletion {
	'B6'
	}
	}

Name	Content

UPD_SPEC_DATA	metadataReq UpdateMetadataRequest ::= {
	
	  serviceSpecificDataStoredInEuicc
	#VENDOR_SPECIFIC_EXTENSION2
	}

REM_SPEC_DATA	metadataReq UpdateMetadataRequest ::= { tagsForDeletion 'BF22'H }

UPD_RPM_CONFIG	metadataReq UpdateMetadataRequest ::= {
	
	  rpmConfiguration {
	  managingDpList {},
	  profileOwnerOid #S_PROFILE_OWNER_OID
	  }
	}

REM_ RPM_CONFIG	metadataReq UpdateMetadataRequest ::= {
	
	  tagsForDeletion
	  'BA'H
	}

UPD_HRI_ADDR	metadataReq UpdateMetadataRequest ::= {
	
	  hriServerAddress #TEST_HRI_ADDRESS3
	}

REM_HRI_ADDR	metadataReq UpdateMetadataRequest ::= { tagsForDeletion '9B'H }

UPD_LPR_CONFIG	metadataReq UpdateMetadataRequest ::= {
	
	  lprConfiguration {
	  pcmpAddress #TEST_PCMP_ADDRESS3
	  }
	
	}

REM_ LPR_CONFIG	metadataReq UpdateMetadataRequest ::= { tagsForDeletion 'BC'H }

UPD_DC_CONFIG_DP	metadataReq UpdateMetadataRequest ::= {
	
	  deviceChangeConfiguration {
	  requestToDp {
	  smdpAddressToBeUsedForDc
	#TEST_DP_ADDRESS2
	  }
	  }
	
	}

REM_DC_CONFIG_DP	metadataReq UpdateMetadataRequest ::= { tagsForDeletion 'BF20'H }

UPD_DC_CONFIG_AC	metadataReq UpdateMetadataRequest ::= {
	
	  deviceChangeConfiguration {
	  usingStoredAc
	  {
	  activationCodeForDc ACTIVATION_CODE_2
	  }

Name	Content
	} }
REM_DC_CONFIG_AC	metadataReq UpdateMetadataRequest ::= { tagsForDeletion 'BF20'H }
REM_SP_NAME	metadataReq UpdateMetadataRequest ::= { tagsForDeletion '91'H }

D.6 VOID

D.7 VOID

D.8 VOID

D.9 VOID

Annex E Profiles

Profile	GENERIC_PROFILE_STRUCTURE
Description	Generic Operational Profile ASN.1 structure to be used as a basis for all Profiles used in this specification.

Details	headerValue ProfileElement ::= header : {
	major-version 2,
	minor-version 3,
	profileType "GSMA Profile Package",
	iccid '89019990001234567893'H,
	eUICC-Mandatory-services {
	usim NULL,
	milenage NULL
	},
	eUICC-Mandatory-GFSTEList {
	-- see Note 1
	id-MF,
	id-USIM
	}
	}
	
