SGP.33-1 eSIM IoT Test Specification for the eUICC

Version 1.0

26 January 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 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.

Compliance Notice

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

This Permanent Reference Document is classified by GSMA as an Industry Specification, as such it has been developed and is maintained by GSMA in accordance with the provisions set out in GSMA AA.35 - Procedures for Industry Specifications.

Table of Contents

1 Introduction

1.1 Overview

The main aim of the eSIM IoT specifications [2] & [3] is to provide solution for the Remote SIM Provisioning of IoT Devices.

This Test Plan provides a set of test cases to be used for testing the implementations of the provisioning system specifications documents [2] & [3]. This document offers to the involved entities an unified test strategy and ensures interoperability between different implementations.

1.2 Scope

This document is intended for:

- Parties which develop test tools and platforms
- Vendors (Device and eUICC Manufacturers, SM-DP+ and SM-DS Providers)
- Operators

The Test Plan consists of a set of relevant test cases for testing the eUICC. The Implementations Under Test (IUT) are:

- the eUICC

The testing scopes developed in this document are:

- Interface compliance testing: Test cases to verify the compliance of the interfaces within the system.
- System behaviour testing: Test cases to verify the functional behaviour of the system.

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 and SGP.32:

- GSMA RSP Technical Specification [4]
- GSMA IoT eSIM Technical Specification [3]

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

1.3 Definition of Terms

In addition to the terms which are defined below, the terms defined in SGP.22 [4] and SGP.32 [3] also apply

Term	Description
Integrated eUICC Test Interface	An external interface provided by its manufacturer for the purpose of testing eUICC functionality.
Test Plan	Current document describing the test cases that allow the eUICC to be tested.

1.4 Abbreviations

In addition to the abbreviations which are defined below, the abbreviations defined in SGP.22 [4] and SGP.32 [3] also apply

Abbreviation	Description
APDU	Application Protocol Data Unit
ATR	Answer To Reset
C-APDU	Command APDU
CCID	(USB) Chip Card Interface Device
DER TLV	Distinguished Encoding Rules - Tag Length Value
FCP	File Control Parameters
HW	Hardware
IUT	Implementation Under Test
KVN	Key Version Number
OCE	Off-Card Entity
OS	Operating System
PIR	Profile Installation Result
POR	Proof Of Receipt
R-APDU	Response APDU
SoC	System on a Chip
SP	Service Provider
SSD	Supplemental Security Domain
USB	Universal Serial Bus

1.5 Document Cross-references

Ref	Document Number	Title
[1]	AA.35	Procedures for Industry Specification version 4.0
[2]	SGP.31	eSIM IoT Architecture and Requirement Specification Version 1.1
[3]	SGP.32	eSIM IoT Technical Specification Version 1.0
[4]	SGP.22	RSP Technical Specification V2.5
[5]	SGP.21	RSP Architecture V2.5
[6]	eUICC Profile Package	Trusted Connectivity Alliance (formerly SIMalliance) eUICC Profile Package: Interoperable Format Technical Specification V2.1 or later

Ref	Document Number	Title
[7]	SGP.26	RSP Test Certificates Definition v3.0.1
[8]	SGP.23	RSP Test Specification v1.14
[5]	ETSI TS 102 221	Smart Cards; UICC-Terminal interface
[6]	GPC_SPE_034	GlobalPlatform Card Specification v2.3
[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
[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
[21]	Void	
[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 Version 3.3.1

Ref	Document Number	Title
[24]	RFC 4492	Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS)
[25]	SGP.26	RSP Test Certificates Definition v1.X
[26]	3GPP TS 29.002	Mobile Application Part (MAP) specification
[27]	RFC 5246	The Transport Layer Security (TLS) Protocol Version 1.2
[28]	GSMA PRD AA.35	Procedures for Industry Specifications Product
[29]	CCID Rev 1.1	CCID Specification for Integrated Circuit(s) Cards Interface Devices

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 5 have 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.
Roles	SM-DP+, SM-DS, Device, LPAd, LPAe or eUICC Entities under test that take in charge the functions used in the test case.
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.

NOTE: this table should contain the eUICC Options dedicated to IoT test cases. Those test cases that are applicable as SGP.23 [8] test cases should use the eUICC Options as defined by SGP.23 [8].

eUICC Options	Mnemonic
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 supports the IPAe	O_E_IPAe
-- as defined in SGP.23 [8]	O_E_2_PIR
-- as defined in SGP.23 [8]	O_E_INTEGRATED

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	Role	SGP.33-1 V1.0	Test Env.
4.2.1.2.1	TC_eUICC_ATR_And_ISDR_Selection	eUICC	SGP.23	TE_eUICC
4.2.2.2.1	TC_eUICC_ES6.UpdateMetadata	eUICC	SGP.23	TE_eUICC
4.2.3.2.1	TC_eUICC_ES8+.InitialiseSecureChannel	eUICC	SGP.23	TE_eUICC
4.2.4.2.1	TC_eUICC_ES8+.ConfigureISDP	eUICC	SGP.23	TE_eUICC
4.2.5.2.1	TC_eUICC_ES8+.StoreMetadata	eUICC	SGP.23	TE_eUICC

Test case	Name	Role	SGP.33-1 V1.0	Test Env.
4.2.5.2.2	TC_eUICC_ES8+.StoreMetadata_Service_Specific_Data	eUICC	SGP.23	TE_eUICC
4.2.6.2.1	TC_eUICC_ES8+.ReplaceSessionKeys	eUICC	SGP.23	TE_eUICC
4.2.7.2.1	TC_eUICC_ES8+.LoadProfileElements	eUICC	SGP.23	TE_eUICC
4.2.8.2.1	TC_eUICC_ES10a.GetEuiccConfiguredAddresses	eUICC	C3004	TE_eUICC
4.2.9.2.1	TC_eUICC_ES10a.SetDefaultDPAddress	eUICC	C3004	TE_eUICC
4.2.10.2.1	TC_eUICC_ES10b.PrepareDownloadNIST	eUICC	C3004	TE_eUICC
4.2.10.2.2	TC_eUICC_ES10b.PrepareDownloadBRP	eUICC	C3004	TE_eUICC
4.2.10.2.3	TC_eUICC_ES10b.PrepareDownloadFRP	eUICC	C3004	TE_eUICC
4.2.10.2.4	TC_eUICC_ES10b.PrepareDownloadErrorCases	eUICC	C3004	TE_eUICC
4.2.11.2.1	TC_eUICC_ES10b.LoadBoundProfilePackageNIST	eUICC	C3004	TE_eUICC
4.2.11.2.2	TC_eUICC_ES10b.LoadBoundProfilePackageBRP	eUICC	C3004	TE_eUICC
4.2.11.2.3	TC_eUICC_ES10b.LoadBoundProfilePackageFRP	eUICC	C3004	TE_eUICC
4.2.11.2.4	TC_eUICC_ES10b.LoadBoundProfilePackage_ErrorCases	eUICC	C3004	TE_eUICC
4.2.12.2.1	TC_eUICC_ES10b.GetEUICCChallenge	eUICC	C3004	TE_eUICC
4.2.13.2.1	TC_eUICC_ES10b.GetEUICCInfo1	eUICC	C3004	TE_eUICC
4.2.13.2.2	TC_eUICC_ES10b.GetEUICCInfo2_RSP_Integrated_eUICC	eUICC	C3004	TE_eUICC
4.2.13.2.4	TC_eUICC_ ES10b.GetEUICCInfo2_v1.0	eUICC	C3004	TE_eUICC
4.2.14.2.1	TC_eUICC_ES10b.ListNotification All test sequences except the sequence #5	eUICC	C3005	TE_eUICC
4.2.14.2.1	TC_eUICC_ES10b.ListNotification Only the test sequence #5	eUICC	C025	TE_eUICC
4.2.15.2.1	TC_eUICC_ES10b.RetrieveNotificationsList All test sequences except the sequences #5 and #15	eUICC	SGP.23	TE_eUICC
4.2.15.2.1	TC_eUICC_ES10b.RetrieveNotificationsList Only the test sequences #5 and #15	eUICC	SGP.23	TE_eUICC
4.2.16.2.1	TC_eUICC_ES10b.RemoveNotificationFromList All test sequences except the sequence #5	eUICC	C3004	TE_eUICC

Test case	Name	Role	SGP.33-1 V1.0	Test Env.
4.2.16.2.1	TC_eUICC_ES10b.RemoveNotificationFromList Only the test sequence #5	eUICC	C3004	TE_eUICC
4.2.16.2.2	TC_eUICC_ES10b. eUICCPackageResultsList	eUICC	C3004	Te_eUICC
4.2.18.2.1	TC_eUICC_ES10b.AuthenticateServer_SM-DP+_NIST	eUICC	C3004	TE_eUICC
4.2.18.2.2	TC_eUICC_ES10b.AuthenticateServer_SM-DP+_BRP	eUICC	C3004	TE_eUICC
4.2.18.2.3	TC_eUICC_ES10b.AuthenticateServer_SM-DP+_FRP	eUICC	C3004	TE_eUICC
4.2.18.2.4	TC_eUICC_ES10b.AuthenticateServer_SM-DP+_ErrorCases	eUICC	C3004	TE_eUICC
4.2.18.2.5	TC_eUICC_ES10b.AuthenticateServer_SM-DS_BRP	eUICC	C3004	TE_eUICC
4.2.18.2.6	TC_eUICC_ES10b.AuthenticateServer_SM-DS_NIST	eUICC	C3004	TE_eUICC
4.2.18.2.7	TC_eUICC_ES10b.AuthenticateServer_SM-DS_FRP	eUICC	C3004	TE_eUICC
4.2.18.2.8	TC_eUICC_ES10b.AuthenticateServer_SM-DS_ErrorCases	eUICC	C3004	TE_eUICC
4.2.19.2.1	TC_eUICC_ES10b.CancelSessionNIST	eUICC	C3004	TE_eUICC
4.2.19.2.2	TC_eUICC_ES10b.CancelSessionBRP	eUICC	C3004	TE_eUICC
4.2.19.2.3	TC_eUICC_ES10b.CancelSessionFRP	eUICC	C3004	TE_eUICC
4.2.19.2.4	TC_eUICC_ES10b.CancelSession_ErrorCase	eUICC	C3004	TE_eUICC
4.2.19.2.5	TC_eUICC_ES10b.CancelSession_Indirect_Profile_Download	eUICC	C3004	TE_eUICC
4.2.19.2.6	TC_eUICC_ES10b.CancelSession_Indirect_Profile_Download_ErrorCases	eUCC	C3004	TE_eUICC
4.2.20.2.1	TC_eUICC_ES10b.eUICCMemoryReset	eUICC	C3004	TE_eUICC
4.2.20.2.2	TC_eUICC_ES10b.eUICCMemoryReset_ErrorCases	eUICC	C3004	TE_eUICC
4.2.21.2.1	TC_eUICC_ES10b.GetEID	eUICC	C3004	TE_eUICC
4.2.22.2.1	TC_eUICC_ES10b.GetRAT	eUICC	C3004	TE_eUICC
4.2.23.2.1	TC_eUICC_ES10b.GetProfilesInfo	eUICC	C3004	TE_eUICC

Test case	Name	Role	SGP.33-1 V1.0	Test Env.
4.2.24.2.1	TC_eUICC_ES10b. LoadEUICCPackage	eUICC	NA	TE_eUICC
4.2.24.2.2	TC_eUICC_ES10b. LoadEUICCPackage_ErrorCases	eUICC	C3004	TE_eUICC
4.2.25.2.1	TC_eUICC_ES10b.AddInitialEim	eUICC	C3004	TE_eUICC
4.2.25.2.2	TC_eUICC_ES10b.AddInitialEim_ErrorCases	eUICC	C3004	TE_eUICC
4.2.26.2.1	TC_eUICC_ES10b.GetCertificates	eUICC	C3004	TE_eUICC
4.2.26.2.2	TC_eUICC_ES10b.GetCertificates_ErrorCases	eUICC	C3004	TE_eUICC
4.2.27.2.1	TC_eUICC_ES10b. EnableUsingDD	eUICC	C3004	TE_eUICC
4.2.27.2.2	TC_eUICC_ES10b. EnableUsingDD_ErrorCases	eUICC	C3004	TE_eUICC
4.2.28.2.1	TC_eUICC_ES10b.ProfileRollback All test sequences except the sequence #3, sequence #4, sequence #5, and sequence #6	eUICC	C3004	TE_eUICC
4.2.28.2.1	TC_eUICC_ES10b.ProfileRollback Only the sequence #3 and sequence #5	eUICC	C3001	TE_eUICC
4.2.28.2.1	TC_eUICC_ES10b.ProfileRollback Only the sequence #4 and sequence #6	eUICC	C3003	TE_eUICC
4.2.28.2.2	TC_eUICC_ES10b.ProfileRollback_ErrorCases All test sequences except the sequence #3, and sequence #4	eUICC	C3004	TE_eUICC
4.2.28.2.2	TC_eUICC_ES10b.ProfileRollback_ErrorCases Only sequence #3	eUICC	C3000	TE_eUICC
4.2.28.2.2	TC_eUICC_ES10b.ProfileRollback_ErrorCases Only sequence #4	eUICC	C3002	TE_eUICC
4.2.29.2.1	TC_eUICC_ES10b ConfigureAutomaticProfileEnabling	eUICC	C3004	TE_eUICC
4.2.29.2.2	TC_eUICC_ES10b ConfigureAutomaticProfileEnabling_ErrorCases	eUICC	C3004	TE_eUICC
4.2.30.2.1	TC_eUICC_ES10b.GetEimConfigurationData	eUICC	C3004	TE_eUICC
4.2.31.2.1	TC_eUICC_ESep.Enable	eUICC	M	TE_eUICC
4.2.31.2.2	TC_eUICC_ESep.Enable_ErrorCases	eUICC	M	TE_eUICC
4.2.32.2.1	TC_eUICC_ESep.Disable	eUICC	M	TE_eUICC

Test case	Name	Role	SGP.33-1 V1.0	Test Env.
4.2.32.2.2	TC_eUICC_ESep.Disable_ErrorCases	eUICC	M	TE_eUICC
4.2.33.2.1	TC_eUICC_ESep.Delete	eUICC	M	TE_eUICC
4.2.33.2.2	TC_eUICC_ESep.Delete_ErrorCases	eUICC	M	TE_eUICC
4.2.34.2.1	TC_eUICC_ESep.ListProfileInfo	eUICC	M	TE_eUICC
4.2.34.2.2	TC_eUICC_ESep.ListProfileInfo_ErrorCases	eUICC	M	TE_eUICC
4.2.35.2.1	TC_eUICC_ESep.GetRat	eUICC	M	TE_eUICC
4.2.36.2.1	TC_eUICC_ESep.AddEIM	eUICC	M	TE_eUICC
4.2.36.2.2	TC_eUICC_ESep.AddEIM_ErrorCases	eUICC	M	TE_eUICC
4.2.37.2.1	TC_eUICC_ESep.UpdateEIM	eUICC	M	TE_eUICC
4.2.37.2.2	TC_eUICC_ESep.UpdateEIM_ErrorCases	eUICC	M	TE_eUICC
4.2.38.2.1	TC_eUICC_ESep.DeleteEIM	eUICC	M	TE_eUICC
4.2.38.2.2	TC_eUICC_ESep.DeleteEIM_ErrorCases	eUICC	M	TE_eUICC
4.2.39.2.1	TC_eUICC_ESep.ListEIM	eUICC	M	TE_eUICC
4.2.39.2.2	TC_eUICC_ESep.ListEIM_ErrorCases	eUICC	NA	NA
Test Specifications				
7.1	TCA eUICC Profile Package Test Specification	eUICC	SGP.23	See section 7.1

Table 5: Applicability of Tests

Conditional item	Condition
C3000	IF (O_E_CATBUSY_EN_DIS_REFRESH AND NOT O_E_IPAe) THEN M ELSE N/A
C3001	IF (NOT O_E_CATBUSY_EN_DIS_REFRESH AND NOT O_E_IPAe) THEN M ELSE N/A
C3002	IF (O_E_CATBUSY_EN_DIS_NOREFRESH AND NOT O_E_IPAe) THEN M ELSE N/A

Conditional item	Condition
C3003	IF (NOT O_E_CATBUSY_EN_DIS_NOREFRESH AND NOT O_E_IPAe) THEN M ELSE N/A
C3004	IF (NOT O_E_IPAe) THEN M ELSE N/A
C3005	IF (O_E_2_PIR AND NOT O_E_IPAe) THEN M ELSE N/A
C3006	IF (O_E_INTEGRATED AND NOT O_E_IPAe) THEN M ELSE N/A

Table 6: Conditional Items Referenced by Table 5

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 all entities under test 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.

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).

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

In addition, following 2.2.1 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.

2.2.2.1 Methods and Procedures

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 is a generic sub-sequence and is referenced as follow:

- `PROC_NAME_OF_THE_PROCEDURE`

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 also 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.

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 [X].

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, Device and eUICC manufacturers and Platforms (i.e. SM-DP+, SM-DS) providers 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.

2.2.3 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.

2.2.3.2 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.

The highest priority CI in euiccCiPKIdListForSigning SHALL be used.

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
- #AUTH_SMDP_MATCH_ID
    - with the <EUICC_CI_PK_ID_TO_BE_USED> set to the CI for signing indicated as highest priority in the #R_EUICC_INFO1
    - with the #CERT_S_SM_DPauth_ECDSA leading to the same CI as the one chosen for signing
    - with the SM-DP+ address #TEST_DP_ADDRESS1
- #PREP_DOWNLOAD_NO_CC
    - with the #CERT_S_SM_DPpb_ECDSA 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>)

2.2.3.3 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.

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).

The DeleteProfileRequest SHALL contain the following parameter:

- ICCID of the Profile to Delete

2.2.3.4 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:

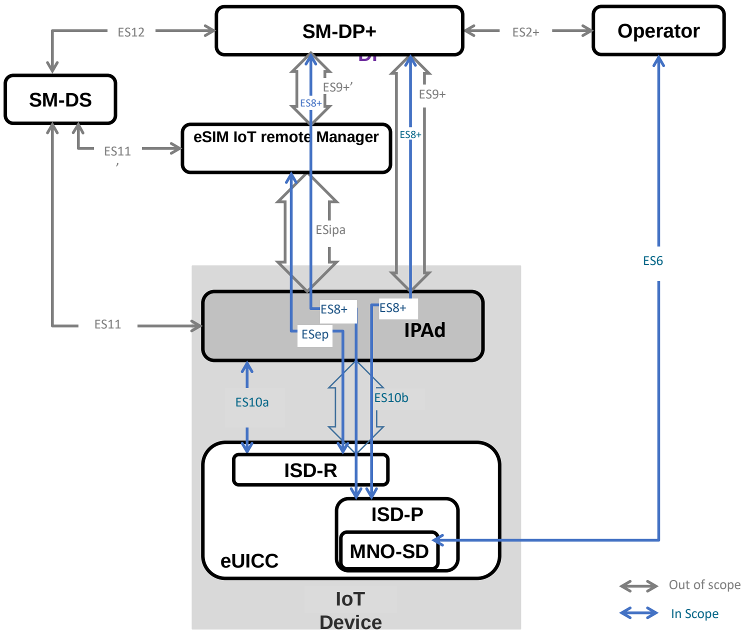
- `notificationMetadataList` of `ListNotificationResponse`
- `profileInfoListOk` of `ProfileInfoListResponse`
- `notificationList` of `RetrieveNotificationsListResponse`

When an Operational Profile class is expected to be indicated in a `ProfileInfoListResponse`, the Test Tool SHALL accept only one DER encoding if the eUICC supports SGP.22 v2.2.x [2] or SGP.22 V2.2 [2b]: a tag 0x95 containing the integer value 2.

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	SGP.33-1
ES2+	Operator	SM-DP+	Used by the Operator to order Profiles for specific eUICCs as well as other administrative functions as defined in SGP.31 [2].	Out of scope
ES6	Operator	eUICC	Used by the Operator for the management of Operator services via OTA services as defined in SGP.31 [2].	In scope
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 as defined in SGP.31 [2].	In scope
ES9+	SM-DP+	IPA	Used to provide a secure transport between the SM-DP+ and the IPA for the delivery of the Bound Profile Package as defined in SGP.31 [2].	Out of scope
ES9+'	SM-DP+	eIM	Used to provide a secure transport between the SM-DP+ and the eIM for the delivery of the Bound Profile Package as defined in SGP.31 [2].	Out of scope
ES10a	IPA	eUICC	Used between the IPA (in the IoT Device) and the eUICC to handle a Profile discovery as defined in SGP.31 [2].	In scope
ES10b	IPA	eUICC	Used between the IPA (in the IoT Device) and the IPA Services to transfer a Bound Profile Package to the eUICC as defined in SGP.31 [2]. This interface plays no role in the decryption of Profile Packages.	In scope

Interface	Between		Description	SGP.33-1
ES11	IPA	SM-DS	Used by the IPA to retrieve Event Records for the respective eUICC as defined in SGP.31 [2].	Out of scope
ES11’	eIM	SM-DS	Used by the eIM to retrieve Event Records for the respective eUICC as defined in SGP.31 [2].	Out of scope
ES12	SM-DP+	SM-DS	Used by the SM-DP+ to issue or remove Event Registrations on the SM-DS as defined in SGP.31 [2].	Out of scope
ESep	eIM	eUICC	Logical end-to-end interface between the eIM and the eUICC used to transfer eUICC Packages for Profile State management and eIM configuration by eIM, as defined in SGP.31 [2].	In scope
ESipa	eIM	IPA	Logical interface between an eIM and an IPA, as defined in SGP.31 [2], used to trigger a Profile download at the IPA and to provide a secure transport for the delivery of eUICC Packages, unless the underlying transport provides necessary security.	Out of scope

Table 7: Interfaces Descriptions

NOTE: Support of the ES10c interface as defined in SGP.22 [4] is out of scope of this specification.

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_IoT_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_IPAd: the IPAd Simulator
- S_eIM: the eIM 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:

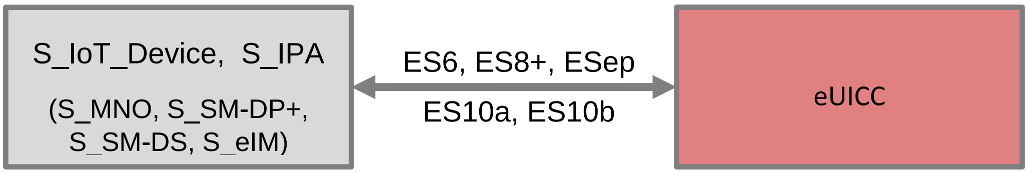
- `S_ComponentName` for a simulated component
- `ComponentName` for the Implementation Under Test (IUT)

- 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 compliant with Annex G.1 – eUICC Initial States
- IPAd / MNO / SM-DP+ / SM-DS / eIM / IoT Device Simulators SHALL be implemented by the test tools

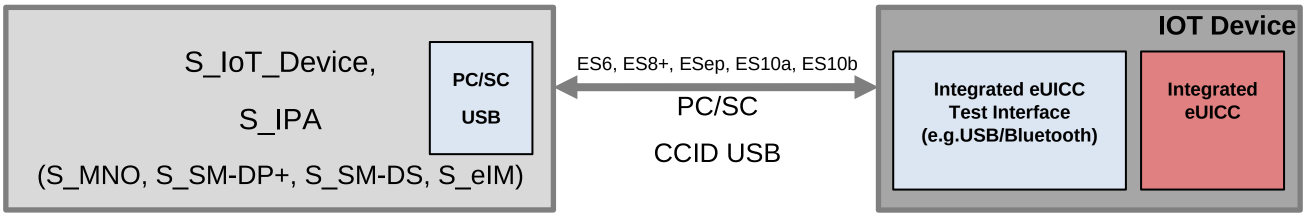


The reference of this Test Environment is TE_eUICC.

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.1 – eUICC Initial States
- IPAd / MNO / SM-DP+ / SM-DS / eIM / IoT Device Simulators SHALL be implemented by the test tools
- Integrated eUICC shall provide a test interface which includes one of the following:
  - ISO/IEC 7816-4 [7]
  - USB CCID [29]
- For Integrated eUICC providing a USB CCID [29] 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


S_IoT_Device,
S_IPA
(S_MNO, S_SM-DP+, S_SM-DS, S_eIM)
PC/SC
USB
ES6, ES8+, ESep, ES10a, ES10b
PC/SC
CCID USB
IOT Device
Integrated eUICC Test Interface (e.g.USB/Bluetooth)
Integrated eUICC

The reference of this [29] 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 eSIM IoT Technical Specification [3]. The aim is to verify the compliance of all interfaces within the system.

4.2 eUICC Interfaces

4.2.1 ATR and ISD-R Selection

4.2.1.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specificaiton [3]

4.2.1.2 Test Cases

4.2.1.2.1 TC_eUICC_ATR_And_ISDR_Selection

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal: ATR and Select ISD-R* defined in section 4.2.1.2.1 TC_eUICC_ATR_And_ISDR_Selection. Where the S_LPAd is playing the role of the S_IPA.

4.2.2 ES6 (Operator -- eUICC): UpdateMetadata

4.2.2.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specificaiton [3]

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

4.2.2.2 Test Cases

4.2.2.2.1 TC_eUICC_ES6.UpdateMetadata

ES6 interface is identical to the one defined in section 5.4 of SGP.22 [4].

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.

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: Unset PPR1* defined in section 4.2.2.2.1 TC_eUICC_ES6.UpdateMetadata

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.

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.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #03 Nominal: Unset PPR1 and PPR2 and update Profile name and Service Provider name* defined in section 4.2.2.2.1 TC_eUICC_ES6.UpdateMetadata

Test Sequence #04 Nominal: Delete PPRs, Service Provider and Profile names

The purpose of this test is to verify that the MNO can delete all PPRs, the Service Provider name and the Profile name from a Profile.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #04 Nominal: Delete PPRs, Service Provider and Profile names* defined in section 4.2.2.2.1 TC_eUICC_ES6.UpdateMetadata

Test Sequence #05 Nominal: Delete icon

This test sequence of this errro is not applicable for this version of the specification because is indicated for FFS in SGP.23 [8]

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.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #06 Nominal: Delete Unset PPRs* defined in section 4.2.2.2.1 TC_eUICC_ES6.UpdateMetadata

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.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #07 Error: Set a pprUpdateControl value to one* defined in section 4.2.2.2.1 TC_eUICC_ES6.UpdateMetadata

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.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #08 Error: Update Metadata on a Disable Profile* defined in section 4.2.2.2.1 TC_eUICC_ES6.UpdateMetadata

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.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #09 Error: Empty request* defined in section 4.2.2.2.1 TC_eUICC_ES6.UpdateMetadata

Test Sequence #10 Error: Update Icon without Icon Type field

The purpose of this test is to verify that the eUICC is correctly rejecting an Update Metadata request from the MNO when the icon field is present but not the icon type field.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #10 Error: Update Icon without Icon Type field* defined in section 4.2.2.2.1 TC_eUICC_ES6.UpdateMetadata

Test Sequence #11 Error: Update Icon Type without Icon field

The purpose of this test is to verify that the eUICC is correctly rejecting an Update Metadata request from the MNO when the Icon Type field is present but not the Icon field.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #11 Error: Update Icon Type without Icon field* defined in section 4.2.2.2.1 TC_eUICC_ES6.UpdateMetadata

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

4.2.3.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.2.3.2 Test Cases

4.2.3.2.1 TC_eUICC_ES8+.InitialiseSecureChannel

All the functions related to ES8+ SHALL be processed as defined in section 5.5 of SGP.22 [4]. They are called by the SM-DP+ and executed by the eUICC.

Test Sequence #01 Error: Invalid Remote Operation

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Error: Invalid Remote Operation* defined in section 4.2.3.2.1 TC_eUICC_ES8+.InitialiseSecureChannel

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Error: Invalid SM-DP+ Signature* defined in section 4.2.3.2.1 TC_eUICC_ES8+.InitialiseSecureChannel

Test Sequence #03 Error: Invalid Transaction Identifier

This test sequence is the same as SGP.23 [8] - the *Test Sequence #03 Error: Invalid Transaction Identifier* defined in section 4.2.3.2.1 TC_eUICC_ES8+.InitialiseSecureChannel

Test Sequence #04 Error: Invalid CRT Values

This test sequence is the same as SGP.23 [8] - the *Test Sequence #04 Error: Invalid CRT Values* defined in section 4.2.3.2.1 TC_eUICC_ES8+.InitialiseSecureChannel

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

This test sequence is the same as SGP.23 [8] – the *Test Sequence #05 Error: InitialiseSecureChannel request while Secure Channel Session* is ongoing defined in section 4.2.3.2.1 TC_eUICC_ES8+.InitialiseSecureChannel

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

All the functions related to ES8+ SHALL be processed as defined in section 5.5 of SGP.22 [4]. They are called by the SM-DP+ and executed by the eUICC. Therefore, the Test Sequence referencing int this section can be applied as defined by SGP.23

4.2.4.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.2.4.2 Test Cases

4.2.4.2.1 TC_eUICC_ES8+.ConfigureISDP

All the functions related to ES8+ SHALL be processed as defined in section 5.5 of SGP.22 [4]. They are called by the SM-DP+ and executed by the eUICC.

Test Sequence #01 Nominal: Empty Proprietary Data

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal: Empty Proprietary Data* defined in section 4.2.4.2.1 TC_eUICC_ES8+.ConfigureISDP

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Nominal: Proprietary Data with the maximum length authorized (i.e. 128 bytes)* defined in section 4.2.4.2.1 TC_eUICC_ES8+.ConfigureISDP

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #03 Error: Proprietary Data with the maximum length exceeded (i.e. 129 bytes)* defined in section 4.2.4.2.1 TC_eUICC_ES8+.ConfigureISDP

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

4.2.5.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.2.5.2 Test Cases

4.2.5.2.1 TC_eUICC_ES8+.StoreMetadata

All the functions related to ES8+ SHALL be processed as defined in section 5.5 of SGP.22 [4]. They are called by the SM-DP+ and executed by the eUICC.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal: All Metadata fields present (PNG icon used and PPR1 set)* defined in section 4.2.5.2.1 TC_eUICC_ES8+.StoreMetadata

Test Sequence #02 Nominal: With JPG icon

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Nominal: With JPG icon* defined in section 4.2.5.2.1 TC_eUICC_ES8+.StoreMetadata.

Test Sequence #03 Nominal: Without providing Profile Class

This test sequence is the same as SGP.23 [8] - the *Test Sequence #03 Nominal: Without providing Profile Class* defined in section 4.2.5.2.1 TC_eUICC_ES8+.StoreMetadata

Test Sequence #04 Nominal: With PPR2 set

This test sequence is the same as SGP.23 [8] - the *Test Sequence #04 Nominal: With PPR2 set* defined in section 4.2.5.2.1 TC_eUICC_ES8+.StoreMetadata

Test Sequence #05 Nominal: With PPR1 and PPR2 set

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal: All Metadata fields present (PNG icon used and PPR1 set)* defined in section 4.2.5.2.1 TC_eUICC_ES8+.StoreMetadata

This test sequence is the same as SGP.23 [8] - the *Test Sequence #05 Nominal: With PPR1 and PPR2 set* defined in section 4.2.5.2.1 TC_eUICC_ES8+.StoreMetadata

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #07 Error: ICCID already present in the eUICC* defined in section 4.2.5.2.1 TC_eUICC_ES8+.StoreMetadata

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #08 Error: Profile Policy Rule is set but Profile Owner is not* defined in section 4.2.5.2.1 TC_eUICC_ES8+.StoreMetadata

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #09 Error: Profile Owner is set with a wildcard ('E') digits* defined in section 4.2.5.2.1 TC_eUICC_ES8+.StoreMetadata

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #10 Error: Icon Type is set but icon is not* defined in section 4.2.5.2.1 TC_eUICC_ES8+.StoreMetadata

4.2.5.2.2 TC_eUICC_ES8+.StoreMetadata_Service_Specific_Data

All the functions related to ES8+ SHALL be processed as defined in section 5.5 of SGP.22 [4]. They are called by the SM-DP+ and executed by the eUICC.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal: Metadata include service-specific data, stored* defined in section 4.2.5.2.2 TC_eUICC_ES8+.StoreMetadata_Service_Specific_Data

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Nominal: Metadata include service-specific data, not stored* defined in section 4.2.5.2.2 TC_eUICC_ES8+.StoreMetadata_Service_Specific_Data

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #03 Nominal: Metadata include service-specific data, stored and not stored* defined in section 4.2.5.2.2 TC_eUICC_ES8+.StoreMetadata_Service_Specific_Data

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

All the functions related to ES8+ SHALL be processed as defined in section 5.5 of SGP.22 [4]. They are called by the SM-DP+ and executed by the eUICC.