	mfValue ProfileElement ::= mf : {
	mf-header {
	mandated NULL,
	identification 1
	},
	templateID id-MF,
	mf {
	fileDescriptor : {
	pinStatusTemplateDO '01020A'H
	}
	},
	ef-pl {
	fileDescriptor : {
	-- EF PL modified to use Access Rule 15 within EF ARR
	securityAttributesReferenced '0F'H
	}
	},
	ef-iccid {
	-- swapped ICCID: 98109909002143658739
	fillFileContent : '98109909002143658739'H
	},
	ef-dir {
	fileDescriptor {
	-- Shareable Linear Fixed File

```
    -- 4 records, record length: 38 bytes
    fileDescriptor '42210026'H,
    efFileSize '98'H
   },
   -- USIM AID: A0000000871002FF33FF018900000100
   fillFileContent
    '61184F10A0000000871002FF33FF0189000001005004555349 4D'H
  },
  ef-arr {
   fileDescriptor : {
    fileDescriptor '42210025'H,
    lcsi '05'H,
    efFileSize '022B'H
   },
   fillFileContent : '8001019000800102A40683010195010880015 8A40683010A950108'H,
   fillFileOffset : 10,
   fillFileContent : '800101A40683010195010880015AA40683010A950108'H,
   fillFileOffset : 15,
   fillFileContent : '80015BA40683010A950108'H,
   fillFileOffset : 26,
   fillFileContent : '800101900080015A9700'H,
   fillFileOffset : 27,
   fillFileContent : '800103A40683010195010880015 8A40683010A950108'H,
   fillFileOffset : 15,
   fillFileContent : '800111A40683010195010880014AA40683010A950108'H,
   fillFileOffset : 15,
   fillFileContent :
'800103A40683010195010880015 8A40683010A950108840132A406830101950108'H,
   fillFileOffset : 4,
   fillFileContent :
'800101A40683010195010880010 2A40683018195010880015 8A40683010A950108'H,
   fillFileOffset : 4,
   fillFileContent : '800101900080011AA40683010195010880014 0A40683010A950108'H,
   fillFileOffset : 10,
   fillFileContent : '800101900080015AA40683010A950108'H,
   fillFileOffset : 21,
   fillFileContent : '8001019000800118A40683010A9501088001429700'H,
   fillFileOffset : 16,
   fillFileContent : '800101A40683010195010880015A9700'H,
   fillFileOffset : 21,
   fillFileContent : '800113A40683010195010880014 8A40683010A950108'H,
   fillFileOffset : 15,
   fillFileContent : '80015EA40683010A950108'H,
   fillFileOffset : 26,
   fillFileContent  '8001019000800102A010A40683010195
                    0108A40683010295010880015 8A40683
```

```
                    010A950108'H
  }
}

pukVal ProfileElement ::= pukCodes : {
  puk-Header {
    mandated NULL,
    identification 2
  },
  pukCodes {
    {
      keyReference pukAppl1,
      pukValue '30303030303030'H,
      -- maxNumOfAttemps:9, retryNumLeft:9
      maxNumOfAttemps-retryNumLeft 153
    },
    {
      keyReference pukAppl2,
      pukValue '3132333435363738'H
    },
    {
      keyReference secondPUKAppl1,
      pukValue '3932393435363738'H,
      -- maxNumOfAttemps:8, retryNumLeft:8
      maxNumOfAttemps-retryNumLeft 136
    }
  }
}
pinVal ProfileElement ::= pinCodes : {
  pin-Header {
    mandated NULL,
    identification 3
  },
  pinCodes pinconfig : {
    {
      keyReference pinAppl1,
      pinValue '31323334FFFFFFFF'H,
      unblockingPINReference pukAppl1
    },
    {
      keyReference pinAppl2,
      pinValue '30303030FFFFFFFF'H,
      unblockingPINReference pukAppl2
    },
    {
```

```
      keyReference adm1,
      pinValue '35363738FFFFFFFF'H,
      pinAttributes 1
    }
  }
}
usimValue ProfileElement ::= usim : {
  usim-header {
    mandated NULL,
    identification 4
  },
  templateID id-USIM,
  adf-usim {
    fileDescriptor : {
      fileID '7FF1'H,
      dfName 'A0000000871002FF33FF018900000100'H,
      pinStatusTemplateDO '01810A'H
    }
  },
  ef-imsi {
    -- numerical format: 234101943787656
    fillFileContent '08294301913487 6765'H
  },
  ef-arr {
    fileDescriptor {
      linkPath '2F06'H
    }
  },
  ef-ust {
    -- Service Dialling Numbers, Short Message Storage…
    fillFileContent '0A2E178CE73204000000000000'H
  },
  ef-spn {
    -- ASCII format: "GSMA eUICC"
    fillFileContent '0247534D41206555494343FFFFFFFFFFFF'H
  },
  ef-est {
    -- Services deactivated
    fillFileContent '00'H
  },
  ef-acc {
    -- Access class 4
    fillFileContent '0040'H
  },
  ef-ecc {
```

```
    -- Emergency Call Code 911
    fillFileContent '19F1FF01'H
  }
}

usimPin ProfileElement ::= pinCodes : {
  pin-Header {
    mandated NULL,
    identification 5
  },
  pinCodes pinconfig : {
    {
      keyReference secondPINAppl1,
      pinValue '39323338FFFFFFFF'H
      unblockingPINReference secondPUKAppl1,
      -- PIN is Enabled
      pinAttributes 1,
      -- maxNumOfAttemps:2, retryNumLeft:2
      maxNumOfAttemps-retryNumLeft 34
    }
  }
}

akaParamValue ProfileElement ::= akaParameter : {
  aka-header {
    mandated NULL,
    identification 6
  },
  algoConfiguration algoParameter : {
    algorithmID milenage,
    -- RES and MAC 64 bits, CK and IK 128 bits
    algorithmOptions '01'H,
    key '000102030405060708090A0B0C0D0E0F'H,
    opc '0102030405060708090A0B0C0D0E0F00'H,
    -- rotationConstants uses default: '4000204060'H
    -- xoringConstants uses default value
    authCounterMax '010203'H
  }
  -- sqnOptions uses default: '02'H
  -- sqnDelta uses default: '000010000000'H
  -- sqnAgeLimit uses default: '000010000000'H
  -- sqnInit uses default: all bytes zero
}

mnoSdValue ProfileElement ::= securityDomain : {
```

```
sd-Header {
  mandated NULL,
  identification 7
},
instance {
  applicationLoadPackageAID 'A0000001515350'H,
  classAID 'A000000151535041'H,
  instanceAID 'A000000151000000'H,
  applicationPrivileges '82FC80'H,
  -- Secured
  lifeCycleState '0F'H,
  -- SCP80 supported
  applicationSpecificParametersC9 '810280008201F08701F0'H,
  -- other parameters MAY be necessary
  applicationParameters {
    -- TAR: B20100, MSL: 12
    uiccToolkitApplicationSpecificParametersField
      '0100000100000002011203B2010000'H
  }
},
keyList {
  {
    -- C-ENC + R-ENC
    keyUsageQualifier '38'H,
    -- ENC key
    keyIdentifier '01'H,
    keyVersionNumber '01'H,
    keyCompontents {
      {
        -- DES mode implicitly known (as an example)
        keyType '80'H,
        -- This value MAY be freely changed
        keyData '112233445566778899AABBCCDDEEFF10'H
      }
    }
  },
  {
    -- C-MAC + R-MAC
    keyUsageQualifier '34'H,
    -- MAC key
    keyIdentifier '02'H,
    keyVersionNumber '01'H,
    keyCompontents {
      {
        -- DES mode implicitly known (as an example)
```

```
        keyType '80'H,
        -- This value MAY be freely changed
        keyData '112233445566778899AABBCCDDEEFF10'H
      }
    }
  },
  {
    -- C-DEK + R-DEK
    keyUsageQualifier 'C8'H,
    -- data ENC key
    keyIdentifier '03'H,
    keyVersionNumber '01'H,
    keyCompontents {
      {
        -- DES mode implicitly known (as an example)
        keyType '80'H,
        -- This value MAY be freely changed
        keyData '112233445566778899AABBCCDDEEFF10'H
      }
    }
  },
    -- AES Token Key (as an example)
    -- This value MAY be freely changed
          keyUsageQualifier  '81'H,
          -- MAY be used by SD
          keyAccess  '01'H,
          -- Key Id 01
          keyIdentifier  '01'H,
          keyVersionNumber '70'H,
          keyCompontents  {
           {
      -- AES (16 bytes key length)
      -- This value MAY be freely changed
            keyType  '88'H,
            -- This value MAY be freely changed
            keyData  'CDFE56B7B72FAE6A047341F003D7A48D'H
           }
    }
  },
  {
    -- Receipt (the AES scheme SHALL be supported)
          keyUsageQualifier '44'H,
          -- MAY be used by SD
          keyAccess '01'H,
          -- Key Id 01
```

```
            keyIdentifier '01'H,
            keyVersionNumber '71'H,
            keyCompontents  {
             {
              -- AES (16 bytes key length)
              keyType  '88'H,
              -- This value MAY be freely changed
              keyData  '11121314212223243132333441424344'H
             }
            }
   }
  }
}

ssdValue ProfileElement ::= securityDomain : {
  sd-Header {
    mandated NULL,
    identification 8
  },
  instance {
    applicationLoadPackageAID 'A0000001515350'H,
    classAID 'A000000151535041'H,
    instanceAID 'A00000055910100102736456616C7565'H,
    -- by default extradited under MNO-SD
    -- Privileges: Security Domain + Trusted Path
    applicationPrivileges '808000'H,
    -- Personalized
    lifeCycleState '0F'H,
    -- SCP80 supported, extradiction supported
    applicationSpecificParametersC9 '810280008201F0'H,
    applicationParameters {
      -- TAR: 6C7565, MSL: 12
      uiccToolkitApplicationSpecificParametersField
        '0100000100000020112036C756500'H
    }
  },
  keyList {
    {
      -- C-ENC + R-ENC
      keyUsageQualifier '38'H,
      keyIdentifier '01'H,
      keyVersionNumber '01'H,
      keyCompontents {
        {
          -- DES mode implicitly known (as an example)
```

```
        keyType '80'H,
        -- This value MAY be freely changed
        keyData '11223344556677881122334455667788'H
      }
    }
  },
  {
    -- C-MAC + R-MAC
    keyUsageQualifier '34'H,
    -- MAC key
    keyIdentifier '02'H,
    keyVersionNumber '01'H,
    keyCompontents {
      {
        -- DES mode implicitly known (as an example)
        keyType '80'H,
        -- This value MAY be freely changed
        keyData '11223344556677881122334455667788'H
      }
    }
  },
  {
    -- C-DEK + R-DEK
    keyUsageQualifier 'C8'H,
    -- data ENC key
    keyIdentifier '03'H,
    keyVersionNumber '01'H,
    keyCompontents {
      {
        -- DES mode implicitly known (as an example)
        keyType '80'H,
        -- This value MAY be freely changed
        keyData '11223344556677881122334455667788'H
      }
    }
  }
 }
}

rfmUicc ProfileElement ::= rfm : {
 rfm-header {
  identification 11
 },
 -- Instance AID
 instanceAID ' A00000055910100001'H,
```

```
  tarList {
    'B00000'H
  },
  -- cryptographic checksum + counter higher
  minimumSecurityLevel '12'H,
  -- full access
  uiccAccessDomain '00'H,
  -- full access
  uiccAdminAccessDomain '00'H
}

rfmUsim ProfileElement ::= rfm : {
  rfm-header {
    identification 12
  },
  -- Instance AID
  instanceAID 'A00000055910100002'H,
  tarList {
    'B00020'H
  },
  -- cryptographic checksum + counter higher
  minimumSecurityLevel '12'H,
  -- full access
  uiccAccessDomain '00'H,
  -- full access
  uiccAdminAccessDomain '00'H,
  adfRFMAccess {
    adfAID 'A0000000871002FF33FF018900000100'H,
    -- UICC access condition: ADM1
    adfAccessDomain '02000100'H,
    -- UICC access condition: ADM1
    adfAdminAccessDomain '02000100'H
  }
}

endValue ProfileElement ::= end : {
  end-header {
    mandated NULL,
    identification 99
  }
}
```

Note 1: The following OIDs are used:

```
   id-MF OBJECT IDENTIFIER ::=
   {joint-iso-itu-t(2) international-organizations(23) tca(143) euicc-profile(1)
template(2) mf(1)}
   id-USIM OBJECT IDENTIFIER ::=
```

```
   {joint-iso-itu-t(2) international-organizations(23) tca(143) euicc-profile(1)
template(2) usim(4)}
```

Profile	PROFILE_OPERATIONAL1
Description	Operational Profile This Profile acts as an Operational Profile in the scope of this specification. NOTE: Milenage algorithm is used in this Profile.
Details	The Profile Metadata SHALL be set to #METADATA_OP_PROF1, except if defined differently in the test sequence. The Unprotected Profile Package content SHALL follow the ASN.1 structures specified above for GENERIC_PROFILE_STRUCTURE except that: • the *iccid* field SHALL be set to #ICCID_OP_PROF1 in the *ProfileHeader* element, in non-swapped format • the ef-iccid present in the PE-MF SHALL be set to #ICCID_OP_PROF1 • the ef-imsi present in the PE-USIM SHALL be set to #IMSI_OP_PROF1 • the pinAttributes of pinAppl1 present in the PE_PIN SHALL be set to 6 • the SCP80 encryption key configured in the PE-SecurityDomain that corresponds to the MNO-SD SHALL be set to #MNO_SCP80_ENC_KEY • the SCP80 message authentication key configured in the PE-SecurityDomain that corresponds to the MNO-SD SHALL be set to #MNO_SCP80_AUTH_KEY • the SCP80 data encryption key configured in the PE-SecurityDomain that corresponds to the MNO-SD SHALL be set to #MNO_SCP80_DATA_ENC_KEY • the instance AID configured in the PE-SecurityDomain that corresponds to the Supplementary Security Domain PE_SSD SHALL be set to #SSD_AID • the ef-dir present in the PE-MF SHALL be configured with the AID #USIM_AID • the ef-ust SHALL be set in accordance to #EF_UST1 (service 17 and 18 are not available) • the applicationPrivileges in PE-MNO-SD SHALL be set to '82DC00'H • the Token Verification and the Receipt Generation keys SHALL not be set in the PE-MNO-SD • the applicationSpecificParametersC9 in PE-MNO-SD SHALL be set to '810280008201F08701F0'H The PROFILE_OPERATIONAL1 UPP is named #UPP_OP_PROF1 in the scope of this document.

Profile	PROFILE_OPERATIONAL2
Description	Operational Profile This Profile acts as an Operational Profile in the scope of this specification. NOTE: Milenage algorithm is used in this Profile.
Details	The Profile Metadata SHALL be set to #METADATA_OP_PROF2, except if defined differently in the test sequence. The Unprotected Profile Package content SHALL follow the ASN.1 structures specified above for GENERIC_PROFILE_STRUCTURE except that: • the *iccid* field SHALL be set to #ICCID_OP_PROF2 in the *ProfileHeader* element, in non-swapped format • the ef-iccid present in the PE-MF SHALL be set to #ICCID_OP_PROF2 • the ef-imsi present in the PE-USIM SHALL be set to #IMSI_OP_PROF2 • The pinAttributes of pinAppl1 present in the PE_PIN SHALL be set to 6

	• the ef-ust SHALL be set in accordance to #EF_UST1 (service 17 and 18 are not available) • the applicationPrivileges in PE-MNO-SD SHALL be set to '82DC00'H • the Token Verification and the Receipt Generation keys SHALL not be set in the PE-MNO-SD • the applicationSpecificParametersC9 in PE-MNO-SD SHALL be set to '810280008201F08701F0'H The PROFILE_OPERATIONAL2 UPP is named #UPP_OP_PROF2 in the scope of this document.

Profile	PROFILE_OPERATIONAL3
Description	Operational Profile with PPR2 but without notification This Profile acts as an Operational Profile in the scope of this specification. NOTE: Milenage algorithm is used in this Profile.
Details	The Profile Metadata SHALL be set to #METADATA_OP_PROF3, except if defined differently in the test sequence. The Unprotected Profile Package content SHALL follow the ASN.1 structures specified above for GENERIC_PROFILE_STRUCTURE except that: • the *iccid* field SHALL be set to #ICCID_OP_PROF3 in the *ProfileHeader* element, in non-swapped format • the ef-iccid present in the PE-MF SHALL be set to #ICCID_OP_PROF3 • the ef-imsi present in the PE-USIM SHALL be set to #IMSI_OP_PROF3 • the pinAttributes of pinAppl1 present in the PE_PIN SHALL be set to 6 • the ef-ust SHALL be set in accordance to #EF_UST1 (service 17 and 18 are not available) • the applicationPrivileges in PE-MNO-SD SHALL be set to '82DC00'H • the Token Verification and the Receipt Generation keys SHALL not be set in the PE-MNO-SD • the applicationSpecificParametersC9 in PE-MNO-SD SHALL be set to '810280008201F08701F0'H The PROFILE_OPERATIONAL3 UPP is named #UPP_OP_PROF3 in the scope of this document.

Profile	PROFILE_OPERATIONAL4
Description	Operational Profile with PPR1 and notification This Profile acts as an Operational Profile in the scope of this specification. NOTE: Milenage algorithm is used in this Profile.
Details	The Profile Metadata SHALL be set to #METADATA_OP_PROF4, except if defined differently in the test sequence. The Profile Package content SHALL follow the ASN.1 structures specified above for GENERIC_PROFILE_STRUCTURE except that: • the *iccid* field SHALL be set to #ICCID_OP_PROF4 in the *ProfileHeader* element, in non-swapped format • the ef-iccid present in the PE-MF SHALL be set to #ICCID_OP_PROF4 • the ef-imsi present in the PE-USIM SHALL be set to #IMSI_OP_PROF4 • the pinAttributes of pinAppl1 present in the PE_PIN SHALL be set to 6 • the ef-ust SHALL be set in accordance to #EF_UST1 (service 17 and 18 are not available)

	• the applicationPrivileges in PE-MNO-SD SHALL be set to '82DC00'H • the Token Verification and the Receipt Generation keys SHALL not be set in the PE-MNO-SD • the applicationSpecificParametersC9 in PE-MNO-SD SHALL be set to '810280008201F08701F0'H The PROFILE_OPERATIONAL4 UPP is named #UPP_OP_PROF4 in the scope of this document.

Profile	PROFILE_OPERATIONAL5
VOID (not used in this version of the specification)	

Profile	PROFILE_OPERATIONAL6
VOID (not used in this version of the specification)	

Profile	PROFILE_OPERATIONAL7
VOID (not used in this version of the specification)	

Profile	PROFILE_OPERATIONAL8
VOID (not used in this version of the specification)	

Profile	PROFILE_OPERATIONAL9
Description	Operational Profile with GID1 and GID2 set This Profile acts as an Operational Profile in the scope of this specification. NOTE: Milenage algorithm is used in this Profile.
Details	The Profile Metadata SHALL be set to #METADATA_OP_PROF9, except if defined differently in the test sequence. The Unprotected Profile Package content SHALL follow the ASN.1 structures specified above for GENERIC_PROFILE_STRUCTURE except that: • the *iccid* field SHALL be set to #ICCID_OP_PROF9 in the *ProfileHeader* element, in non-swapped format • the ef-iccid present in the PE-MF SHALL be set to #ICCID_OP_PROF9 • the ef-imsi present in the PE-USIM SHALL be set to #IMSI_OP_PROF9 • the pinAppl1 present in the PE_PIN SHALL be enabled and has the value #PO1_PIN1 • the ef-ust SHALL be set to #EF_UST2 (service 17 and 18 are available) • the applicationPrivileges in PE-MNO-SD SHALL be set to '82DC00'H • the Token Verification and the Receipt Generation keys SHALL not be set in the PE-MNO-SD • the applicationSpecificParametersC9 in PE-MNO-SD SHALL be set to '810280008201F08701F0'H • the following new Profile Element PE_OPT_USIM SHALL be inserted right after PE_USIM: **PE_OPT_USIM**

```
optusimValue ProfileElement ::= opt-usim : {
  optusim-header {
    mandated NULL,
    identification 15
  },
  templateID id-OPT-USIM,
  ef-gid1 {
    fileDescriptor {
      efFileSize '04'H
    },
    fillFileContent #GID1
  },
  ef-gid2 {
    fileDescriptor {
      efFileSize '04'H
    },
    fillFileContent #GID2
  }
}
```

NOTE: The following OIDs are used:

```
id-OPT-USIM OBJECT IDENTIFIER ::=
{joint-iso-itu-t(2) international-organizations(23) tca(143)
euicc-profile(1) template(2) opt-usim(5)}
```

The PROFILE_OPERATIONAL9 UPP is named #UPP_OP_PROF9 in the scope of this document.