4.2.6.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specificaiton [3]

4.2.6.2 Test Cases

4.2.6.2.1 TC_eUICC_ES8+.ReplaceSessionKeys

All the functions related to ES8+ SHALL be processed as defined in section 5.5 of SGP.22 [4]. They are called by the SM-DP+ and executed by the eUICC.

Test Sequence #01 Error: Incorrect PPK size

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Error: Incorrect PPK size* defined in section *4.2.6.2.1 TC_eUICC_ES8+.ReplaceSessionKeys.*

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

4.2.7.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specificaiton [3]

4.2.7.2 Test Cases

4.2.7.2.1 TC_eUICC_ES8+.LoadProfileElements

All the functions related to ES8+ SHALL be processed as defined in section 5.5 of SGP.22 [4]. They are called by the SM-DP+ and executed by the eUICC.

Test Sequence #01 Error: EFICCID different from the ICCID provided in the Profile Metadata

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Error: EFICCID different from the ICCID provided in the Profile Metadata* defined in section *4.2.7.2.1 TC_eUICC_ES8+.LoadProfileElements.*

Test Sequence #02 Error: MCC / MNC of EFIMSI different from MCC / MNC of Profile Owner present in Metadata

This test sequence is the same as SGP.23 [8] - the Test Sequence #02 Error: MCC / MNC of EFIMSI different from MCC / MNC of Profile Owner present in Metadata defined in section *4.2.7.2.1 TC_eUICC_ES8+.LoadProfileElements.*

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

This test sequence is the same as SGP.23 [8] - the Test Sequence #03 Error: Session MAC chaining used instead of new Initial MAC chaining defined in section *4.2.7.2.1 TC_eUICC_ES8+.LoadProfileElements.*

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #04 Error: S-MAC used instead of PPK-MAC* defined in section *4.2.7.2.1 TC_eUICC_ES8+.LoadProfileElements.*

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #05 Error: S-ENC used instead of PPK-ENC* defined in section *4.2.7.2.1 TC_eUICC_ES8+.LoadProfileElements.*

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #06 Error: Profile Downloading stopped by a Reset* defined in section *4.2.7.2.1 TC_eUICC_ES8+.LoadProfileElements.*

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #07 Nominal: ICCID in the 'ProfileHeader' PE is ignored by the eUICC* defined in section *4.2.7.2.1 TC_eUICC_ES8+.LoadProfileElements.*

Test Sequence #08 Nominal: With gid1 and gid2 set

This test sequence is the same as SGP.23 [8] - the *Test Sequence #08 Nominal: With gid1 and gid2 set* defined in section 4.2.7.2.1 TC_eUICC_ES8+.LoadProfileElements

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #09 Error: gid1 and gid2 provided in the Profile Metadata but not in the Profile Package* defined in section 4.2.7.2.1 TC_eUICC_ES8+.LoadProfileElements

Test Sequence #10 Error: gid1 and gid2 not provided in the Profile Metadata but present in Profile Package

This test sequence is the same as SGP.23 [8] - the *Test Sequence #10 Error: gid1 and gid2 not provided in the Profile Metadata but present in Profile Package* defined in section 4.2.7.2.1 TC_eUICC_ES8+.LoadProfileElements

4.2.8 ES10a (IPA -- eUICC): GetEuiccConfiguredAddresses

4.2.8.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.2.8.2 Test Cases

4.2.8.2.1 TC_eUICC_ES10a.GetEuiccConfiguredAddresses

This ES10a.GetEuiccConfiguredAddresses function is identical to the ES10a.GetEuiccConfiguredAddresses function defined in section 5.7.3 of SGP.22 [4], where the IPA plays the role of LPA.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal: Only Root SM-DS Address* defined in 4.2.8.2.1 TC_eUICC_ES10a.GetEuiccConfiguredAddresses, where the S_LPAd plays the role of S_IPA.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Nominal: Root SM-DS and Default SM-DP+ Addresses* defined in 4.2.8.2.1 TC_eUICC_ES10a.GetEuiccConfiguredAddresses, where the S_LPAd plays the role of S_IPA.

4.2.9 ES10a (IPA -- eUICC): SetDefaultDPAddress

4.2.9.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.2.9.2 Test Cases

4.2.9.2.1 TC_eUICC_ES10a.SetDefaultDPAddress

This ES10a.SetDefaultDPAddress function is identical to the ES10a. SetDefaultDPAddress function defined in section 5.7.4 of SGP.22 [4], where the IPA plays the role of LPA.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal: Set SM-DP+ Address with Address Empty in eUICC* defined in 4.2.9.2.1 TC_eUICC_ES10a.SetDefaultDPAddress, where the S_LPAd plays the role of S_IPA.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Nominal: Set SM-DP+ Address with SM-DP+ Address already in eUICC* defined in 4.2.9.2.1 TC_eUICC_ES10a.SetDefaultDPAddress, where the S_LPAd plays the role of S_IPA.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #03 Nominal: Set Empty SM-DP+ Address with SM-DP+ Address already in eUICC* defined in 4.2.9.2.1 TC_eUICC_ES10a.SetDefaultDPAddress, where the S_LPAd plays the role of S_IPA.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #04 Nominal: Set Empty SM-DP+ Address with Empty SM-DP+ Address in eUICC* defined in 4.2.9.2.1 TC_eUICC_ES10a.SetDefaultDPAddress, where the S_LPAd plays the role of S_IPA.

4.2.10 ES10b (IPA -- eUICC): PrepareDownload

4.2.10.1 Conformance Requirements

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.2.10.2 Test Cases

4.2.10.2.1 TC_eUICC_ES10b.PrepareDownloadNIST

This ES10b.PrepareDownload function is identical to the ES10b.PrepareDownload function defined in section 5.7.5 of SGP.22 [4], where the IPA plays the role of LPA.

Test Sequence #01 Nominal: Without Confirmation Code

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal: Without Confirmation Code* defined in 4.2.10.2.1 TC_eUICC_ES10b.PrepareDownloadNIST, where the S_LPAd plays the role of S_IPA.

Test Sequence #02 Nominal: With Confirmation Code

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Nominal: With Confirmation Code* defined in 4.2.10.2.1 TC_eUICC_ES10b.PrepareDownloadNIST, where the S_LPAd plays the role of S_IPA.

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.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #03 Nominal: With an unknown otPK.EUICC.ECKA* defined in 4.2.10.2.1 TC_eUICC_ES10b.PrepareDownloadNIST, where the S_LPAd plays the role of S_IPA.

4.2.10.2.2 TC_eUICC_ES10b.PrepareDownloadBRP

This ES10b.PrepareDownload function is identical to the ES10b.PrepareDownload function defined in section 5.7.5 of SGP.22 [4], where the IPA plays the role of LPA.

Test Sequence #01 Nominal: Without Confirmation Code

This test sequence SHALL be the same as SGP.23 [8] - 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 SGP.23 [8] - 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 SGP.23 [8] - 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

This ES10b.PrepareDownload function is identical to the ES10b.PrepareDownload function defined in section 5.7.5 of SGP.22 [4], where the IPA plays the role of LPA.

Sequence #01 Error: No Hashed Confirmation Code but with Confirmation Code Required Flag set to TRUE

This test sequence is the same as SGP.23 [8] - the *Sequence #01 Error: No Hashed Confirmation Code but with Confirmation Code Required Flag set to TRUE* defined in 4.2.10.2.4 TC_eUICC_ES10b.PrepareDownloadErrorCases, where the S_LPAd plays the role of S_IPA.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Error: With incorrect CERT.DPpb.ECDSA (i.e. invalid signature)* defined in 4.2.10.2.4 TC_eUICC_ES10b.PrepareDownloadErrorCases, where the S_LPAd plays the role of S_IPA.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #03 Error: CERT.DPpb.ECDSA and CERT.DPauth.ECDSA not belonging to the same entity* defined in 4.2.10.2.4 TC_eUICC_ES10b.PrepareDownloadErrorCases, where the S_LPAd plays the role of S_IPA.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #04 Error: With invalid SM-DP+ signature* defined in 4.2.10.2.4 TC_eUICC_ES10b.PrepareDownloadErrorCases, where the S_LPAd plays the role of S_IPA.

Test Sequence #05 Error: With invalid Transaction ID

This test sequence is the same as SGP.23 [8] - the *Test Sequence #05 Error: With invalid Transaction ID* defined in 4.2.10.2.4 TC_eUICC_ES10b.PrepareDownloadErrorCases, where the S_LPAd plays the role of S_IPA.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #06 Error: SM-DP+ has not been previously authenticated* defined in 4.2.10.2.4 TC_eUICC_ES10b.PrepareDownloadErrorCases, where the S_LPAd plays the role of S_IPA.

Test Sequence #07 Error: Unsupported curve

This test sequence is the same as SGP.23 [8] - the *Test Sequence #07 Error: Unsupported curve* defined in 4.2.10.2.4 TC_eUICC_ES10b.PrepareDownloadErrorCases, where the S_LPAd plays the role of S_IPA.

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.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #08 Error: Invalid Certificate Role OID* defined in 4.2.10.2.4 TC_eUICC_ES10b.PrepareDownloadErrorCases, where the S_LPAd plays the role of S_IPA.

4.2.11 ES10b (IPA -- eUICC): LoadBoundProfilePackage

4.2.11.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.2.11.2 Test Cases

4.2.11.2.1 TC_eUICC_ES10b.LoadBoundProfilePackageNIST

This ES10b.LoadBoundProfilePackage function is identical to the ES10b.LoadBoundProfilePackage function defined in section 5.7.6 of SGP.22 [4], where the IPA plays the role of LPA.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal: By using S-ENC and S-MAC* defined in 4.2.11.2.1 TC_eUICC_ES10b.LoadBoundProfilePackageNIST, where the S_LPAd plays the role of S_IPA.

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.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Nominal: By using PPK-ENC and PPK-MAC* defined in 4.2.11.2.1 TC_eUICC_ES10b.LoadBoundProfilePackageNIST, where the S_LPAd plays the role of S_IPA.

4.2.11.2.2 TC_eUICC_ES10b.LoadBoundProfilePackageBRP

This ES10b.LoadBoundProfilePackage function is identical to the ES10b.LoadBoundProfilePackage function defined in section 5.7.6 of SGP.22 [4], where the IPA plays the role of LPA.

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

This test sequence SHALL be the same as SGP.23 [8] - 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 SGP.23 [8] - the *Test Sequence #02 Nominal* 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 TC_eUICC_ES10b.LoadBoundProfilePackageFRP

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

4.2.11.2.4 TC_eUICC_ES10b.LoadBoundProfilePackage_ErrorCases

This ES10b.LoadBoundProfilePackage function is identical to the ES10b.LoadBoundProfilePackage function defined in section 5.7.6 of SGP.22 [4], where the IPA plays the role of LPA.

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.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Error: Unrecognized leading tag in BPP* defined in 4.2.11.2.4 TC_eUICC_ES10b.LoadBoundProfilePackage_ErrorCases, where the S_LPAd plays the role of S_IPA.

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.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Error: GetEUICCChallenge during BPP loading* defined in 4.2.11.2.4 TC_eUICC_ES10b.LoadBoundProfilePackage_ErrorCases, where the S_LPAd plays the role of S_IPA.

4.2.12 ES10b (LPA -- eUICC): GetEUICCChallenge

4.2.12.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.2.12.2 Test Cases

4.2.12.2.1 TC_eUICC_ES10b.GetEUICCChallenge

This function is identical to the ES10b.GetEUICCChallenge function defined in section 5.7.7 of SGP.22 [4], where the IPA plays the role of LPA.

Test Sequence #01 Nominal

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal* defined in 4.2.12.2.1 TC_eUICC_ES10b.GetEUICCChallenge, where the S_LPAd plays the role of S_IPA.

4.2.13 ES10b (IPA -- eUICC): GetEUICCInfo

4.2.13.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.2.13.2 Test Cases

4.2.13.2.1 TC_eUICC_ES10b.GetEUICCInfo1

This function is identical to the ES10b.GetEUICCInfo function defined in section 5.7.8 of SGP.22 [4], where the IPA plays the role of LPA.

Test Sequence #01 Nominal

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal* defined in 4.2.13.2.1 TC_eUICC_ES10b.GetEUICCInfo1, where the S_LPAd plays the role of S_IPA.

Test Sequence #02 Nominal: GetEUICCInfo call after GetEUICCChallenge

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Nominal: GetEUICCInfo call after GetEUICCChallenge* defined in 4.2.13.2.1 TC_eUICC_ES10b.GetEUICCInfo1, where the S_LPAd plays the role of S_IPA.

Test Sequence #03 Nominal: GetEUICCInfo1 call after AuthenticateServer

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal* defined in 4.2.13.2.1 TC_eUICC_ES10b.GetEUICCInfo1, where the S_LPAd plays the role of S_IPA.

4.2.13.2.2 TC_eUICC_ES10b.GetEUICCInfo2_RSP_Integrated_eUICC

NOTE: This test case is FFS.

4.2.14 ES10b (IPA -- eUICC): ListNotification

4.2.14.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

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 [5], SGP.22 [4], SGP.31 [2] or SGP.32 [3] for the number of Notifications that can be stored by the eUICC.

4.2.14.2.1 TC_eUICC_ES10b.ListNotification

4.2.15 ES10b (IPA -- eUICC): RetrieveNotificationsList

4.2.15.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.2.15.2 Test Cases

4.2.15.2.1 TC_eUICC_ES10b.RetrieveNotificationsList

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal Retrieve by Sequence Number for Install Notification Step Direction Sequence / Description Expected result REQ* defined in 4.2.15.2.1 TC_eUICC_ES10b.RetrieveNotificationsList, where the S_LPAd plays the role of S_IPA.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Retrieve by Sequence Number for Enable Notification* defined in 4.2.15.2.1 TC_eUICC_ES10b.RetrieveNotificationsList, where the S_LPAd plays the role of S_IPA.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #03 Nominal: Retrieve by Sequence Number for Disable Notification* defined in 4.2.15.2.1 TC_eUICC_ES10b.RetrieveNotificationsList, where the S_LPAd plays the role of S_IPA.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #04 Nominal: Retrieve by Sequence Number for Delete Notification* defined in 4.2.15.2.1 TC_eUICC_ES10b.RetrieveNotificationsList, where the S_LPAd plays the role of S_IPA.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #05 Nominal: Retrieve by Sequence Number for Two Install (PIR) Notifications with different Notification Addresses* defined in 4.2.15.2.1 TC_eUICC_ES10b.RetrieveNotificationsList, where the S_LPAd plays the role of S_IPA.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #06 Nominal: Retrieve by Sequence Number for Install (PIR) and Enable Notifications* defined in 4.2.15.2.1 TC_eUICC_ES10b.RetrieveNotificationsList, where the S_LPAd plays the role of S_IPA.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #07 Nominal: Retrieve by Sequence Number for Disable and Delete Notifications* defined in 4.2.15.2.1 TC_eUICC_ES10b.RetrieveNotificationsList, where the S_LPAd plays the role of S_IPA.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #08 Nominal: Retrieve by Sequence Number for Install (OtherSignedNotification) and Enable Notifications* defined in 4.2.15.2.1 TC_eUICC_ES10b.RetrieveNotificationsList, where the S_LPAd plays the role of S_IPA.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #09 Nominal: Retrieve by Sequence Number for Enable and Install (PIR) notifications* defined in 4.2.15.2.1 TC_eUICC_ES10b.RetrieveNotificationsList, where the S_LPAd plays the role of S_IPA.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #10 Nominal: Retrieve Sequence Numbers that are not present* defined in 4.2.15.2.1 TC_eUICC_ES10b.RetrieveNotificationsList, where the S_LPAd plays the role of S_IPA.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #11 Nominal: Retrieve by Notification Type for Install Notifications* defined in 4.2.15.2.1 TC_eUICC_ES10b.RetrieveNotificationsList, where the S_LPAd plays the role of S_IPA.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #12 Nominal: Retrieve by Notification Type for Enable Notification* defined in 4.2.15.2.1 TC_eUICC_ES10b.RetrieveNotificationsList, where the S_LPAd plays the role of S_IPA.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #13 Nominal: Retrieve by Notification Type for Disable Notification* defined in 4.2.15.2.1 TC_eUICC_ES10b.RetrieveNotificationsList, where the S_LPAd plays the role of S_IPA.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #14 Nominal: Retrieve by Notification Type for Delete Notification* defined in 4.2.15.2.1 TC_eUICC_ES10b.RetrieveNotificationsList, where the S_LPAd plays the role of S_IPA

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #15 Nominal: Retrieve by Notification Type for Two Install (PIR) Notifications with different Notification Addresses* defined in 4.2.15.2.1 TC_eUICC_ES10b.RetrieveNotificationsList, where the S_LPAd plays the role of S_IPA

Test Sequence #16 Nominal: Retrieve by Notification Type for Install (PIR) and Enable Notifications

This test sequence is the same as SGP.23 [8] - the *Test Sequence #16 Nominal: Retrieve by Notification Type for Install (PIR) and Enable Notifications* defined in 4.2.15.2.1 TC_eUICC_ES10b.RetrieveNotificationsList, where the S_LPAd plays the role of S_IPA

Test Sequence #17 Nominal: Retrieve by Notification Type for Disable and Delete Notifications

This test sequence is the same as SGP.23 [8] - the *Test Sequence #17 Nominal: Retrieve by Notification Type for Disable and Delete Notifications* defined in 4.2.15.2.1 TC_eUICC_ES10b.RetrieveNotificationsList, where the S_LPAd plays the role of S_IPA

Test Sequence #18 Nominal: Retrieve by Notification Type for Install (OtherSignedNotification) and Enable Notifications

This test sequence is the same as SGP.23 [8] - the *Test Sequence #18 Nominal: Retrieve by Notification Type for Install (OtherSignedNotification) and Enable Notifications* defined in 4.2.15.2.1 TC_eUICC_ES10b.RetrieveNotificationsList, where the S_LPAd plays the role of S_IPA

Test Sequence #19 Nominal: Retrieve by Notification Type for Enable and Install (PIR) notifications

This test sequence is the same as SGP.23 [8] - the *Test Sequence #19 Nominal: Retrieve by Notification Type for Enable and Install (PIR) notifications* defined in 4.2.15.2.1 TC_eUICC_ES10b.RetrieveNotificationsList, where the S_LPAd plays the role of S_IPA

Test Sequence #20 Nominal: Retrieve by Notification Type for No Notifications available

This test sequence is the same as SGP.23 [8] - the *Test Sequence #20 Nominal: Retrieve by Notification Type for No Notifications available* defined in 4.2.15.2.1 TC_eUICC_ES10b.RetrieveNotificationsList, where the S_LPAd plays the role of S_IPA

4.2.16 ES10b (IPA -- eUICC): RemoveNotificationFromList

4.2.16.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

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 [5], SGP.22 [4], SGP.31 [2] or SGP.32 [3] 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

This function is identical to the ES10b.RemoveNotificationFromList function defined in section 5.7.11 of SGP.22 [4], where the IPA plays the role of LPA.