Profile	5G_PROFILE_STRUCTURE
Description	5G Operational Profile ASN.1 structure to be used as a basis for specific test scenario where explicitly stated.

Details	headerValue ProfileElement ::= header : {
	major-version 3,
	minor-version 1,
	profileType "GSMA Profile Package",
	iccid '89019990001234567893'H,
	eUICC-Mandatory-services {
	usim NULL,
	milenage NULL,
	get-identity NULL,
	profile-a-x25519 NULL
	},
	eUICC-Mandatory-GFSTEList {
	-- see Note 1
	id-MF,
	id-USIM
	}
	}

```
mfValue ProfileElement ::= mf : {
 mf-header {
  mandated NULL,
  identification 1
 },
 templateID id-MF,
 mf {
  fileDescriptor : {
   pinStatusTemplateDO '01020A'H
  }
 },
 ef-pl {
  fileDescriptor : {
  -- EF PL modified to use Access Rule 15 within EF ARR
   securityAttributesReferenced '0F'H
  }
 },
 ef-iccid {
  -- swapped ICCID: 98109909002143658739
  fillFileContent : '98109909002143658739'H
 },
 ef-dir {
  fileDescriptor : {
   -- Shareable Linear Fixed File
   -- 4 records, record length: 38 bytes
   fileDescriptor '42210026'H,
   efFileSize '98'H
  },
  -- USIM AID: A000000087100BFF33FF018900000100
  fillFileContent :
   '61184F10A000000087100BFF33FF01890000010050045553494D'H
 },
 ef-arr {
  fileDescriptor : {
   fileDescriptor '42210025'H,
   lcsi '05'H,
   efFileSize '022B'H
  },
  fillFileContent : '8001019000800102A40683010195010880015 8A40683010A950108'H,
  fillFileOffset : 10,
  fillFileContent : '800101A40683010195010880015AA40683010A950108'H,
  fillFileOffset : 15,
  fillFileContent : '80015BA40683010A950108'H,
  fillFileOffset : 26,
  fillFileContent : '800101900080015A9700'H,
```

```
    fillFileOffset : 27,
    fillFileContent : '800103A406830101950108800158A40683010A950108'H,
    fillFileOffset : 15,
    fillFileContent : '800111A406830101950108800014AA40683010A950108'H,
    fillFileOffset : 15,
    fillFileContent :
'800103A406830101950108800158A40683010A950108840132A406830101950108'H,
    fillFileOffset : 4,
    fillFileContent :
'800101A406830101950108800102A406830181950108800158A40683010A950108'H,
    fillFileOffset : 4,
    fillFileContent : '800101900080011AA406830101950108800140A40683010A950108'H,
    fillFileOffset : 10,
    fillFileContent : '800101900080015AA40683010A950108'H,
    fillFileOffset : 21,
    fillFileContent : '8001019000800118A40683010A9501088001429700'H,
    fillFileOffset : 16,
    fillFileContent : '800101A406830101950108800015A9700'H,
    fillFileOffset : 21,
    fillFileContent : '800113A406830101950108800148A40683010A950108'H,
    fillFileOffset : 15,
    fillFileContent : '80015EA40683010A950108'H,
    fillFileOffset : 26,
    fillFileContent : '8001019000800102A010A40683010195
                       0108A406830102950108800158A40683
                       010A950108'H
  }
}

pukVal ProfileElement ::= pukCodes : {
  puk-Header {
    mandated NULL,
    identification 2
  },
  pukCodes {
    {
      keyReference pukAppl1,
      pukValue '3030303030303030'H,
      -- maxNumOfAttemps:9, retryNumLeft:9
      maxNumOfAttemps-retryNumLeft 153
    },
    {
      keyReference pukAppl2,
      pukValue '3132333435363738'H
    },
    {
```

```
    keyReference secondPUKAppl1,
    pukValue '3932393435363738'H,
    -- maxNumOfAttemps:8, retryNumLeft:8
    maxNumOfAttemps-retryNumLeft 136
   }
  }
}
pinVal ProfileElement ::= pinCodes : {
  pin-Header {
   mandated NULL,
   identification 3
  },
  pinCodes pinconfig : {
   {
    keyReference pinAppl1,
    pinValue '31323334FFFFFFFF'H,
    unblockingPINReference pukAppl1
   },
   {
    keyReference pinAppl2,
    pinValue '30303030FFFFFFFF'H,
    unblockingPINReference pukAppl2
   },
   {
    keyReference adm1,
    pinValue '35363738FFFFFFFF'H,
    pinAttributes 1
   }
  }
}
usimValue ProfileElement ::= usim : {
  usim-header {
   mandated NULL,
   identification 4
  },
  templateID id-USIM,
  adf-usim {
   fileDescriptor : {
    fileID '7FF1'H,
    dfName 'A0000000087100BFF33FF018900000100'H,
    pinStatusTemplateDO '01810A'H
   }
  },
  ef-imsi {
   doNotCreate : NULL
```

```
  },
  ef-arr {
    fileDescriptor : {
      linkPath '2F06'H
    }
  },
  ef-ust {
    -- Service Dialling Numbers, Short Message Storage…
    fillFileContent : '0A2E178CE732040000000000000001882'H
  },
  ef-spn {
    -- ASCII format: "GSMA eUICC"
    fillFileContent : '0247534D41206555494343FFFFFFFFFFF'H
  },
  ef-est {
    -- Services deactivated
    fillFileContent : '00'H
  },
  ef-acc {
    -- Access class 4
    fillFileContent : '0040'H
  },
  ef-ecc {
    -- Emergency Call Code 911
    fillFileContent : '19F1FF01'H
  }
}

usimPin ProfileElement ::= pinCodes : {
  pin-Header {
    mandated NULL,
    identification 5
  },
  pinCodes pinconfig : {
    {
      keyReference secondPINAppl1,
      pinValue '39323338FFFFFFFF'H,
      unblockingPINReference secondPUKAppl1,
      -- PIN is Enabled
      pinAttributes 1,
      -- maxNumOfAttemps:2, retryNumLeft:2
      maxNumOfAttemps-retryNumLeft 34
    }
  }
}
```

```
df5gsValue ProfileElement ::= df-5gs : {
  df-5gs-header {
    mandated NULL,
    identification 556
  },
templateID { 2 23 143 1 2 13 3 },
  df-df-5gs {
    fileDescriptor : {
      pinStatusTemplateDO '01810A'H
    }
  },
  ef-supinai {
    fileDescriptor : {
      lcsi '05'H,
      efFileSize '18'H
    },
    fillFileContent : '80117465737475736572406773 6D612E6F7267'H
  },
  ef-routing-indicator {
  }
}
dfsaipValue ProfileElement ::= df-saip : {
  df-saip-header {
    mandated NULL,
    identification 261
  },
templateID { 2 23 143 1 2 14 },
  df-df-saip {
    fileDescriptor : {
      lcsi '05'H,
      pinStatusTemplateDO '01810A'H
    }
  },
  ef-suci-calc-info-usim {
    fileDescriptor : {
      lcsi '05'H,
      efFileSize '04'H,
      proprietaryEFInfo {
        specialFileInformation '40'H,
        fileDetails '01'H
      }
    },
    fillFileContent : 'A0020100'H
  }
```

```
}

akaParamValue ProfileElement ::= akaParameter : {
  aka-header {
    mandated NULL,
    identification 6
  },
  algoConfiguration algoParameter : {
    algorithmID milenage,
    -- RES and MAC 64 bits, CK and IK 128 bits
    algorithmOptions '01'H,
    key '000102030405060708090A0B0C0D0E0F'H,
    opc '0102030405060708090A0B0C0D0E0F00'H,
    -- rotationConstants uses default: '4000204060'H
    -- xoringConstants uses default value
    authCounterMax '010203'H
  }
  -- sqnOptions uses default: '02'H
  -- sqnDelta uses default: '000010000000'H
  -- sqnAgeLimit uses default: '000010000000'H
  -- sqnInit uses default: all bytes zero
}

mnoSdValue ProfileElement ::= securityDomain : {
  sd-Header {
    mandated NULL,
    identification 7
  },
  instance {
    applicationLoadPackageAID 'A0000001515350'H,
    classAID 'A000000151535041'H,
    instanceAID 'A000000151000000'H,
    applicationPrivileges '82FC80'H,
    -- Secured
    lifeCycleState '0F'H,
    -- SCP80 supported
    applicationSpecificParametersC9 '810280008201F08701F0'H,
    -- other parameters MAY be necessary
    applicationParameters {
      -- TAR: B20100, MSL: 12
      uiccToolkitApplicationSpecificParametersField
        '0100000100000002011203B2010000'H
    }
  },
  keyList {
```

```
{
 -- C-ENC + R-ENC
 keyUsageQualifier '38'H,
 -- ENC key
 keyIdentifier '01'H,
 keyVersionNumber '01'H,
 keyCompontents {
  {
   -- DES mode implicitly known (as an example)
   keyType '80'H,
   -- This value MAY be freely changed
   keyData '112233445566778899AABBCCDDEEFF10'H
  }
 }
},
{
 -- C-MAC + R-MAC
 keyUsageQualifier '34'H,
 -- MAC key
 keyIdentifier '02'H,
 keyVersionNumber '01'H,
 keyCompontents {
  {
   -- DES mode implicitly known (as an example)
   keyType '80'H,
   -- This value MAY be freely changed
   keyData '112233445566778899AABBCCDDEEFF10'H
  }
 }
},
{
 -- C-DEK + R-DEK
 keyUsageQualifier 'C8'H,
 -- data ENC key
 keyIdentifier '03'H,
 keyVersionNumber '01'H,
 keyCompontents {
  {
   -- DES mode implicitly known (as an example)
   keyType '80'H,
   -- This value MAY be freely changed
   keyData '112233445566778899AABBCCDDEEFF10'H
  }
 }
},
```

```
    {
      -- AES Token Key (as an example)
      -- This value MAY be freely changed
            keyUsageQualifier  '81'H,
            -- MAY be used by SD
            keyAccess  '01'H,
            -- Key Id 01
            keyIdentifier  '01'H,
            keyVersionNumber '70'H,
            keyCompontents  {
             {
        -- AES (16 bytes key length)
        -- This value MAY be freely changed
              keyType  '88'H,
              -- This value MAY be freely changed
              keyData  'CDFE56B7B72FAE6A047341F003D7A48D'H
             }
     }
    },
    {
      -- Receipt (the AES scheme SHALL be supported)
            keyUsageQualifier '44'H,
            -- MAY be used by SD
            keyAccess '01'H,
            -- Key Id 01
            keyIdentifier '01'H,
            keyVersionNumber '71'H,
            keyCompontents  {
             {
              -- AES (16 bytes key length)
              keyType  '88'H,
              -- This value MAY be freely changed
              keyData  '11121314212223243132333441424344'H
             }
            }
    }
  }
}

ssdValue ProfileElement ::= securityDomain : {
  sd-Header {
    mandated NULL,
    identification 8
  },
  instance {
```

```
  applicationLoadPackageAID 'A0000001515350'H,
  classAID 'A000000151535041'H,
  instanceAID 'A00000055910100102736456616C7565'H,
  -- by default extradited under MNO-SD
  -- Privileges: Security Domain + Trusted Path
  applicationPrivileges '808000'H,
  -- Personalized
  lifeCycleState '0F'H,
  -- SCP80 supported, extradiction supported
  applicationSpecificParametersC9 '810280008201F0'H,
  applicationParameters {
    -- TAR: 6C7565, MSL: 12
    uiccToolkitApplicationSpecificParametersField
      '0100000100000020112036C756500'H
  }
},
keyList {
  {
    -- C-ENC + R-ENC
    keyUsageQualifier '38'H,
    keyIdentifier '01'H,
    keyVersionNumber '01'H,
    keyCompontents {
      {
        -- DES mode implicitly known (as an example)
        keyType '80'H,
        -- This value MAY be freely changed
        keyData '11223344556677881122334455667788'H
      }
    }
  },
  {
    -- C-MAC + R-MAC
    keyUsageQualifier '34'H,
    -- MAC key
    keyIdentifier '02'H,
    keyVersionNumber '01'H,
    keyCompontents {
      {
        -- DES mode implicitly known (as an example)
        keyType '80'H,
        -- This value MAY be freely changed
        keyData '11223344556677881122334455667788'H
      }
    }
```

```
    },
    {
      -- C-DEK + R-DEK
      keyUsageQualifier 'C8'H,
      -- data ENC key
      keyIdentifier '03'H,
      keyVersionNumber '01'H,
      keyCompontents {
       {
         -- DES mode implicitly known (as an example)
         keyType '80'H,
         -- This value MAY be freely changed
         keyData '11223344556677881122334455667788'H
       }
      }
    }
  }
}

rfmUicc ProfileElement ::= rfm : {
  rfm-header {
    identification 11
  },
  -- Instance AID
  instanceAID ' A00000055910100001'H,
  tarList {
    'B00000'H
  },
  -- cryptographic checksum + counter higher
  minimumSecurityLevel '12'H,
  -- full access
  uiccAccessDomain '00'H,
  -- full access
  uiccAdminAccessDomain '00'H
}

rfmUsim ProfileElement ::= rfm : {
  rfm-header {
    identification 12
  },
  -- Instance AID
  instanceAID 'A00000055910100002'H,
  tarList {
    'B00020'H
  },
```

	-- cryptographic checksum + counter higher
	minimumSecurityLevel '12'H,
	-- full access
	uiccAccessDomain '00'H,
	-- full access
	uiccAdminAccessDomain '00'H,
	adfRFMAccess {
	adfAID ' A000000087100BFF33FF018900000100'H,
	-- UICC access condition: ADM1
	adfAccessDomain '02000100'H,
	-- UICC access condition: ADM1
	adfAdminAccessDomain '02000100'H
	}
	}
	
	endValue ProfileElement ::= end : {
	end-header {
	mandated NULL,
	identification 99
	}
	}

Note 1: The following OIDs are used: *id-MF OBJECT IDENTIFIER ::=* *{joint-iso-itu-t(2) international-organizations(23) simalliance(143) euicc-profile(1) template(2) mf(1)}* *id-USIM OBJECT IDENTIFIER ::=* *{joint-iso-itu-t(2) international-organizations(23) simalliance(143) euicc-profile(1) template(2) usim(4) version2(2)}*	

Profile	PROFILE_OPERATIONAL10
Description	Operational Profile This Profile acts as an Operational Profile in the scope of this specification. NOTE: Milenage algorithm is used in this Profile
Details	The Profile Metadata SHALL be set to #METADATA_OP_PROF10, except if defined differently in the test sequence. The Unprotected Profile Package content SHALL follow the ASN.1 structure specified above for 5G_PROFILE_STRUCTURE except that: • the *iccid* field SHALL be set to #ICCID_OP_PROF10 in the *ProfileHeader* element, in non-swapped format • the ef-iccid present in the PE-MF SHALL be set to #ICCID_OP_PROF10 • the pinAttributes of pinAppl1 present in the PE_PIN SHALL be set to 6 • the SCP80 encryption key configured in the PE-SecurityDomain that corresponds to the MNO-SD SHALL be set to #MNO_SCP80_ENC_KEY • the SCP80 message authentication key configured in the PE-SecurityDomain that corresponds to the MNO-SD SHALL be set to #MNO_SCP80_AUTH_KEY • the SCP80 data encryption key configured in the PE-SecurityDomain that corresponds to the MNO-SD SHALL be set to #MNO_SCP80_DATA_ENC_KEY