Test Sequence #01 Nominal: Install Notification

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal: Install Notification* defined in 4.2.16.2.1 TC_eUICC_ES10b.RemoveNotificationFromListwhere the S_LPAd plays the role of S_IPA.

Test Sequence #02 Nominal: Enable Notification

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Nominal: Enable Notification* defined in 4.2.16.2.1 TC_eUICC_ES10b.RemoveNotificationFromListwhere the S_LPAd plays the role of S_IPA.

Test Sequence #03 Nominal: Disable Notification

This test sequence is the same as SGP.23 [8] - the *Test Sequence #03 Nominal: Disable Notification* defined in 4.2.16.2.1 TC_eUICC_ES10b.RemoveNotificationFromListwhere the S_LPAd plays the role of S_IPA.

Test Sequence #04 Nominal: Delete Notification

This test sequence is the same as SGP.23 [8] - the *Test Sequence #04 Nominal: Delete Notification* defined in 4.2.16.2.1 TC_eUICC_ES10b.RemoveNotificationFromListwhere the S_LPAd plays the role of S_IPA.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #05 Nominal: Two Install (PIR) Notifications with different Notification Addresses* defined in 4.2.16.2.1 TC_eUICC_ES10b.RemoveNotificationFromListwhere the S_LPAd plays the role of S_IPA.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #06 Nominal: Install (PIR) and Enable Notifications* defined in 4.2.16.2.1 TC_eUICC_ES10b.RemoveNotificationFromListwhere the S_LPAd plays the role of S_IPA.

Test Sequence #07 Nominal: Disable and Delete Notifications

This test sequence is the same as SGP.23 [8] - the *Test Sequence #07 Nominal: Disable and Delete Notifications* defined in 4.2.16.2.1 TC_eUICC_ES10b.RemoveNotificationFromListwhere the S_LPAd plays the role of S_IPA.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #08 Nominal: Install (OtherSignedNotification) and Enable Notifications* defined in 4.2.16.2.1 TC_eUICC_ES10b.RemoveNotificationFromListwhere the S_LPAd plays the role of S_IPA.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #09 Nominal: Enable and Install (PIR) notifications* defined in 4.2.16.2.1 TC_eUICC_ES10b.RemoveNotificationFromListwhere the S_LPAd plays the role of S_IPA.

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

This test sequence is the same as SGP.23 [8] - the *Test Sequence #10 Nominal: Removing Sequence Numbers that are not present* defined in 4.2.16.2.1 TC_eUICC_ES10b.RemoveNotificationFromListwhere the S_LPAd plays the role of S_IPA.

4.2.16.2.2 VOID

4.2.17 VOID

4.2.18 ES10b (IPA -- eUICC): AuthenticateServer

4.2.18.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.2.18.2 Test Cases

4.2.18.2.1 TC_eUICC_ES10b.AuthenticateServer_SM-DP+_NIST

Test Sequence #01 Nominal: Without MatchingID in CtxParams1

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal: Without MatchingID in CtxParams1* defined in 4.2.18.2.1 TC_eUICC_ES10b.AuthenticateServer_SM-DP+_NIST where the S_LPAd plays the role of S_IPA.

Test Sequence #02 Nominal: With MatchingID in CtxParams1

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Nominal: With MatchingID in CtxParams1* defined in 4.2.18.2.1 TC_eUICC_ES10b.AuthenticateServer_SM-DP+_NIST where the S_LPAd plays the role of S_IPA.

Test Sequence #03 Nominal: With IMEI in Device Capabilities

This test sequence is the same as SGP.23 [8] - the *Test Sequence #03 Nominal: With IMEI in Device Capabilities* defined in 4.2.18.2.1 TC_eUICC_ES10b.AuthenticateServer_SM-DP+_NIST where the S_LPAd plays the role of S_IPA.

4.2.18.2.2 TC_eUICC_ES10b.AuthenticateServer_SM-DP+_BRP

Test Sequence #01 Nominal: Without MatchingID in CtxParams1

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Nominal: Without MatchingID in CtxParams1* defined in 4.2.18.2.2 TC_eUICC_ES10b.AuthenticateServer_SM-DP+_BRP where the S_LPAd plays the role of S_IPA.

Test Sequence #02 Nominal: With MatchingID in CtxParams1

This test sequence is the same as SGP.23 [8] - the Test Sequence #02 Nominal: With MatchingID in CtxParams1 defined in 4.2.18.2.2 TC_eUICC_ES10b.AuthenticateServer_SM-DP+_BRP where the S_LPAd plays the role of S_IPA.

Test Sequence #03 Nominal: With IMEI in Device Capabilities

This test sequence is the same as SGP.23 [8] - the *Test Sequence #03 Nominal: With IMEI in Device Capabilities* defined in 4.2.18.2.2 TC_eUICC_ES10b.AuthenticateServer_SM-DP+_BRP where the S_LPAd plays the role of S_IPA.

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

This function is identical to the ES10b.AuthenticateServer function defined in section 5.7.13 of SGP.22 [4], where the IPA plays the role of LPA.

Test Sequence #01 Error: With Incorrect SM-DPauth certificate (i.e. invalid signature)

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Error: With Incorrect SM-DPauth certificate (i.e. invalid signature)* defined in 4.2.18.2.4 TC_eUICC_ES10b.AuthenticateServer_SM-DP+_ErrorCases where the S_LPAd plays the role of S_IPA.

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

This test sequence is the same as SGP.23 [8] - the Test Sequence #02 Error: With Invalid SM-DP+ Signature defined in 4.2.18.2.4 TC_eUICC_ES10b.AuthenticateServer_SM-DP+_ErrorCases where the S_LPAd plays the role of S_IPA.

Test Sequence #03 Error: Unsupported Curve

This test sequence is the same as SGP.23 [8] - the *Test Sequence #03 Error: Unsupported Curve* defined in 4.2.18.2.4 TC_eUICC_ES10b.AuthenticateServer_SM-DP+_ErrorCases where the S_LPAd plays the role of S_IPA.

Test Sequence #04 Error: eUICC Challenge Mismatch

This test sequence is the same as SGP.23 [8] - the *Test Sequence #04 Error: eUICC Challenge Mismatch* defined in 4.2.18.2.4 TC_eUICC_ES10b.AuthenticateServer_SM-DP+_ErrorCases where the S_LPAd plays the role of S_IPA.

Test Sequence #05 Error: Unknown CI PK

This test sequence is the same as SGP.23 [8] - the *Test Sequence #05 Error: Unknown CI PK* defined in 4.2.18.2.4 TC_eUICC_ES10b.AuthenticateServer_SM-DP+_ErrorCases where the S_LPAd plays the role of S_IPA.

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.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #06 Error: Invalid Certificate Role OID* defined in 4.2.18.2.4 TC_eUICC_ES10b.AuthenticateServer_SM-DP+_ErrorCases where the S_LPAd plays the role of S_IPA.

Test Sequence #07 Error: No RSP session on-going

This test sequence is the same as SGP.23 [8] - the *Test Sequence #07 Error: No RSP session on-going* defined in 4.2.18.2.4 TC_eUICC_ES10b.AuthenticateServer_SM-DP+_ErrorCases where the S_LPAd plays the role of S_IPA.

4.2.18.2.5 TC_eUICC_ES10b.AuthenticateServer_SM-DS_BRP

This function is identical to the ES10b.AuthenticateServer function defined in section 5.7.13 of SGP.22 [4], where the IPA plays the role of LPA.

Test Sequence #01 Nominal: With EventID in CtxParams1

This test sequence is the same as SGP.23 [8] - *the Test Sequence #01 Nominal: With EventID in CtxParams1* defined in 4.2.18.2.5 TC_eUICC_ES10b.AuthenticateServer_SM-DS_BRP where the S_LPAd plays the role of S_IPA.

Test Sequence #02 Nominal: With IMEI and EventID in Device Capabilities

This test sequence is the same as SGP.23 [8] - *Test Sequence #02 Nominal: With IMEI and EventID in Device Capabilities* defined in 4.2.18.2.5 TC_eUICC_ES10b.AuthenticateServer_SM-DS_BRP where the S_LPAd plays the role of S_IPA.

4.2.18.2.6 TC_eUICC_ES10b.AuthenticateServer_SM-DS_NIST

This function is identical to the ES10b.AuthenticateServer function defined in section 5.7.13 of SGP.22 [4], where the IPA plays the role of LPA.

Test Sequence #01 Nominal: With EventID in CtxParams1

This test sequence is the same as SGP.23 [8] – *the Test Sequence #01 Nominal: With EventID in CtxParams1* defined in 4.2.18.2.6 TC_eUICC_ES10b.AuthenticateServer_SM-DS_NIST where the S_LPAd plays the role of S_IPA.

Test Sequence #02 Nominal: With IMEI and EventID in Device Capabilities

This test sequence is the same as SGP.23 [8] – the *Test Sequence #02 Nominal: With IMEI and EventID in Device Capabilities* defined in 4.2.18.2.6 TC_eUICC_ES10b.AuthenticateServer_SM-DS_NIST where the S_LPAd plays the role of S_IPA.

4.2.18.2.7 VOID

4.2.18.2.8 TC_eUICC_ES10b.AuthenticateServer_SM-DS_ErrorCases

This function is identical to the ES10b.AuthenticateServer function defined in section 5.7.13 of SGP.22 [4], where the IPA plays the role of LPA

Test Sequence #01 Error: With Incorrect SM-DSauth certificate (i.e. invalid signature)

This test sequence is the same as SGP.23 [8] – the *Test Sequence #01 Error: With Incorrect SM-DSauth certificate (i.e. invalid signature)* defined in 4.2.18.2.8 TC_eUICC_ES10b.AuthenticateServer_SM-DS_ErrorCases where the S_LPAd plays the role of S_IPA.

Test Sequence #02 Error: With Invalid SM-DS Signature

This test sequence is the same as SGP.23 [8] – the *Test Sequence #02 Error: With Invalid SM-DS Signature* defined in 4.2.18.2.8 TC_eUICC_ES10b.AuthenticateServer_SM-DS_ErrorCases where the S_LPAd plays the role of S_IPA.

Test Sequence #03 Error: Unsupported Curve

This test sequence is the same as SGP.23 [8] – the *Test Sequence #03 Error: Unsupported Curve* defined in 4.2.18.2.8 TC_eUICC_ES10b.AuthenticateServer_SM-DS_ErrorCases where the S_LPAd plays the role of S_IPA.

Test Sequence #04 Error: eUICC Challenge Mismatch

This test sequence is the same as SGP.23 [8] – the *Test Sequence #04 Error: eUICC Challenge Mismatch* defined in 4.2.18.2.8 TC_eUICC_ES10b.AuthenticateServer_SM-DS_ErrorCases where the S_LPAd plays the role of S_IPA.

Test Sequence #05 Error: Unknown CI PK

This test sequence is the same as SGP.23 [8] – the *Test Sequence #05 Error: Unknown CI PK* defined in 4.2.18.2.8 TC_eUICC_ES10b.AuthenticateServer_SM-DS_ErrorCases where the S_LPAd plays the role of S_IPA.

Test Sequence #06 Error: No RSP session on-going

This test sequence is the same as SGP.23 [8] – the *Test Sequence #06 Error: No RSP session on-going* defined in 4.2.18.2.8 TC_eUICC_ES10b.AuthenticateServer_SM-DS_ErrorCases where the S_LPAd plays the role of S_IPA.

4.2.19 ES10b (LPA -- eUICC): CancelSession

4.2.19.1 Conformance Requirements

This function is identical to the ES10b.CancelSession function defined in section 5.7.14 of SGP.22 [4], where the IPA plays the role of LPA, with the following additional eUICC behaviour:

- The eUICC SHALL revoke any grant to automatically enable a Profile.

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.2.19.2 Test Cases

4.2.19.2.1 TC_eUICC_ES10b.CancelSessionNIST

This function is identical to the ES10b.CancelSession function defined in section 5.7.14 of SGP.22 [4], where the IPA plays the role of LPA.

Test Sequence #01 Nominal: VOID

Test Sequence #02 Nominal: VOID

Test Sequence #03 Nominal: VOID

Test Sequence #04 Nominal: PPR not allowed

The RSP session is terminated because the IPAd detected that PPR(s) set in the Profile Metadata is/are not allowed.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #04 Nominal: PPR not allowed* defined in 4.2.19.2.1 TC_eUICC_ES10b.CancelSessionNIST where the S_LPAd plays the role of S_IPA.

Test Sequence #05 Nominal: Metadata Mismatch

The RSP session is terminated because the IPAd detected that the Profile Metadata in the response to "ES9+.AuthenticateClient" does not match the Profile Metadata in the Bound Profile Package.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #05 Nominal: Metadata Mismatch* defined in 4.2.19.2.1 TC_eUICC_ES10b.CancelSessionNIST where the S_LPAd plays the role of S_IPA.

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+.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #05 Nominal: Metadata Mismatch* defined in 4.2.19.2.1 TC_eUICC_ES10b.CancelSessionNIST where the S_LPAd plays the role of S_IPA.

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+.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #07 Nominal: Load BPP Execution Error* defined in 4.2.19.2.1 TC_eUICC_ES10b.CancelSessionNIST where the S_LPAd plays the role of S_IPA.

Test Sequence #08 Nominal: Undefined Reason

This test sequence is the same as SGP.23 [8] - the *Test Sequence #08 Nominal: Undefined Reason* defined in 4.2.19.2.1 TC_eUICC_ES10b.CancelSessionNIST where the S_LPAd plays the role of S_IPA.

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.

Test Sequence #01 Nominal: VOID

Test Sequence #02 Nominal: VOID

Test Sequence #03 Nominal: VOID

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.

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

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

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

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 TC_eUICC_ES10b.CancelSessionFRP

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

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.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #01 Error: No on-going RSP session* defined in 4.2.19.2.4 TC_eUICC_ES10b.CancelSession_ErrorCasewhere the S_LPAd plays the role of S_IPA.

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.

This test sequence is the same as SGP.23 [8] - the *Test Sequence #02 Error: Invalid Transaction ID* defined in 4.2.19.2.4 TC_eUICC_ES10b.CancelSession_ErrorCasewhere the S_LPAd plays the role of S_IPA.

4.2.19.2.5 VOID

4.2.19.2.6 VOID

4.2.20 ES10b (LPA -- eUICC): eUICCMemoryReset

4.2.20.1 Conformance Requirements

This function is the same from the ES10c.eUICCMemoryReset function defined in SGP.22 [4] with the exception of new Nominal and error cases that are for further study.

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.2.20.2 Test Cases

4.2.20.2.1 TC_eUICC_ES10b.eUICCMemoryReset

Test Sequence #01 Nominal: Reset All Operational Profiles

This test sequence is the same as SGP.23 [8] – the *Test Sequence #01 Nominal: Reset All Operational Profiles (without Enabled Profile)* defined in 4.2.24.2.1 TC_eUICC_ES10c.eUICCMemoryReset where the S_LPAd plays the role of S_IPA.

Test Sequence #02 Nominal: Reset Field Loaded Test Profiles

This test sequence is the same as SGP.23 [8] – the *Test Sequence #02 Nominal: Reset Field Loaded Test Profile)* defined in 4.2.24.2.1 TC_eUICC_ES10c.eUICCMemoryReset where the S_LPAd plays the role of S_IPA.

Test Sequence #03 Nominal: Reset the Default SM-DP+ Address only

This test sequence is the same as SGP.23 [8] – the *Test Sequence #03 Nominal: Reset the Default SM-DP+ Address only* defined in 4.2.24.2.1 TC_eUICC_ES10c.eUICCMemoryReset where the S_LPAd plays the role of S_IPA.

Test Sequence #04 Nominal: EIM Reset

NOTE: This test sequence is FFS.

Test Sequence #05 Nominal: Auto Enable Reset

NOTE: This test sequence is FFS.

4.2.24.2.2 TC_eUICC_ES10b.eUICCMemoryReset_ErrorCases

Test Sequence #01 Error: eUICC Memory Reset while there is no Profile to delete

This test sequence is the same as SGP.23 [8] – the *Test Sequence #01 Error: eUICC Memory Reset while there is no Profile to delete* defined in 4.2.24.2.2 TC_eUICC_ES10c.eUICCMemoryReset_ErrorCases where the S_LPAd plays the role of S_IPA.

Test Sequence #02 Error: eUICC Memory Reset – Cat Busy

This test sequence is the same as SGP.23 [8] – the *Test Sequence #02 Error: eUICC Memory Reset – Cat Busy* defined in 4.2.24.2.2 TC_eUICC_ES10c.eUICCMemoryReset_ErrorCases where the S_LPAd plays the role of S_IPA.

Test Sequence #03 Error: EIM Reset – Nothing to delete

NOTE: This test sequence is FFS.

Test Sequence #04 Error: EIM Reset – EIM Reset not supported

NOTE: This test sequence is FFS.

Test Sequence #05 Error: Auto Enable Reset – Auto Enable Reset not supported

NOTE: This test sequence is FFS.

4.2.21 ES10b (LPA -- eUICC): GetEID

This function is identical to the ES10c.GetEID function defined in section 5.7.20 of SGP.22 [4], where the IPA plays the role of LPA.

4.2.21.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.2.21.2 Test Cases

4.2.21.2.1 TC_eUICC_ES10b.GetEID

Test Sequence #01 Nominal

The purpose of this test is to ensure that it is possible to retrieve the EID. This test sequence is the same as the *Test Sequence #01 Nominal* defined in SGP.23 [8] - 4.2.25.2.1 TC_eUICC_ES10c.GetEID, where the S_LPAd plays the role of S_IPA.

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.

This test sequence is the same as the *Test Sequence #02 Error* defined in SGP.23 [8] - 4.2.25.2.1 TC_eUICC_ES10c.GetEID, where the S_LPAd plays the role of S_IPA.

.4.2.22 ES10b (LPA -- eUICC): GetRAT

4.2.22.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.2.22.2 Test Cases

4.2.22.2.1 TC_eUICC_ES10b.GetRAT

This eUICC_ES10b.GetRAT function is identical to the ES10b.GetRAT function defined in in SGP.23 [8] - section 5.7.22 of SGP.22 [4], where the IPA plays the role of LPA.

Test Sequence #01 Nominal: Get Default RAT

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: Only Root SM-DS Address* defined in 4.2.27.2.1 TC_eUICC_ES10b.GetRAT, where the S_LPAd plays the role of S_IPA.

Test Sequence #02 Nominal: With additional PPARs

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Nominal: With additional PPARs* defined in 4.2.27.2.1 TC_eUICC_ES10b.GetRAT, where the S_LPAd plays the role of S_IPA.

4.2.23 ES10b (LPA -- eUICC): GetProfilesInfo

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

This function is identical to the ES10c.GetProfilesInfo function defined in section 5.7.15 of SGP.22 [4], where the IPA plays the role of LPA.

4.2.23.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.2.23.2 Test Cases

4.2.23.2.1 TC_eUICC_ES10b.GetProfilesInfo

Test Sequence #01 Nominal: Get All Profiles

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: Get All Profiles* - 4.2.20.2.1 TC_eUICC_ES10c.GetProfilesInfo, where the S_LPAd plays the role of S_IPA.

Test Sequence #02 Nominal: Get Profile by ICCID

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Nominal: Get Profile by ICCID* - 4.2.20.2.1 TC_eUICC_ES10c.GetProfilesInfo, where the S_LPAd plays the role of S_IPA.

Test Sequence #03 Nominal: Get Profile by AID

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #03 Nominal: Get Profile by AID - 4.2.20.2.1 TC_eUICC_ES10c.GetProfilesInfo, where the S_LPAd plays the role of S_IPA.

Test Sequence #04 Nominal: Get All Operational Profiles

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #04 Nominal: Get All Operational Profiles Test Sequence #02 Nominal: Get Profile by ICCID 4.2.20.2.1 TC_eUICC_ES10c.GetProfilesInfo, where the S_LPAd plays the role of S_IPA.

Test Sequence #05 Nominal: Get Profile ICCID list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #05 Nominal: Get Profile ICCID list defined 4.2.20.2.1 TC_eUICC_ES10c.GetProfilesInfo, where the S_LPAd plays the role of S_IPA.

Test Sequence #06 Nominal: Get Profile AID list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #06 Nominal: Get Profile AID list - 4.2.20.2.1 TC_eUICC_ES10c.GetProfilesInfo, where the S_LPAd plays the role of S_IPA.

Test Sequence #07 Nominal: Get Profile Nickname list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #07 Nominal: Get Profile Nickname list - 4.2.20.2.1 TC_eUICC_ES10c.GetProfilesInfo, where the S_LPAd plays the role of S_IPA.

Test Sequence #08 Nominal: Get Profile SP Name list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #08 Nominal: Get Profile SP Name list - 4.2.20.2.1 TC_eUICC_ES10c.GetProfilesInfo, where the S_LPAd plays the role of S_IPA.

Test Sequence #09 Nominal: Get Profile Name list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #09 Nominal: Get Profile Name list defined - 4.2.20.2.1 TC_eUICC_ES10c.GetProfilesInfo, where the S_LPAd plays the role of S_IPA.

Test Sequence #10 Nominal: Get Profile Icon list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #10 Nominal: Get Profile Icon list defined - 4.2.20.2.1 TC_eUICC_ES10c.GetProfilesInfo, where the S_LPAd plays the role of S_IPA.

Test Sequence #11 Nominal: Get Profile Owner list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #11 Nominal: Get Profile Owner list by ICCID - 4.2.20.2.1 TC_eUICC_ES10c.GetProfilesInfo, where the S_LPAd plays the role of S_IPA.

Test Sequence #12 Nominal: Get Profile SM-DP+ proprietary data list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #12 Nominal: Get Profile SM-DP+ proprietary data list - 4.2.20.2.1 TC_eUICC_ES10c.GetProfilesInfo, where the S_LPAd plays the role of S_IPA.

Test Sequence #13 Nominal: Get Profile ICCID and State list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #13 Nominal: Get Profile ICCID and State list - 4.2.20.2.1 TC_eUICC_ES10c.GetProfilesInfo, where the S_LPAd plays the role of S_IPA.

Test Sequence #14 Nominal: Get Profile Nickname and State list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #14 Nominal: Get Profile Nickname and State list - 4.2.20.2.1 TC_eUICC_ES10c.GetProfilesInfo, where the S_LPAd plays the role of S_IPA.

Test Sequence #15 Nominal: Get Profile Icon and Icon Type list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #15 Nominal: Get Profile Icon and Icon Type list - 4.2.20.2.1 TC_eUICC_ES10c.GetProfilesInfo, where the S_LPAd plays the role of S_IPA.

Test Sequence #16 Nominal: Get Profile Icon and State list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #16 Nominal: Get Profile Icon and State list - 4.2.20.2.1 TC_eUICC_ES10c.GetProfilesInfo, where the S_LPAd plays the role of S_IPA.

Test Sequence #17 Nominal: Get Operational Profile ICCID and State list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #17 Nominal: Get Operational Profile ICCID and State list - 4.2.20.2.1 TC_eUICC_ES10c.GetProfilesInfo, where the S_LPAd plays the role of S_IPA.

Test Sequence #18 Nominal: Get Operational Profile Nickname and State list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #18 Nominal: Get Operational Profile Nickname and State list - 4.2.20.2.1 TC_eUICC_ES10c.GetProfilesInfo, where the S_LPAd plays the role of S_IPA.

Test Sequence #19 Nominal: Get Operational Profile Icon and Icon type list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #19 Nominal: Get Operational Profile Icon and Icon type list - 4.2.20.2.1 TC_eUICC_ES10c.GetProfilesInfo, where the S_LPAd plays the role of S_IPA.

Test Sequence #20 Nominal: Get Operational Profile Icon and State list

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #20 Nominal: Get Operational Profile Icon and State list - 4.2.20.2.1 TC_eUICC_ES10c.GetProfilesInfo, where the S_LPAd plays the role of S_IPA.

Test Sequence #21 Nominal: Get Profile State of the defined Profile

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #21 Nominal: Get Profile State of the defined Profile - 4.2.20.2.1 TC_eUICC_ES10c.GetProfilesInfo, where the S_LPAd plays the role of S_IPA.

Test Sequence #22 Nominal: Get Profile Icon Type of the defined Profile

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #22 Nominal: Get Profile Icon Type of the defined Profile - 4.2.20.2.1 TC_eUICC_ES10c.GetProfilesInfo, where the S_LPAd plays the role of S_IPA.

Test Sequence #23 Nominal: Get Profile Class of the defined Profile

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #23 Nominal: Get Profile Class of the defined Profile - 4.2.20.2.1 TC_eUICC_ES10c.GetProfilesInfo, where the S_LPAd plays the role of S_IPA.

Test Sequence #24 Nominal: Get Notification Configuration of the defined Profile

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #24 Nominal: Get Notification Configuration of the defined Profile - 4.2.20.2.1 TC_eUICC_ES10c.GetProfilesInfo, where the S_LPAd plays the role of S_IPA.

Test Sequence #25 Nominal: Get Profile Policy Rules of the defined Profile

This test sequence is the same as defined in SGP.23 [8] - the Test Sequence #25 Nominal: Get Profile Policy Rules of the defined Profile - 4.2.20.2.1 TC_eUICC_ES10c.GetProfilesInfo, where the S_LPAd plays the role of S_IPA.

Test Sequence #26 Nominal: Get empty Profile list. No Profile installed

This test sequence is the same as defined in SGP.23 [8] - ***the*** Test Sequence #26 Nominal: Get empty Profile list. No Profile installed ***- 4.2.20.2.1 TC_eUICC_ES10c.GetProfilesInfo,*** where the S_LPAd plays the role of S_IPA.

4.2.24 ES10b(IPA – eUICC) LoadEUICCPackage

This function executes a eUICC Package.

4.2.24.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.2.24.2 Test Cases

4.2.24.2.1 TC_eUICC_ES10b. LoadEUICCPackage

The Nominal Test Cases for LoadeUICCPackage function are covered by the defined test cases for ESep interface (Section from 4.2.31 to 4.2.39).**4.2.24.2.2TC_eUICC_ES10b. LoadEUICCPackage_ErrorCases**

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	#EIM_CONFIG_DATA1 is configured in the eUICC.

Test Sequence #01 Error: eimID is unknown

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_EUICC_PACKAGE_REQUEST_UPDATE_EIM(TEST_EIM_ID2, EID1, S_COUNTER_VALUE, NO_PARAM, EIM_CONFIG_DATA1_UPDATE_COUNTER, S_EIM_SIGNATURE))	SW=0x9000 with response data: MTD_EUICC_PKG_ERROR_UNSIGNED (#TEST_EIM_ID2)

Test Sequence #02 Error: Signature Invalid

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_EUICC_PACKAGE_REQUEST_UPDATE_EIM(TEST_EIM_ID1, EID1, S_COUNTER_VALUE, NO_PARAM, EIM_CONFIG_DATA1_UPDATE_COUNTER, S_EIM_SIGN_RANDOM))	SW=0x9000 with response data: MTD_EUICC_PKG_ERROR_UNSIGNED (#TEST_EIM_ID1)

Test Sequence #03 Error: EID does not match/invalid

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_EUICC_PACKAGE_REQUEST_UPDATE_EIM(TEST_EIM_ID1, EID2, S_COUNTER_VALUE, NO_PARAM, EIM_CONFIG_DATA1_UPDATE_COUNTER, S_EIM_SIGN_RANDOM))	SW=0x9000 with response data: MTD_EUICC_PKG_ ERROR_SIGNED (#LOAD_EUICC_PKG_ ERR_EID)

Test Sequence #04 Error: Counter value less than the counter value storted 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_EUICC_PACKAGE_REQUEST_UPDATE_EIM(TEST_EIM_ID1, EID1, S_LOW_CNTR_VALUE, NO_PARAM, EIM_CONFIG_DATA1_UPDATE_COUNTER, S_EIM_SIGNATURE))	SW=0x9000 with response data: MTD_EUICC_PKG_ ERROR_SIGNED (#LOAD_EUICC_PKG_ ERR_LOW_CNTR)

Test Sequence #05 Error: PSMO invalid due to the size limit of the response data

This test sequence of this error is not defined for this version of the specification.

Test Sequence #06 Error: eCO invalid or unsupported

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_EUICC_PKG_REQUEST(	SW=0x9000 with response data: MTD_EUICC_PKG_RESULT(

		#TEST_EIM_ID1, NO_PARAM, <S_TRANSACTION_ID>, euiccPackage #LOAD_EUICC_PKG_INVALID_ECO)))	(#LOAD_EUICC_PKG_ ERR_ UNKNOWN_CMD)

Test Sequence #07 Error: PSMO invalid or unsupported

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_EUICC_PKG_REQUEST(#TEST_EIM_ID1, NO_PARAM, <S_TRANSACTION_ID>, euiccPackage #LOAD_EUICC_PKG_INVALID_PSMO)))	SW=0x9000 with response data: MTD_EUICC_PKG_RESULT((#LOAD_EUICC_PKG_ ERR_ UNKNOWN_CMD)

4.2.25 ES10b (IPA -- eUICC): AddInitialEim

This function is used by the IPAd to store eIM Configuration Data to the eUICC in case the eUICC does not contain any eIM Configuration Data.

4.2.25.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.2.24.2 Test Cases

4.2.25.2.1 TC_eUICC_ES10b.AddInitialEim

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The eUICC does not contain any eIM Configuration Data.

Test Sequence #01 Nominal 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(
		MTD_ADD_INITIAL_EIM_SINGLE(
		#EIM_CONFIG_DATA1
		)
		)
			response AddInitialEimResponse ::= {
			#ADD_INITIAL_EIM_RES_OK}
			SW=0x9000

4.2.25.2.1 TC_eUICC_ES10b.AddInitialEim_ErrorCases

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The eUICC does not contain any eIM Configuration Data.