	• the instance AID configured in the PE-SecurityDomain that corresponds to the Supplementary Security Domain PE_SSD SHALL be set to #SSD_AID • the ef-dir present in the PE-MF SHALL be configured with the AID #USIM_AID • the applicationPrivileges in PE-MNO-SD SHALL be set to '82DC00'H • the Token Verification and the Receipt Generation keys SHALL not be set in the PE-MNO-SD • the applicationSpecificParametersC9 in PE-MNO-SD SHALL be set to '810280008201F08701F0'H

Annex F eUICC Settings

F.1 eUICC Settings

In order to execute the test cases defined in this document, the eUICC Manufacturer SHALL deliver following settings:

eUICC Setting name	Description
IUT_ADDITIONAL_EUICC_INFO	Information about the eUICC as defined by the EUM. It MAY correlate with the additional issuer information contained in the EID
IUT_DLOA_URL	Discovery Base URL of the SE default DLOA Registrar as defined in GlobalPlatform DLOA specification [19] (optional).
IUT_EUICC_ADD_PP_VERSIONS	The expected content, if any, of the `additionalEuiccProfilePackageVersions` field in `EUICCInfo2`, coded as binary value without tag and length. The version(s) indicated in this field SHALL be version(s) listed in section 7.1 in the “Allowed values for #IUT_EUICC_ADD_PP_VERSIONS” column.
IUT_EUICC_CATEGORY	The category SHALL be present as defined below: • other(0) • or basicEuicc(1) • or mediumEuicc(2) • or contactlessEuicc(3)
IUT_EUICC_CERT_CHAIN_VARIANT	Preferred eUICC Certificate chain variant. It SHALL be either O, Ov3, A, B or C. NOTE: the eUICC MAY be configured with several Certificates using different Variants. Unless otherwise specified, the test sequences in this document let the eUICC use the variant indicated in this IUT setting.
IUT_EUICC_CERT_CHAIN_VARIANT_CI_PK_ID	The Root CI Public Key Identifier of the Root CI corresponding to the eUICC Certificate implied by #IUT_EUICC_CERT_CHAIN_VARIANT. It SHALL be a Root CI defined in SGP.26 [25].
IUT_EUICC_FF_TYPE	The form factor type of the eUICC as an ASN.1 INTEGER value, either removableEuicc(0) if eUICC can be removed, or nonRemovableEuicc(1) if eUICC cannot be removed-
IUT_EUICC_FIRMWARE_VER	eUICC Firmware version coded as binary value (3 bytes representing major/minor/revision).
IUT_EUICC_MULTIPLEXING_LSI_INDICATION	One of: • MANAGE LSI(Select LSI) • T=1 + NAD byte
IUT_GLOBALPLATFORM_VERSION	GlobalPlatform version coded as binary value (3 bytes representing major/minor/revision, 2.3.0 or higher). The support of GlobalPlatform is considered as mandatory in the scope of this specification.
IUT_MEP_LSI_OPTIONS	LSI options supported by the eUICC and Terminal
IUT_MEP_MAX_LSIS	Maximum number of LSIs supported for Enabled Profiles by the eUICC

IUT_PLATFORM_LABEL	Platform_Label as defined in GlobalPlatform DLOA specification [19] (optional).
IUT_PP_VERSION	Protection Profile version coded as binary value (3 bytes representing major/minor/revision).
IUT_RSP_VERSION_HIGHEST	Highest SGP.22 version supported, encoded as the value part of an ASN.1 VersionType (e.g. `0x03 01 00`)
IUT_SAS_ACCREDITATION_NUMBER	SAS Accreditation Number, coded as ASN.1 UTF8String.
IUT_SERVER_CI_PK_ID	The Root CI Public Key Identifier of a Root CI on the curve of which the eUICC supports server Certificates chain verification. It SHALL be a Root CI defined in SGP.26 [25].
IUT_SIMA_VERSION	Version of TCA (formerly SIMalliance) eUICC Profile Package Specification [4] supported by the eUICC (3 bytes representing major/minor/revision) e.g. 0x020100.
IUT_TS102241_VERSION	The ts102241 version field is coded as binary value (3 bytes representing major/minor/revision, 17.0.0 or higher). The ts102241 Version field indicates the latest version of ETSI TS102 241[17] supported by the eUICC.
IUT_UICC_CAPABILITY	Sequence is derived from ServicesList[] defined in eUICC Profile Package PEDefinitions, coded as ASN.1 BIT STRING (19 bits).

F.2 VOID

F.3 VOID

F.4 VOID

Annex G Initial States

Unless it is defined differently in a particular test case, the IUTs SHALL be set in the following initial state before the test case execution.

G.1 VOID

G.2 eUICC

Depending on the test cases and on the supported options, the EUM SHALL configure the eUICC according to the following Initial States. The initial conditions SHALL be restored, as described in the following subsections, after each test sequence.

G.2.1 Common Initial States

The following initial states apply for all test cases defined in this Test Plan whatever the options supported by the eUICC:

- The eUICC is configured with the ISD-R AID #ISD_R_AID and the EID #EID1.
- The eUICC does not contain any Profile.
- The eUICC's Pending Notifications List is empty.
- No RSP session is ongoing.
- The eUICC is configured with the default SM-DS address #TEST_ROOT_DS_ADDRESS.
- No additional Root DS addresses has been configured.
- The eUICC is configured without Default SM-DP+ address.
- No CRL is loaded on the eUICC.
- The ECASD is configured as defined in section G.2.2 and/or G.2.3 depending on the support of the options O_E_NIST and O_E_BRP. Other configurations are FFS.
    - If the eUICC only supports O_E_NIST, the ECASD is configured as defined in section G.2.2.
    - If the eUICC only supports O_E_BRP, the ECASD is configured as defined in section G.2.3.
    - If the eUICC supports O_E_NIST and O_E_BRP, the ECASD is configured as defined in sections G.2.2 and G.2.3 (i.e. several EUM / eUICC Certificates and Keys are configured in the eUICC).
- Independent of configurations defined in sections G.2.2 and/or G.2.3, the eUICC may be configured with certificates according to the different variants identified by the following options: O_varO, O_varOv3, O_varA, O_varB, O_varC.

The CI, identified as highest priority in euiccCiPKIdListForSigning or (if variants A/B/C are supported) in euiccCiPKIdListForSigningV3, is also selectable in the euiccCiPKIdListForVerification (i.e. all EUM and eUICC Certificates lead to a Root CI certificate linked to a #PK_CI_SIG contained in the eUICC).

This CI corresponds to the SubjectKeyIdentifier of one of the #CERT_CI_SIG defined in sections G.2.2 and G.2.3.

The RAT configuration defined in section G.2.4 applies for all test sequences if Multiple Enabled Profiles feature is not supported by the eUICC, except if the Test Case overrides it. Alternative RAT configurations for specific Test Cases or eUICC settings are defined in sections G.2.5, G.2.6 and G.2.7.

G.2.2 For eUICC supporting NIST P-256

If the eUICC supports O_E_NIST, the ECASD contains at least:

- The eUICC's Private Key #SK_EUICC_SIG (for creating ECDSA signatures) based on NIST P-256 [11]
- The eUICC's Certificate #CERT_EUICC_SIG (for eUICC authentication) containing the eUICC's Public Key #PK_EUICC_SIG based on NIST P-256 [11]
- A Root Certificate Issuer's Public Key #PK_CI_SIG (for verifying off-card entities certificates) defined in SGP.26 [25] and based on NIST P-256 [11]
- The Certificate of the EUM #CERT_EUM_SIG defined in SGP.26 [25] and based on NIST P-256 [11]

Other Certificates and Keys MAY be present.

G.2.3 For eUICC supporting BrainpoolP256r1

If the eUICC supports O_E_BRP, the ECASD contains at least:

- The eUICC's Private Key #SK_EUICC_SIG (for creating ECDSA signatures) based on brainpoolP256r1 [8]
- The eUICC's Certificate #CERT_EUICC_SIG (for eUICC authentication) containing the eUICC's Public Key #PK_EUICC_SIG based on brainpoolP256r1 [8]
- A Root Certificate Issuer's Public Key #PK_CI_SIG (for verifying off-card entities certificates) defined in SGP.26 [25] and based on brainpoolP256r1 [8]
- The Certificate of the EUM #CERT_EUM_SIG defined in SGP.26 [25] and based on brainpoolP256r1 [8]
- Other Certificates and Keys MAY be present.