Test Sequence #01 Error: Unsigned Eim Config Disallowed

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_ADD_INITIAL_EIM_SINGLE(
		#EIM_CONFIG_DATA1
		)
		)
			response AddInitialEimResponse ::= {
			#ADD_INITIAL_EIM_RES_OK
			}
			SW=0x9000

2	S_LPAd → eUICC	MTD_STORE_DATA(
		MTD_ADD_INITIAL_EIM_SINGLE(
		#EIM_CONFIG_DATA2
		)
		)
			response AddInitialEimResponse ::= {
			#ADD_INITIAL_EIM_ ERR_EIM_CONF
			}
			SW=0x9000

Test Sequence #02 Error: Insufficient Memory

Note: This error cannot be tested because there is no interoperable way at the time of writing to set a specified amount of available memory in an eUICC

Test Sequence #03 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(
		MTD_ADD_INITIAL_EIM_SINGLE(
		#EIM_CONFIG_DATA2_ADD_UKNOWN_EUICC_CI_PKID
		)
		)
			response AddInitialEimResponse ::= {
			#ADD_INITIAL_EIM_ERR_UNKNOWN_CI
			}
			SW=0x9000

Test Sequence #04 Error: Counter Value out of Range

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_ADD_INITIAL_EIM_SINGLE(
		#EIM_CONFIG_DATA2_ADD_COUNTER_OUT_OF_RANGE
		)
		)
			response AddInitialEimResponse ::= {
			#ADD_INITIAL_EIM_ERR_CNTR_OUT_OF_RANGE
			}
			SW=0x9000

Test Sequence #05 Error: Invalid Association Token

The test sequence for this rrror case is not defined in this version of the specification.

4.2.26 ES10b (IPA -- eUICC): GetCerts

This function is used by the IPA to retrieve the eUICC Certificate and the EUM Certificate from the eUICC. This function can be used at any time by the IPA.

4.2.26.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.2.26.2 Test Cases

4.2.26.2.1 TC_eUICC_ES10b.GetCertificates

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	S_eIM with #EIM_CONFIG_DATA1 is configured in the eUICC.
eUICC	Set the <EUICC_CI_PK_ID_TO_BE_USED> to the CI Key ID based on NIST P-256 curve

Test Sequence #01 Nominal Case: euiccCiPKId set

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_CERTS_REQUEST (
		NO_PARAM
		)
		)
			SW=0x9000 with response data:
			GetCertsResponse ::= {
			#GET_CERTS_RESULT_OK
			}
			Verify that the EUM's Certificate #CERT_EUM_SIG contains an EUM's Public Key #PK_EUM_SIG based on NIST P-256 curve
			Verify that the eUICC's Certificate #CERT_EUICC_SIG contains an eUICC's Public Key #PK_EUICC_SIG based on NIST P-256 curve

Test Sequence #02 Nominal Case with euiccCiPKId set

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_CERTS_REQUEST (
		#CI_KEY_ID_BP256
		)
		)
			SW=0x9000 with response data:
			GetCertsResponse ::= {
			#GET_CERTS_RESULT_OK
			}

			Verify that the EUM's Certificate #CERT_EUM_SIG contains an EUM's Public Key #PK_EUM_SIG based on brainpoolP256r1 curve. Verify that the eUICC's Certificate #CERT_EUICC_SIG contains an eUICC's Public Key #PK_EUICC_SIG based on brainpoolP256r1 curve.

4.2.26.2.2 TC_eUICC_ES10b.GetCertificates_ErrorCases

Test Sequence #01 Error: Invalid CI PKid

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_CERTS_REQUEST (
		#CI_PKI_RANDOM
		)
		)
			SW=0x9000 with response data:
			GetCertsResponse ::= {
			#GET_CERTS_ERR_UNKNOWN_CI
			}

4.2.27 ES10b (IPA -- eUICC): EnableUsingDD

This function is used by IPA to request automatic Profile enabling without eIM involvement. The eUICC MAY be configured to support the automatic enabling.

4.2.27.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

- Section 5.9.15

4.2.27.2 Test Cases

4.2.27.2.1 TC_eUICC_ES10b.EnableUsingDD

Test Sequence #01 Nominal Case

Test the nominal case when the Automatic Profile Enabling is activated

Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	Automatic Profile Enabling is activated
eUICC	The configured OID is set to S_SM_DP+_OID1 and the SMDP_ADDRESS is set to SMDP_ADDRESS1

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_IPAd → eUICC	Install PROFILE_OPERATIONAL1 with OID value set to S_SM_DP+_OID1 and address value set to SMDP_ADDRESS1	Profile successfully downloaded (i.e. ProfileInstallationResult contains a SuccessResult)

2	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_USING_DD()
		)
			response EnableUsingDDResponse ::= {
			#ENABLE_USING_DD_RESULT_OK
			}
			SW=0x91XX

3	S_Device → eUICC	FETCH 'XX'	REFRESH Command ("UICC Reset")
4	Repeat IC1 and IC2		

5	S_IPAd → 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

4.2.27.2.2 TC_eUICC_ES10b.EnableUsingDD_ErrorCases

Test Sequence #01 Error: No Session Context

Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	Automatic Profile Enabling is activated
eUICC	The configured OID is set to S_SM_DP+_OID1 and the SMDP_ADDRESS is set to SMDP_ADDRESS1

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_IPAd → eUICC	Install PROFILE_OPERATIONAL1 with OID value set to S_SM_DP+_OID1 and address value set to SMDP_ADDRESS1	Profile successfully downloaded (i.e. ProfileInstallationResult contains a SuccessResult)

2	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_GET_RAT ()
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			euiccPackageResultDataSigned {
			eimId paramEIMId,
			counterValue paramCounterValue,
			seqNumber SEQ_NUMBER,
			euiccResult {
			getRATResult : {
			#PPRS_ALLOWED
			}
			}
			},
			euiccSignEPR EUICC_SIGN_EPR
			}
			SW = 0x9000

3	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_USING_DD()
		)
			response EnableUsingDDResponse ::= {
			#ENABLE_USING_DD_RESULT_NO_SESSION_CONTEXT
			}
			SW=0x9000

Test Sequence #02 Error: Auto-Enable is Not Available

Test the nominal case when the Automatic Profile Enabling is not activated

Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	Automatic Profile Enabling is not activated
eUICC	The configured OID is set to S_SM_DP+_OID1 and the SMDP_ADDRESS is set to SMDP_ADDRESS1

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_IPAd → eUICC	Install PROFILE_OPERATIONAL1 with OID value set to S_SM_DP+_OID1 and address value set to SMDP_ADDRESS1	Profile successfully downloaded (i.e. ProfileInstallationResult contains a SuccessResult)

2	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_USING_DD()
		)
			response EnableUsingDDResponse ::= {
			#ENABLE_USING_DD_RESULT_AUTO_ENABLE_NOT_AVAILABLE
			}
			SW=0x9000

4.2.28 ES10b (IPA -- eUICC): ProfileRollback

This function is used by the IPA to request to roll back to the previously Enabled Profile, if any.

4.2.28.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

- Section 2.11.1.1.3
- Section 2.11.2.1
- Section 3.3.1
- Section 3.4.1
- Section 5.9.16

4.2.28.2 Test Cases

4.2.28.2.1 TC_eUICC_ES10b.ProfileRollback

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_OPERATIONAL1 is enabled with rollbackFlag set

Test Sequence #1 Nominal Case: Profile Rollback with refreshFlag set

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_IPAd → eUICC	MTD_STORE_DATA(
		
		MTD_EUICC_PACKAGE_REQUEST_ENABLE(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		ICCID_OP_PROF2,
		TRUE,
		S_EIM_SIGNATURE
		)
		)
			response
			EuiccPackageResult ::=
			euiccPackageResultSigned : {
			#ENABLE_RES_OK_1
			}
			
			SW=0x91XX

2	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“UICC Reset”)
3	Repeat IC1 and IC2		

4	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID2>)
		)
			response
			ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

5	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_PROFILE_ROLLBACK_REQUEST(
		TRUE)
		)
			response
			ProfileRollbackResponse::= {
			#PROFILE_ROLLBACK_RES_OK
			}
			SW=0x9000

6	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“UICC Reset”)
7	Repeat IC1 and IC2		

8	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>)
		)
			response
			ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

9	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

10	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #2 Nominal Case: Profile Rollback with refreshFlag not set

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_IPAd → eUICC	MTD_STORE_DATA(
		
		MTD_EUICC_PACKAGE_REQUEST_ENABLE(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		ICCID_OP_PROF2,
		TRUE,
		S_EIM_SIGNATURE
		)
		)
			response
			EuiccPackageResult ::=
			euiccPackageResultSigned : {
			#ENABLE_RES_OK_1
			}
			
			SW=0x91XX

2	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“UICC Reset”)
3	Repeat IC1 and IC2		

4	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID2>)
		)
			response
			ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

5	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_PROFILE_ROLLBACK_REQUEST(
		FALSE)
		)
			response
			ProfileRollbackResponse::= {
			#PROFILE_ROLLBACK_RES_OK
			}
			SW=0x9000

6	S_IPAd → 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 #3 Nominal Case: Profile Rollback with refreshFLag set while proactive session is ongoing – catBusy not supported

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_IPAd → eUICC	MTD_STORE_DATA(
		
		MTD_EUICC_PACKAGE_REQUEST_ENABLE(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		ICCID_OP_PROF2,
		TRUE,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#ENABLE_RES_OK_1
			}
			
			SW=0x91XX

2	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“UICC Reset”)
3	Repeat IC1 and IC2		

4	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID2>)
		)
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX

6	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_PROFILE_ROLLBACK_REQUEST(
		TRUE)
		)
			response ProfileRollbackResponse::= {
			#PROFILE_ROLLBACK_RES_OK
			}
			SW=0x91YY

7	S_Device → eUICC	FETCH 'YY'	REFRESH Command (“UICC Reset”)
8	Repeat IC1 and IC2		

9	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>)
		)
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

10	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

11	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #4 Nominal Case: Profile Rollback with refreshFLag not set while proactive session is ongoing – catBusy not supported

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_IPAd → eUICC	MTD_STORE_DATA(
		
		MTD_EUICC_PACKAGE_REQUEST_ENABLE(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		ICCID_OP_PROF2,
		TRUE,
		S_EIM_SIGNATURE
		)
		)
			response
			EuiccPackageResult ::=
			euiccPackageResultSigned : {
			#ENABLE_RES_OK_1
			}
			
			SW=0x91XX

2	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“UICC Reset”)
3	Repeat IC1 and IC2		

4	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID2>)
		)
			response
			ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX

6	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_PROFILE_ROLLBACK_REQUEST(
		FALSE)
		)
			response
			ProfileRollbackResponse::= {
			#PROFILE_ROLLBACK_RES_OK
			}
			SW=0x9000

7	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>)
		)
			response
			ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

8	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

9	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #5 Nominal Case: Profile Rollback with refreshFLag set while proactive session is ongoing with Terminal Response outstanding – catBusy not supported

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_REQUEST_ENABLE(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		ICCID_OP_PROF2,
		TRUE,
		S_EIM_SIGNATURE
		)
		)
			response
			EuiccPackageResult ::=
			euiccPackageResultSigned : {
			#ENABLE_RES_OK_1
			}
			SW=0x91XX

2	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“UICC Reset”)
3	Repeat IC1 and IC2		

4	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID2>)
		)
			response
			ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
6	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK

7	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_PROFILE_ROLLBACK_REQUEST(
		TRUE)
		)
			response
			ProfileRollbackResponse::= {
			#PROFILE_ROLLBACK_RES_OK
			}
			SW=0x9000

8	S_Device → eUICC	TERMINAL RESPONSE	SW=0x91YY
9	S_Device → eUICC	FETCH 'YY'	REFRESH Command (“UICC Reset”)
10	Repeat IC1 and IC2		

11	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>)
		)
			response
			ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

12	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

13	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #6 Nominal Case: Profile Rollback with refreshFLag not set while proactive session is ongoing with Terminal Response outstanding – catBusy not supported

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_REQUEST_ENABLE(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		ICCID_OP_PROF2,
		TRUE,
		S_EIM_SIGNATURE
		)
		)
			response
			EuiccPackageResult ::=
			euiccPackageResultSigned : {
			#ENABLE_RES_OK_1
			}
			SW=0x91XX

2	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“UICC Reset”)
3	Repeat IC1 and IC2		

4	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID2>)
		)
			response
			ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX
6	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK

7	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_PROFILE_ROLLBACK_REQUEST(
		FALSE)
		)
			response
			ProfileRollbackResponse::= {
			#PROFILE_ROLLBACK_RES_OK
			}
			SW=0x9000

8	S_Device → eUICC	TERMINAL RESPONSE	SW= any value except91XX

9	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>)
		)
			response
			ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

10	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

11	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

4.2.28.2.2 TC_eUICC_ES10b.ProfileRollback_ErrorCases

Test Sequence #01 Error: Roll Back is not allowed

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_REQUEST_ENABLE(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		ICCID_OP_PROF2,
		FALSE,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#ENABLE_RES_OK_1
			}
			SW=0x91XX

2	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“UICC Reset”)
3	Repeat IC1 and IC2		

4	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_PROFILE_ROLLBACK_REQUEST(
		FALSE)
		)
			response ProfileRollbackResponse::= {
			#PROFILE_ROLLBACK_RES_ROLLBACK_NOT_ALLOWED
			}
			SW=0x9000

5	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID2>)
		)
			response ProfileInfoListResponse::= profileInfoListOk : {
			#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 #02 Error: Command Error

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_PROFILE_ROLLBACK_REQUEST(
		FALSE)
		)
			response ProfileRollbackResponse::= {
			#PROFILE_ROLLBACK_RES_COMMAND_ERROR
			}
			SW=0x9000

2	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>)
		)
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO1_ENABLED
			}
			SW=0x9000

3	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

4	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

Test Sequence #3 Error: catBusy with refreshFlag set while proactive session is ongoing – catBusy supported

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_REQUEST_ENABLE(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		ICCID_OP_PROF2,
		TRUE,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#ENABLE_RES_OK_1
			}
			SW=0x91XX

2	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“UICC Reset”)
3	Repeat IC1 and IC2		

4	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID2>)
		)
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX

6	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_PROFILE_ROLLBACK_REQUEST(
		TRUE)
		)
			response ProfileRollbackResponse::= {
			#PROFILE_ROLLBACK_RES_CATBUSY
			}
			SW=0x91XX

7	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
8	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000

9	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID2>)
		)
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

10	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

11	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

Test Sequence #4 Error: catBusy with refreshFlag not set while proactive session is ongoing – catBusy supported

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_REQUEST_ENABLE(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		ICCID_OP_PROF2,
		TRUE,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#ENABLE_RES_OK_1
			}
			SW=0x91XX

2	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“UICC Reset”)
3	Repeat IC1 and IC2		

4	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID2>)
		)
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

5	S_Device → eUICC	MTD_SEND_SMS_PP([GET_MNO_SD])	SW=0x91XX

6	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_PROFILE_ROLLBACK_REQUEST(
		FALSE)
		)
			response ProfileRollbackResponse::= {
			#PROFILE_ROLLBACK_RES_CATBUSY
			}
			SW=0x91XX

7	S_Device → eUICC	FETCH 'XX'	SMS POR received SCP80 response status code equal to 0x00 – POR OK
8	S_Device → eUICC	TERMINAL RESPONSE	SW=0x9000

9	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID2>)
		)
			response ProfileInfoListResponse::= profileInfoListOk : {
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

10	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

11	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF2
			SW=0x9000

-

4.2.29 ES10b (IPA -- eUICC): ConfigureAutomaticProfileEnabling

This function is used by the IPA to activate or deactivate automatic Profile enabling in the eUICC and to add or update automatic enabling data used in the automatic Profile enabling.

4.2.29.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

- Section 5.9.17

4.2.29.2 Test Cases

4.2.29.2.1 TC_eUICC_ES10b.ConfigureAutomaticProfileEnabling

Test Sequence #01 Nominal Case- active automatic profile enabling

Test the nominal case when: autoEnableFlag, smdpOid and smdpAddress are present in the command.

Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	Automatic Profile Enabling is not activated

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_CONFIGURE_AUTOMATIC_PROFILE_ENABLING (
		AUTO_ENABLE_FLAG,
		S_SM_DP+_OID1,
		SMDP_ADDRESS1,
		)
		)
			#CONFIG_AUTO_ENABLE_RES_OK
			SW=0x9000

2	S_IPAd → eUICC	Install PROFILE_OPERATIONAL1 with OID value set to S_SM_DP+_OID1 and address value set to SMDP_ADDRESS1	Profile successfully downloaded (i.e. ProfileInstallationResult contains a SuccessResult)

3	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_USING_DD()
		)
			response EnableUsingDDResponse ::= {
			#ENABLE_USING_DD_RESULT_OK
			}
			SW=0x91XX

4	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“UICC Reset”)
5	Repeat IC1 and IC2		

6	S_IPAd → 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 #02 Nominal Case - deactivate automatic profile enabling

Test the nominal case when autoEnableFlag is not present, smdpOid and smdpAddress are present in the command.

Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	Automatic Profile Enabling is activated

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_CONFIGURE_AUTOMATIC _PROFILE_ENABLING (
		NO_PARAM,
		S_SM_DP+_OID1,
		SMDP_ADDRESS1
		)
		)
			#CONFIG_AUTO_ENABLE_RES_OK
			SW=0x9000

2	S_IPAd → eUICC	Install PROFILE_OPERATIONAL1 with OID value set to S_SM_DP+_OID1 and address value set to SMDP_ADDRESS1	Profile successfully downloaded (i.e. ProfileInstallationResult contains a SuccessResult)

3	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_USING_DD()
		)
			response EnableUsingDDResponse ::= {
			#ENABLE_USING_DD_RESULT_AUTO_ENABLE_NOT_AVAILABLE
			}
			SW=0x9000

Test Sequence #03 Nominal Case - fail the check of SD-DP+ parameter

Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	Automatic Profile Enabling is not activated

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_IPAd → eUICC	MTD_STORE_DATA(MTD_CONFIGURE_AUTOMATIC_PROFILE_ENABLING (AUTO_ENABLE_FLAG, S_SM_DP+_OID1, SMDP_ADDRESS1))	#CONFIG_AUTO_ENABLE_RES_OK SW=0x9000
2	S_IPAd → eUICC	Install PROFILE_OPERATIONAL1 with OID value set to S_SM_DP+_OID2 and address value set to SMDP_ADDRESS1	Profile successfully downloaded (i.e. ProfileInstallationResult contains a SuccessResult)

3	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_ENABLE_USING_DD()
		)
			response EnableUsingDDResponse ::= {
			#ENABLE_USING_DD_RESULT_AUTO_ENABLE_NOT_AVAILABLE
			}
			SW=0x9000

4.2.29.2.2 TC_eUICC_ES10b.ConfigureAutomaticProfileEnabling_ErrorCase

Test Sequence #01 Error: Unsigned Auto Enable Configuration Disallowed

Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	No Eim configuration present on card.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_IPAd → eUICC	MTD_STORE_DATA(MTD_CONFIGURE_AUTOMATIC_PROFILE_ENABLING (AUTO_ENABLE_FLAG, S_SM_DP+_OID1, SMDP_ADDRESS1))	#CONFIG_AUTO_ENABLE_RES_CONFIG_DISALLOWED SW=0x9000

Test Sequence #02 Error: Insufficient Memory

This Test Sequence of this error is not defned in this version of the specification.

4.2.30 ES10b (IPA -- eUICC): GetEimConfigurationData

This function is used by the IPA to read eIM Configuration Data stored in the eUICC.

4.2.30.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.2.30.2 Test Cases

4.2.30.2.1 TC_eUICC_ES10b.GetEimConfigurationData

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The eUICC does not contain any eIM Configuration Data.

Test Sequence #01 Nominal Case: empty list of EIM

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_EIM(
		)
		)
			response GetEimConfigurationDataResponse ::= {
			#EIM_CONFIG_DATA_LIST_EMPTY}
			SW=0x9000

Test Sequence #02 Nominal Case: single EIM

Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	#EIM_CONFIG_DATA1 is configured 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(
		MTD_GET_EIM(
		)
		)
			response GetEimConfigurationDataResponse ::= {
			#EIM_CONFIG_DATA_LIST1}
			SW=0x9000

Test Sequence #03 Nominal Case: multiple EIMs

Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	#EIM_CONFIG_DATA1 and #EIM_CONFIG_DATA2 are configured 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(
		MTD_GET_EIM(
		)
		)
			response GetEimConfigurationDataResponse ::= {
			#EIM_CONFIG_DATA_LIST2}
			SW=0x9000

4.2.31 ESep (eIM -- eUICC): eUICC Package with single PSMO command: Enable

This function can be tested as Enable Procedures

4.2.31.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

- Section 2.11.1.1
- Section 2.11.1.1.3
- Section 2.11.2.1
- Section 3.3.1
- Section 5.13.1

4.2.31.2 Test Cases

4.2.31.2.1 TC_eUICC_ESep.Enable

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.
eUICC	S_eIM with #EIM_CONFIG_DATA1 is configured in the eUICC.

Test Sequence #01 Nominal Case: Enable Profile No transaction Id.

The purpose of this test is to verify that a Profile can be Enabled when the transaction Id is not present in the request.

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_IPAd → eUICC	MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_REQUEST_ENABLE(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		ICCID_OP_PROF1,
		NO_PARAM,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#ENABLE_RES_OK_1
			}
			
			SW=0x91XX

2	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“UICC Reset”)
3	Repeat IC1 and IC2		

4	S_IPAd → 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 Case: Enable Profile with transaction Id.

The purpose of this test is to verify that a Profile can be Enabled when the transaction Id is present in the request.

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_IPAd → eUICC	MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_REQUEST_ENABLE(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		S_TRANSACTION_ID,
		ICCID_OP_PROF1,
		NO_PARAM,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#ENABLE_RES_OK_2
			}
			SW=0x91XX

2	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“UICC Reset”)
3	Repeat IC1 and IC2		

4	S_IPAd → 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 #03 Nominal Case: Enable Profile with transaction and the rollbackFlag.

The purpose of this test is to verify that a Profile can be Enabled when the transaction Id and the rollbackFlag are present in the request.

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_IPAd → eUICC	MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_REQUEST_ENABLE(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		S_TRANSACTION_ID,
		ICCID_OP_PROF1,
		TRUE,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#ENABLE_RES_OK_2
			}
			
			SW=0x91XX

2	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“UICC Reset”)
3	Repeat IC1 and IC2		

4	S_IPAd → 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 #04 Nominal Case: Enable Profile Device supports “eUICC Profile State Change”.

The purpose of this test is to verify that a Profile can be Enabled when the transaction Id and the rollbackFlag are present in the request.

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_IPAd → eUICC	MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_REQUEST_ENABLE(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		ICCID_OP_PROF1,
		NO_PARAM,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#ENABLE_RES_OK_1
			}
			SW=0x91XX

2	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“eUICC Profile State change”)
3	Execute IC1 from step 2 to step 4		
4	Repeat IC2		

4	S_IPAd → 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

4.2.31.2.2 TC_eUICC_ESep.Enable_ErrorCases

Test Sequence #01 Error: Command already processed

The purpose of this test is to verify that a Profile cannot be enabled because the command has been already processed.

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	S_eIM with #EIM_CONFIG_DATA1 is configured in the eUICC.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_REQUEST_ENABLE(
		TEST_EIM_ID1,,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		ICCID_OP_PROF1,
		NO_PARAM,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::=
			euiccPackageResultSigned : {
			#ENABLE_RES_OK_1
			}
			SW=0x91XX

2	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“UICC Reset”)
3	Repeat IC1 and IC2		
4	S_Device → eUICC	[SELECT_ICCID]	SW=0x9000

5	S_Device → eUICC	[READ_BINARY] with <L>=0x0A	#ICCID_OP_PROF1
			SW=0x9000

6	S_eIM → eUICC	Resend the same command as in step 1, so the counter is also the same.	response EuiccPackageResult ::=
			euiccPackageErrorDataSigned: {
			#PACKAGE_RES_ERROR_CODE_REPLAY_ERROR
			}
			SW=0x9000

Test Sequence #02 Error: Target profile is not found

The purpose of this test is to verify that a Profile cannot be enabled because the target profile is not found.

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 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_eIM → eUICC	MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_REQUEST_ENABLE(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		ICCID_OP_PROFX,
		NO_PARAM,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#ENABLE_RES_PROFILE_ICCID_NOT_FOUND
			}
			SW=0x9000

2	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response
			ProfileInfoListResponse::=
			profileInfoListOk : {
			#
			PROFILE_INFO1_DISABLED
			}
			SW=0x9000

Test Sequence #03 Error: Target profile is not in a Disable State

The purpose of this test is to verify that a Profile cannot be enabled because the target profile is not in a Disable 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_eIM → eUICC	MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_REQUEST_ENABLE(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		ICCID_OP_PROF1,
		NO_PARAM,
		S_EIM_SIGNATURE
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#
			ENABLE_RES_PROFILE_NOT_IN_DISABLED_STATE
			}
			SW=0x9000

		)	

2	S_IPAd →
	eUICC
		MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response
			ProfileInfoListResponse::=
			profileInfoListOk : {
			# PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #04 Error: Usage of the Rollback Mechanism is granted

The purpose of this test is to verify that a Profile cannot be enabled because the usage of the Rollback Mechanism is granted and there is no currently Enabled Profile.

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The PROFILE_OPERATIONAL1 is Disable on the eUICC.
eUICC	The PROFILE_OPERATIONAL1 corresponds to <ISD_P_AID1>.
eUICC	There is no currently Enabled Profile.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_eIM →
	eUICC
		MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_REQUEST_ENABLE(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		ICCID_OP_PROF1,
		NO_PARAM,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			# ENABLE_RES_PROFILE_UNDEFINED_ERROR
			}
			
			SW=0x9000

4.2.32 ESep (eIM -- eUICC): eUICC Package with single PSMO command: Disable

4.2.32.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

- Section 2.11.1.1
- Section 2.11.1.1.3
- Section 2.11.2.1
- Section 5.13.2
- Section 3.3.1

4.2.32.2 Test Cases

4.2.32.2.1 TC_eUICC_ESep.Disable

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.
eUICC	S_eIM with #EIM_CONFIG_DATA1 is configured in the eUICC.

Test Sequence #01 Nominal Case: Disable Profile No transaction Id

The purpose of this test is to verify that a Profile can be disabled when the transaction Id is not present in the request.

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_eIM → eUICC	MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_REQUEST_DISABLE(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		ICCID_OP_PROF1,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#DISABLE_RES_OK_1
			}
			
			SW=0x91XX

2	S_Device → eUICC	FETCH 'XX'	REFRESH Command ("UICC Reset")
3	Repeat IC1 and IC2		

4	S_IPAd → 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 #02 Nominal Case: Disable Profile when the transaction Id is present

The purpose of this test is to verify that a Profile can be disabled when the transaction Id is present.

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_eIM → eUICC	MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_REQUEST_DISABLE(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		ICCID_OP_PROF1,
		S_TRANSACTION_ID,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#DISABLE_RES_OK_2
			}
			SW=0x91XX

2	S_Device → eUICC	FETCH 'XX'	REFRESH Command ("UICC Reset")

3	Repeat IC1 and IC2		

4	S_IPAd → 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 #03 Nominal Case: Disable Profile when Device supports "eUICC Profile State Change"

The purpose of this test is to verify that a Profile can be disabled with REFRESH type "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_eIM → eUICC	MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_REQUEST_DISABLE(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		ICCID_OP_PROF1,
		S_TRANSACTION_ID,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#DISABLE_RES_OK_2
			}
			SW=0x91XX

2	S_Device → eUICC	FETCH 'XX'	REFRESH Command ("eUICC Profile State changed")
3	Execute IC1 from step 2 to step 4		
4	Repeat IC2		

5	S_IPAd → 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

4.2.32.2.2 TC_eUICC_ESep.Disable_ErrorCases

Test Sequence #01 Error: Command already processed

The purpose of this test is to verify that a Profile cannot be disabled because the command has been already processed.

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_eIM → eUICC	MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_REQUEST_DISABLE(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		ICCID_OP_PROF1,
		NO_PARAM,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::=
			euiccPackageResultSigned : {
			#DISABLE_RES_OK_1
			}
			SW=0x91XX

2	S_Device → eUICC	FETCH 'XX'	REFRESH Command (“UICC Reset”)
3	Repeat IC1 and IC2		
4	S_Device → eUICC	[SELECT_ICCID]	SW=0x6A82

5	S_eIM → eUICC	Resend the same command as in step 1, so the counter is also the same.	response EuiccPackageResult ::= euiccPackageErrorDataSigned: {
			#PACKAGE_RES_ERROR_CODE_REPLAY_ERROR
			}
			SW=0x9000

Test Sequence #02 Error: Target profile is not found

The purpose of this test is to verify that a Profile cannot be disabled because the target profile is not found.

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 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_eIM → eUICC	MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_REQUEST_DISABLE(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		ICCID_OP_PROFX,
		NO_PARAM,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#DISABLE_RES_PROFILE_ICCID_NOT_FOUND
			}
			
			SW=0x9000

2	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response
			ProfileInfoListResponse::=
			profileInfoListOk : {
			#PROFILE_INFO1
			}
			SW=0x9000

Test Sequence #03 Error: Target profile is not in Enabled State

The purpose of this test is to verify that a Profile cannot be disabled because the target profile is not 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>.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_eIM → eUICC	MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_REQUEST_DISABLE(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		ICCID_OP_PROF1,
		NO_PARAM,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#DISABLE_RES_PROFILE_NOT_IN_ENABLED_STATE
			}
			
			SW=0x9000

2	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response
			ProfileInfoListResponse::=
			profileInfoListOk : {
			
			#PROFILE_INFO1_DISABLED
			}
			SW=0x9000

4.2.33 ESep (eIM -- eUICC): eUICC Package with single PSMO command: Delete

4.2.33.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

- Section 2.11.1.1
- Section 2.11.1.1.3
- Section 2.11.2.1
- Section 3.3.1

- Section 5.13.3

4.2.33.2 Test Cases

4.2.33.2.1 TC_eUICC_ESep.Delete

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	The PROFILE_OPERATIONAL1 has been installed on the eUICC.
eUICC	S_eIM with #EIM_CONFIG_DATA1 is configured in the eUICC.

Test Sequence #01 Nominal Case: Delete Profile No transaction Id.

The purpose of this test is to verify that a Profile can be deleted when the transaction Id is not present in the request.

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_eIM → eUICC	MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_REQUEST_DELETE (
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		ICCID_OP_PROF1,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#DELETE_RES_OK_1
			}
			SW=0x9000

2	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response ProfileInfoListResponse::= profileInfoListOk: {
			}
			SW=0x9000

Test Sequence #02 Nominal Case: Delete Profile Transaction Id.

The purpose of this test is to verify that a Profile can be deleted when the transaction Id is present in the request.

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_eIM → eUICC	MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_REQUEST_DELETE (
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		S_TRANSACTION_ID,
		ICCID_OP_PROF1,
		S_EIM_SIGNATURE
		)
		)
			response
			EuiccPackageResult ::=
			euiccPackageResultSigned
			: {
			#DELETE_RES_OK_2
			}
			
			SW=0x9000

2	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_GET_PROFILE_INFO(
		NO_PARAM,
		<ISD_P_AID1>))
			response
			ProfileInfoListResponse::=
			profileInfoListOk: {
			}
			SW=0x9000

4.2.33.2.2 TC_eUICC_ESep.Delete_ErrorCases

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	S_eIM with #EIM_CONFIG_DATA1 is configured in the eUICC.

Test Sequence #01 Error: Target profile is not found

The purpose of this test is to verify that a Profile cannot be deleted because the target profile is not found.

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_eIM → eUICC	MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_REQUEST_DELETE (
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		ICCID_OP_PROFX,
		S_EIM_SIGNATURE
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#DELETE_PROFILE_ICCID_NOT_FOUND
			}
			SW=0x9000

2	S_IPAd → eUICC	MTD_STORE_DATA(
		#GET_PROFILES_INFO_ALL)
			response ProfileInfoListResponse::= profileInfoListOk: {
			#PROFILE_INFO1_DISABLED,
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

Test Sequence #02 Error: Profile not in disable state

The purpose of this test is to verify that a Profile cannot be deleted because the target profile is not in disable state.

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_eIM → eUICC	MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_REQUEST_DELETE (
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		ICCID_OP_PROF2,
		S_EIM_SIGNATURE
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#DELETE_PROFILE_ICCID_NOT_IN_DISABLED_STATE
			}
			SW=0x9000

2	S_IPAd → eUICC	MTD_STORE_DATA(#GET_PROFILES_INFO_ALL)	response ProfileInfoListResponse::= profileInfoListOk: {
			#PROFILE_INFO1_DISABLED,
			#PROFILE_INFO2_ENABLED
			}
			SW=0x9000

4.2.34 ESep (eIM -- eUICC): eUICC Package with single PSMO command: ListProfileInfo

This function allows the eIM to retrieve the list of Profile information for installed Profiles including their current state (Enabled/Disabled) and their associated Profile Metadata..