G.2.4 With RAT configuration for SEP

The eUICC's RAT is configured as detailed in SGP.22 section 2.9.2:

- Only one PPAR authorizing PPR1 and PPR2 for all MNOs with End User consent required i.e. #PPRS_ALLOWED

This configuration SHALL be used as a default if the eUICC does not support MEP feature.

G.2.5 With Additional PPARs in the RAT

The eUICC's RAT is configured as below (following this order):

- Additional PPARs representing custom agreements between MNOs and OEMs:
    - #PPR1_WITH_OWNER_GID
    - #PPR1_WITHOUT_GID
    - #PPR2_WITHOUT_CONSENT

- The last PPAR authorizes PPR1 and PPR2 for all MNOs with End User consent required i.e. #PPRS_ALLOWED

This configuration is used in specific test cases if the eUICC does not support MEP feature.

G.2.6 Clean-up procedure

Unless differently specified in the test case, the following procedure SHALL be executed after each test sequence to bring the eUICC back to its Common Initial State:

- eUICC Memory Reset to delete all profiles and reset the SM-DP+ Address
- Retrieve and Remove all pending notifications

Where necessary, in addition to the above, other steps may be executed to restore the initial state specified in this Annex.

G.2.7 With RAT configuration for MEP

The eUICC's RAT is configured as below:

- Only one PPAR authorizing PPR2 for all MNOs with End User consent required i.e. #PPR2_ALLOWED

This configuration SHALL be used as a default if the eUICC supports MEP feature and allows PPR2.

G.3 VOID

Annex H Icons and QR Codes

The files for the eUICC Consumer Devices Icons are provided within SGP.23_AnnexH_Icons.zip package, which accompany the present document.

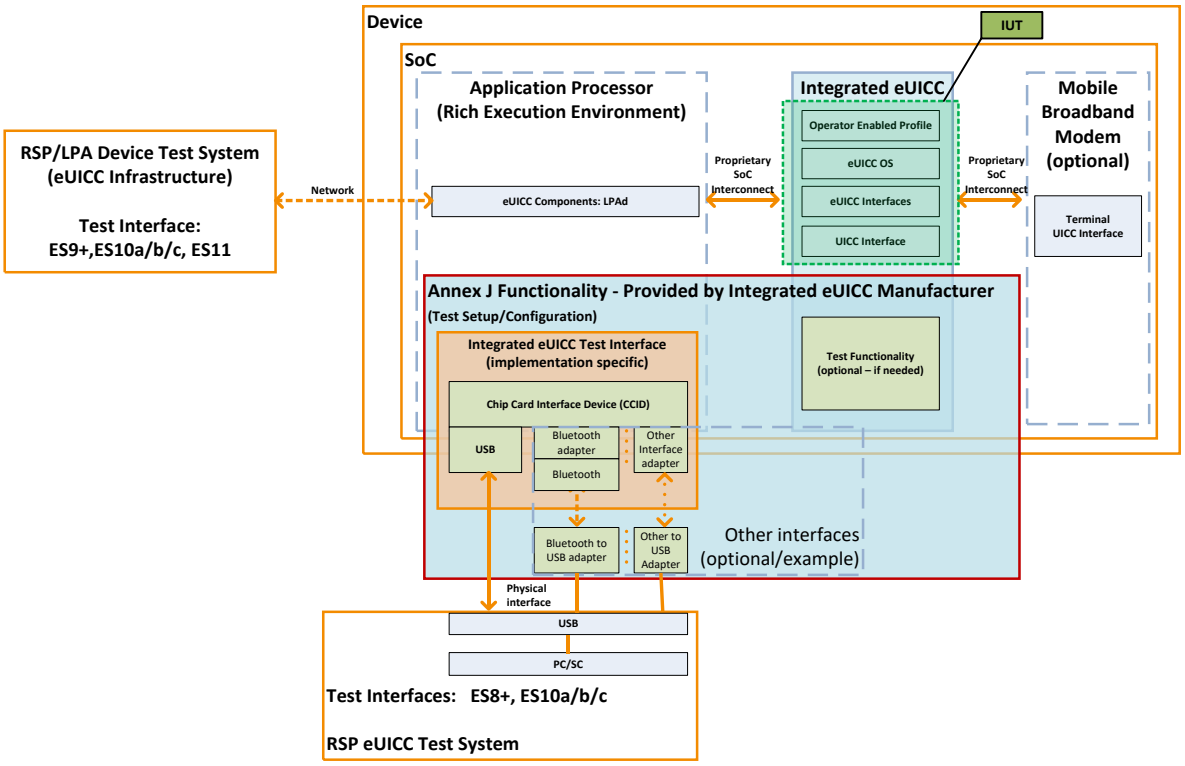
Annex J Integrated eUICC Testing (Normative)

J.1 Overview (Informative)

An Integrated eUICC hardware resides in an SoC along with other subsystems such as general processing and mobile broadband modem, all connected through a proprietary SoC interconnect channel. Alternatively, an Integrated eUICC may communicate with a mobile broadband modem external to the SoC via an external interface, which may be proprietary or based on a standard not associated with UICC. As such, Integrated eUICC may not include a physical UICC-Terminal interface [5].

In order to test the functionality and compliancy of an Integrated eUICC, hardware and OS, Integrated eUICC manufacturers need to provide and support a test interface to which testing equipment can be connected to. Having a standardized testing interface, will increase interoperable and reusability between different manufacturer of Integrated eUICC and test equipment.

For Integrated eUICC with a USB CCID [28] test interface, this annex describes its properties. In cases where a USB interface is not available in a device containing an Integrated eUICC, an adapter to USB CCID needs to be provided, e.g. Bluetooth to USB CCID. The functionality needed to provide and support the test interface, shall be considered part of the test environment and not the IUT.



Integrated eUICC with USB CCID [28] Test Interface

Note: The mechanism providing USB CCID to the RSP eUICC Test System, and described in this Annex, is implementation specific. As such, it may be implemented in the SoC, on-Device, off-Device or any combination thereof.

J.2 Integrated eUICC test requirements

An Integrated eUICC manufacturer shall provide a USB CCID test interface implementing the functionality specified in J.3.

The test interface shall maintain the integrity and order of the data between the Integrated eUICC and the test system.

The Integrated eUICC manufacturer shall ensure that during testing no other clients or SoC subsystems interfere with the testing.

The Integrated eUICC may use any physical or logical interface between the Integrated eUICC and the test system, as long as a USB CCID is provided to the test system and the channel is reliable (i.e. maintain integrity and order).

J.3 USB CCID test interface

The Integrated eUICC USB CCID test interface shall operate in a card reader mode.

The Integrated eUICC USB CCID test interface shall support the following [26] section 6 messages:

- [26] section 6.1 Messages:
  - PC_to_RDR_IccPowerOn
  - PC_to_RDR_IccPowerOff
  - PC_to_RDR_GetSlotStatus
  - PC_to_RDR_Escape
  - PC_to_RDR_XfrBlock
  - PC_to_RDR_T0APDU
  - PC_to_RDR_Secure
  - PC_to_RDR_Abort
- [26] section 6.2 Messages:
  - RDR_to_PC_SlotStatus
  - RDR_to_PC_Escape
  - RDR_to_PC_DataBlock

Note: For test systems using wincard.h/PCSC lite APIs to connect to the Integrated eUICC USB CCID test interface, the following APIs are expected to be used:

- SCardEstablishContext
- SCardListReaders[A|W]
- SCardConnect[A|W]
- SCardControl
- SCardTransmit
- SCardDisconnect
- SCardStatus[A|W]
- SCardReleaseContext
- SCardReconnect

- SCardBeginTransaction
- SCardEndTransaction
- SCardGetStatusChange
- SCardFreeMemory
- SCardGetAttrib

Annex L Document Management

L.1 Document History

Version	Date	CR No	Brief Description of Change	Entity	Approval Authority	Editor / Company
v1.0	9th June 2017		Initial version of SGP.23 v1.0 Test Specification		PSMC	Yolanda Sanz, GSMA
v1.1	28th Sept 2017		Minor version of SGP.23 Test specifications		RSPLEN	Yolanda Sanz, GSMA
v1.2	3rd Jan 2018		Minor version of SGP.23 Test specifications		RSPLEN	Yolanda Sanz, GSMA
SGP.23-1 V2.0 Draft 0	27th April 2018		The first draft 1 after Dividing SGP.23 in three different documents (SGP.23-1, 2 and 3)		RSPTEST	Sebastien Kuras, FIME
SGP.23-1 v3.1 Draft 31	5 October 2023		Draft sent to eSIM for issuing	eUICC	eSIMWG3	Guido Abate, STMicroelectronics
SGP.23-1 v3.1.1 Draft 0	17 October 2023	Editor's review	Same content as SGP.23-1 v3.1 Draft 31. All changes accepted	eUICC	ISAG	Guido Abate, STMicroelectronics
SGP.23-1 v3.1.1 Draft 1	31 October 2023	CR3110 03r01	MEP_MEP-B without REFRESH optional	eUICC	ISAG	Guido Abate, STMicroelectronics
		CR3110 04R01	ES10c_EnableProfile_MEPA1 _A2			
		CR3110 07	FixContactPcmpErrorCodes			
		CR3110 08R01	MultipleRPMCommands			
		CR3110 09	FixStoreMetadataEnterpriseProfiles			
		CR3110 11	RemovePPRs			
		Editor's review	As agreed during eSIMWG3#99: - Modified condition in applicability table for test 4.2.27.2.1, sequences 1 and 2: from 'M' to 'C319'			

			- Modified condition in applicability table for test 5.2.3.2.1: from 'M' to 'C319' - Voided section G.2.7			
SGP.23-1 v3.1.1 Draft 2	5 December 2023	Editor's review	As agreed during eSIMWG3#100: - Un-voided section G.2.7	eUICC		Guido Abate, STMicroelectronics
		CR3110 02R04	MEP_Fix_eUICCMemoryReset_GetProfileInfo_TstSeq			
		Editor's review	Fixed tables shading in the MEP test sequences			
		CR3110 05R01	ES10c_DisableProfile_MEPA1_A2			
		CR3110 06R01	Fix_MEP_ApplicabilityTable_UpdateMetaDataV3Param			
		CR3110 12	FixRpmDeleteProfile			
		CR3110 14	FixCodingOfEnterpriseRules			
		CR3110 15R01	FixRpmUpdateMetadata			
		CR3110 16R01	FixRpmEnterpriseProfiles			
		CR3110 17	FixMccMnc3			
		CR3110 01r04	MEP_Fix MEP-B Test Sequences			
		CR3110 13R02	FixRpmListProfileInfo			
		CR3110 20	MetadataOpProf10			
		CR3110 21	AddMissingCommands			
		CR3110 22R01	FixRpmMultipleCommands			
		CR3110 23	FixAuthServer			
		CR3110 24	MissingConstants			
		CR3110 25	FixUpdateMetadata			
		CR3110 26	FixRpmMetadata			

		CR3110 28r00	Specify LSI multiplexing			
		CR3110 10	UpdateMetadataEnterpriseConfig			
		CR3110 31r01	Reduce cognitive load - What's a procedure			
SGP.23-1 v3.1.1 Draft 3	20 December 2023	CR3110 29r00	Clarify management of Logical Channels	eUICC		Guido Abate, STMicroelectronics
		CR3110 32r03	Retrofit CR311001 on MEP-B Test Sequences for Disable			
		CR3110 34R00	Refactoring of procedural methods is FFS			
		CR3110 35R03	Retrofit CR311001 on Error Test Sequences			
		CR3110 36R03	Additional fixes MEP tests			
		CR3110 37R01	Revising Applicability Table regarding PPR1 in MEP			
		Editor's review	Reviewed Applicability table for consistency: - "test sequence" was sometimes written with initials uppercase -> all in lowercase now - "sequence" was often singular in spite it indicated more than one - "sequence" and "test sequence" were both used. Now "test sequence" is consistently used - "test sequence" is sometimes preceded by "the", some other times it is not. The article is now consistently used.			
		CR3110 38	Fix_MTD_RES_RPR_FOR_SINGLE_CMND			
		CR3110 39	PoR_for_UpdateMetadata			
		CR3110 40	Select_ISDR			
		Editor's review	- Removed empty lines from tables in annexes			

			- Removed a duplicated sentence in section 2.2.3.5 - Fixed an implementation error of CR2343R01 in section 2.2.3.4 - Fixed font sizes in several parts of the document			
SGP.23-1 v3.1.2	26 April 2024	Editor's review	Same content as SGP.23-1 v3.1.1 Draft 3. All changes accepted	eUICC		Guido Abate, STMicroelectronics
		a	Revising Applicability Table regarding PPR1 in MEP (Note: in v3.1.1 Draft 3 R01 was implemented)	eUICC		Guido Abate, STMicroelectronics
		Editor's review	Fixed an implementation error of CR311040 "Select_ISDR" in section 4.2.1.2.1			
		CR312003r00	Typo in Refresh mode 'UICC Reset'			
		CR312004r01	Clarify definition of IUT_MEP_MODE			
		Editor's review	Changed first page as per new GSMA template	eUICC		Guido Abate, STMicroelectronics
		CR312002R02	Fix_MTD_MEP_TERMINAL_PROFILE			
		CR312008R00	RevertRpmTestMetadata			
		CR312001R07	Reduce Cognitive Load - Fix MEP Procedures Methods Constants			
		CR312005R01	Adjust applicability table			
		CR312011R01	Align ATR+ISDR Selection with methods and procedures			
		CR312013R00	Cleanup target port from MEP-A2 command			
		CR312014R00	Fix empty command in MEP testing			

		CR3120 07R00	Java Card Optionality SGP23-1 eUICC v3.1.2	eUICC	eSIMWG3	Guido Abate, STMicroelectronics
		CR3120 09r02	Align disableProfile with methods and procedures			
		CR3120 12r01	Align eUICCMemoryReset with methods and procedures			
		CR3120 16R01	Fix leftovers of method renaming	eUICC	eSIMWG3	Guido Abate, STMicroelectronics
		CR3120 17R04	Fix_AuthServerTestSequence Header			
		CR3120 18R00	Make_MEP_TestSequence_FlexibleFor_T0_T1			
		CR3120 21R02	MEP Cleanup			
		CR3120 22r03	Support Variants of server Certificates chains			
		CR3120 23R01	Fix_MEP-Flexisibility_EnableProfile			
		CR3120 24R00	Fix_MEP-Flexisibility_DisableProfile			
		CR3120 25R00	Fix_MEP-Flexisibility_eUICCMemoryReset			
		CR3120 26R00	Fix_MEP-Flexisibility_GetProfileInfo_ATR			
		CR3120 27r00	Fixes_in_AnnexE			
		CR3120 28R00	Remove O_E_SERVICE_SPECIFIC_DATA_IN_PM			
		CR3120 029R01	Fix Variant OV3 on the conditions			
		CR3120 030R00	Fix condtions on applicability table			
		Editor's review	Fixed #IUT_RSP_VERSION_HIGHEST that is now set to v3.1 as agreed during eSIMWG3#107			
		CR3120 20R00	Fix_GetEuiccConfiguredDataResponse			

L.2 Other Information

Type	Description
Document Owner	Yolanda Sanz

Editor / Company	GSMA

It is our intention to provide a quality product for your use. If you find any errors or omissions, please contact us with your comments. You may notify us at prd@gsma.com.

Your comments or suggestions & questions are always welcome.