4.2.34.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

- Section 2.11.1.1
- Section 2.11.1.1.3
- Section 2.11.2.1
- Section 3.3.1
- Section 5.13.4

4.2.34.2 Test Cases

4.2.34.2.1 TC_eUICC_ESep.ListProfileInfo

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 Case: Retrieve all Profiles with no searchCriteria

The purpose of this test is to verify the ListProfileInfo function when the searchCriteria is not present.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_REQUEST_LIST_PRO_FILE_INFO (
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		NO_PARAM,
		NO_PARAM,
		NO_PARAM,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			euiccPackageResultDataSigned {
			eimId paramEIMId,
			counterValue paramCounterValue,
			seqNumber SEQ_NUMBER,
			euiccResult {
			listProfileInfoResult : profileInfoListOk : {
			#PROFILE_INFO1,
			#PROFILE_INFO2,
			#PROFILE_INFO3
			}
			}
			},
			euiccSignEPR EUICC_SIGN_EPR
			}
			SW = 0x9000

Test Sequence #02 Nominal Case: Retrieve Profile by AID

The purpose of this test is to verify the ListProfileInfo function when the AID criteria is provided.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_REQUEST_LIST_PRO_FILE_INFO (
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		ISD_P_AID1,
		NO_PARAM,
		NO_PARAM,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			euiccPackageResultDataSigned {
			eimId paramEIMId,
			counterValue paramCounterValue,
			seqNumber SEQ_NUMBER,
			euiccResult {
			listProfileInfoResult : profileInfoListOk : {
			#PROFILE_INFO1
			}
			}
			},
			euiccSignEPR EUICC_SIGN_EPR
			}
			SW = 0x9000

Test Sequence #03 Nominal Case: Retrieve Profile by Iccid

The purpose of this test is to verify the ListProfileInfo function when the IccID criteria is provided

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_REQUEST_LIST_PRO_FILE_INFO (
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		ICCID_OP_PROF2,
		NO_PARAM,
		NO_PARAM,
		NO_PARAM,
		S_EIM_SIGNATURE
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			euiccPackageResultDataSigned {
			eimId paramEIMId,
			counterValue paramCounterValue,
			seqNumber SEQ_NUMBER,
			euiccResult {
			listProfileInfoResult : profileInfoListOk : {
			#PROFILE_INFO2
			}

		)	}
			},
			euiccSignEPR EUICC_SIGN_EPR
			}
			SW = 0x9000

Test Sequence #04 Nominal Case: Retrieve Profiles by profileClass

The purpose of this test is to verify the ListProfileInfo function when the profileClass criteria is provided

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_REQUEST_LIST_PRO_FILE_INFO (
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		NO_PARAM,
		operational,
		NO_PARAM,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			euiccPackageResultDataSigned {
			eimId paramEIMId,
			counterValue paramCounterValue,
			seqNumber SEQ_NUMBER,
			euiccResult {
			listProfileInfoResult : profileInfoListOk : {
			#PROFILE_INFO1,
			#PROFILE_INFO2,
			#PROFILE_INFO3
			}
			}
			},
			euiccSignEPR EUICC_SIGN_EPR
			}
			SW = 0x9000

Test Sequence #05 Nominal Case: Retrieve Profiles with tagList

The purpose of this test is to verify the ListProfileInfo function when the tagList criteria is provided

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		
1	S_IPAd → eUICC	MTD_STORE_DATA(	response EuiccPackageResult ::= euiccPackageResultSigned : {

		MTD_EUICC_PACKAGE_REQUEST_LIST_PRO_FILE_INFO (
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		NO_PARAM,
		NO_PARAM,
		TAGS_AID_PROFILENAME,
		S_EIM_SIGNATURE
		)
		)
			euiccPackageResultDataSigned {
			eimId paramEIMId,
			counterValue paramCounterValue,
			seqNumber SEQ_NUMBER,
			euiccResult {
			#PROFILE_INFO_TAG_LIST
			},
			euiccSignEPR EUICC_SIGN_EPR
			}
			SW = 0x9000

4.2.34.2.2 TC_eUICC_ESep.ListProfileInfo_ErrorCases

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_OPERATIONAL1 is Enabled.

Test Sequence #01 Error: Incorrect Input Values

The purpose of this test is to verify the ListProfileInfo function when the tagList value is invalid.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_REQUEST_LIST_PRO_FILE_INFO (
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		NO_PARAM,
		NO_PARAM,
		PROFILE_TAG_INVALID,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			euiccPackageResultDataSigned {
			eimId paramEIMId,
			counterValue paramCounterValue,
			seqNumber SEQ_NUMBER,
			euiccResult {
			listProfileInfoResult : profileInfoListError : 1
			}
			} , euiccSignEPR EUICC_SIGN_EPR
			}
			SW = 0x9000

Test Sequence #02 Error: Proflie Change Ongoing

The purpose of this test is to verify that a listProfileInfo function SHALL NOT occur after an enable command in an eUICC Package.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD LIST_PROFILE_INFO CHANGE_ONGOING_ERROR (
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			euiccPackageResultDataSigned {
			eimId paramEIMId,
			counterValue paramCounterValue,
			seqNumber SEQ_NUMBER,
			euiccResult {
			enableResult : 0,
			listProfileInfoResult : profileInfoListError : 11
			}
			},
			euiccSignEPR EUICC_SIGN_EPR
			}
			SW = 0x9000

4.2.35 ESep (eIM -- eUICC): eUICC Package with single PSMO command: GetRat

This function allows the eIM to retrieve the Rules Authorisation Table (RAT) from the eUICC.

4.2.35.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

- Section 2.11.1.1
- Section 2.11.1.1.3
- Section 2.11.2.1
- Section 3.3.1
- Section 5.13.5

4.2.35.2 Test Cases

4.2.35.2.1 TC_eUICC_ESep.GetRat

Test Sequence #01 Nominal Case

Initial Conditions	
Entity	**Description of the initial condition**
eUICC	The EUM has configured the eUICC's RAT as defined in section G.1.4.

Step	Direction	Sequence / Description	Expected result
IC1	PROC_EUICC_INITIALIZATION_SEQUENCE		
IC2	PROC_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR		

1	S_IPAd → eUICC	MTD_STORE_DATA(
		MTD_EUICC_PACKAGE_GET_RAT (
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			euiccPackageResultDataSigned {
			eimId paramEIMId,
			counterValue paramCounterValue,
			seqNumber SEQ_NUMBER,
			euiccResult {
			getRATResult : {
			#PPRS_ALLOWED
			}
			}
			},
			euiccSignEPR EUICC_SIGN_EPR
			}
			SW = 0x9000

4.2.36 ESep (eIM -- eUICC): eUICC Package with single eCO command: AddEim

This function adds an Associated eIM to the eUICC by providing its eIM Configuration Data including the eimID to the eUICC.

4.2.36.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.2.36.2 Test Cases

4.2.36.2.1 TC_eUICC_ESep.AddEim

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	S_eIM with #EIM_CONFIG_DATA1 is configured in the eUICC.

Test Sequence #01 Nominal Case: AddEim no eimTransactionId

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_EUICC_PACKAGE_REQUEST_ADD_EIM(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		EIM_CONFIG_DATA2,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#ADD_EIM_RES_OK_1
			}
			
			SW=0x9000

Test Sequence #02 Nominal Case: AddEim with eimTransactionId

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_EUICC_PACKAGE_REQUEST_ADD_EIM(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		S_EIM_TRANSACTION_ID,
		EIM_CONFIG_DATA2,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#ADD_EIM_RES_OK_2
			}
			
			SW=0x900

4.2.36.2.2 TC_eUICC_ESep.AddEim_ErrorCases

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	S_eIM with #EIM_CONFIG_DATA1 is configured in the eUICC.

Test Sequence #01 Error: Insufficient Memory

The test sequence of this error is not defined for this version of the specification.

Test Sequence #02 Error: CI PK Unknown

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_EUICC_PACKAGE_REQUEST_ADD_EIM(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		EIM_CONFIG_DATA2_ADD_UNKNOWN_EUICC_CI_PKID,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#ADD_EIM_RES_ERR_CI_PK_UNKNOWN
			}
			
			SW=0x9000

Test Sequence #03 Error: invalid Association Token

The test sequence of this error is not defined for this version of the specification.

Test Sequence #04 Error: counter Value Out Of Range

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_EUICC_PACKAGE_REQUEST_ADD_EIM(
		TEST_EIM_ID1,
		EID1,
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#ADD_EIM_RES_ERR_COUNTER_OUT_OF_RANGE
			}

		S_COUNTER_VALUE, NO_PARAM, EIM_CONFIG_DATA2_ADD_COUNTER_OUT_OF_RANGE, S_EIM_SIGNATURE))	SW=0x9000

Test Sequence #05 Error: no eimId is provided

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_EUICC_PACKAGE_REQUEST_ADD_EIM(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		EIM_CONFIG_DATA_NO_EIM_ID,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#ADD_EIM_RES_ERR_COMMAND_ERROR
			}
			
			SW=0x9000

Test Sequence #06 Error: eimId exceeds the limit

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_EUICC_PACKAGE_REQUEST_ADD_EIM(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		EIM_CONFIG_DATA2_EIM_ID_129_CHARACTERS,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#ADD_EIM_RES_ERR_COMMAND_ERROR
			}
			
			SW=0x9000

Test Sequence #07 Error: eimId already presents

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_EUICC_PACKAGE_REQUEST_ADD_EIM(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		EIM_CONFIG_DATA1,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#ADD_EIM_RES_ERR_COMMAND_ERROR
			}
			
			SW=0x9000

4.2.37 ESep (eIM -- eUICC): eUICC Package with single eCO command: UpdateEim

This function updates eIM Configuration Data, i.e., the public key or Certificate and the related anti-replay counter value of an Associated eIM with a given eimID within the eUICC while keeping the same eimID.

4.2.37.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.2.37.2 Test Cases

4.2.37.2.1 TC_eUICC_ESep.UpdateEIM

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	S_eIM with #EIM_CONFIG_DATA1 is configured in the eUICC.

Test Sequence #01 Nominal Case: UpdateEim no eimTransactionId

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(	response EuiccPackageResult ::= euiccPackageResultSigned : {

		MTD_EUICC_PACKAGE_REQUEST_UPDATE_EIM(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		EIM_CONFIG_DATA1_UPDATE_COUNTER,
		S_EIM_SIGNATURE
		)
		)
			#UPDATE_EIM_RES_OK_1
			}
			
			SW=0x9000

Test Sequence #02 Nominal Case: UpdateEim with eimTransactionId

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_EUICC_PACKAGE_REQUEST_UPDATE_EIM(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		S_EIM_TRANSACTION_ID,
		EIM_CONFIG_DATA1_UPDATE_COUNTER,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#UPDATE_EIM_RES_OK_2
			}
			
			SW=0x9000

4.2.37.2.2 TC_eUICC_ESep.UpdateEim_ErrorCases

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	S_eIM with #EIM_CONFIG_DATA1 is configured in the eUICC.

Test Sequence #01 Error: EIM not Found

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_EUICC_PACKAGE_REQUEST_UPDATE_EIM(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		EIM_CONFIG_DATA2_UPDATE_COUNTER,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#UPDATE_EIM_RES_ERR_EIM_NOT_FOUND
			}
			
			SW=0x9000

Test Sequence #02 Error: CI PK Unknown

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_EUICC_PACKAGE_REQUEST_UPDATE_EIM(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		EIM_CONFIG_DATA1_UPDATE_UNKNOWN_EUICC_CI_PKID,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#UPDATE_EIM_RES_ERR_UNKNOWN_EUICC_CI_PKID
			}
			
			SW=0x9000

Test Sequence #03 Error: Counter Value Out Of Range

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_EUICC_PACKAGE_REQUEST_UPDATE_EIM(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		EIM_CONFIG_DATA1_UPDATE_COUNTER_OUT_OF_RANGE,
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#UPDATE_EIM_RES_ERR_COUNTER_OUT_OF_RANGE
			}
			
			SW=0x9000

		S_EIM_SIGNATURE))	

Test Sequence #04 Error: Lowering Counter Value

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_EUICC_PACKAGE_REQUEST_UPDATE_EIM(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		EIM_CONFIG_DATA1_UPDATE_LOWER_COUNTER,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#UPDATE_EIM_RES_ERR_COMMAND_ERROR
			}
			
			SW=0x9000

4.2.38 ESep (eIM -- eUICC): eUICC Package with single eCO command: DeleteEim

This function deletes an Associated eIM identified by its eimID from the eUICC. If the successfully deleted Associated eIM was the last available Associated eIM, the eUICC SHALL allow ES10b.AddInitialEim again.

4.2.38.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.2.38.2 Test Cases

4.2.38.2.1 TC_eUICC_ESep.DeleteEim

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	S_eIM with #EIM_CONFIG_DATA1 is configured in the eUICC. #EIM_CONFIG_DATA2 is configured in the eUICC.

Test Sequence #01 Nominal Case: DeleteEim no eimTransactionId

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_EUICC_PACKAGE_REQUEST_DELETE_EIM(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		TEST_EIM_ID2,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#DELETE_EIM_RES_OK_1
			}
			
			SW=0x9000

Test Sequence #02 Nominal Case: DeleteEim with eimTransactionId

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_EUICC_PACKAGE_REQUEST_DELETE_EIM(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		S_EIM_TRANSACTION_ID,
		TEST_EIM_ID2,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#DELETE_EIM_RES_OK_2
			}
			
			SW=0x9000

4.2.38.2.2 TC_eUICC_ESep.DeleteEim_ErrorCases

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	S_eIM with #EIM_CONFIG_DATA1 is configured in the eUICC.

Test Sequence #01 Error: EIM Not Found

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_EUICC_PACKAGE_REQUEST_DELETE_EIM(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		S_TRANSACTION_ID,
		TEST_EIM_ID2,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#DELETE_EIM_RES_ERR_EIM_NOT_FOUND
			}
			
			SW=0x9000

Test Sequence #02 Error: Last EIM Deleted

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_EUICC_PACKAGE_REQUEST_DELETE_EIM(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		S_TRANSACTION_ID,
		TEST_EIM_ID1,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#DELETE_EIM_RES_ERR_LAST_EIM_DELETED
			}
			
			SW=0x9000

Test Sequence #03 Error: eimId is provided

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_EUICC_PACKAGE_REQUEST_DELETE_EIM(
		TEST_EIM_ID1,
		EID1,
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#DELETE_EIM_RES_ERR_COMMAND_ERROR
			}

		S_COUNTER_VALUE, S_TRANSACTION_ID, NO_PARAM, S_EIM_SIGNATURE))	SW=0x9000

4.2.39 ESep (eIM -- eUICC): eUICC Package with single eCO command: ListEim

This function requests the eUICC to provide a list of all currently configured Associated eIMs to the eIM.

4.2.39.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.2.39.2 Test Cases

4.2.39.2.1 TC_eUICC_ESep.ListEim

General Initial Conditions	
Entity	**Description of the general initial condition**
eUICC	S_eIM with #EIM_CONFIG_DATA1 is configured in the eUICC. #EIM_CONFIG_DATA2 is configured in the eUICC.

Test Sequence #01 Nominal Case: ListEim no eimTransactionId

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_EUICC_PACKAGE_REQUEST_LIST_EIM(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		NO_PARAM,
		NO_PARAM,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#LIST_EIM_RES_OK_1
			}
			
			SW=0x9000

Test Sequence #02 Nominal Case: ListEim with eimTransactionId

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_EUICC_PACKAGE_REQUEST_LIST_EIM(
		TEST_EIM_ID1,
		EID1,
		S_COUNTER_VALUE,
		S_EIM_TRANSACTION_ID,
		NO_PARAM,
		S_EIM_SIGNATURE
		)
		)
			response EuiccPackageResult ::= euiccPackageResultSigned : {
			#LIST_EIM_RES_OK_2
			}
			
			SW=0x9000

4.2.39.2.2 TC_eUICC_ESep.ListEim_ErrorCases

Test Sequence #01 Error: Command Error

The test sequence of this error is not defined in this version of the specification.

4.3 IPAe Interfaces

4.3.1 ES9+ (IPAe -- SM-DP+): InitiateAuthentication

ES9+ interface between the IPAe and SM-DP+ is identical to the LPAd and SM-DP+.

4.3.1.1 Conformance Requirements

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.3.1.2 Test Cases

4.3.1.2.1 TC_LPAe_InitiateAuthentication_Nominal

Test Sequence #01 Nominal: Initiate Authentication

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: Initiate Authentication* defined in section 4.4.21.2.1 TC_LPAe_InitiateAuthentication_Nominal where the LPAd plays the IPAe role

4.3.1.2.2 TC_LPAd_InitiateAuthentication_ErrorCases

Test Sequence #01 Error: Invalid SM-DP+ Address

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Error: Invalid SM-DP+ Address* defined in section 4.4.21.2.2 TC_LPAe_InitiateAuthentication_ErrorCases where the LPAd plays the IPAe role

Test Sequence #02 Error: Unsupported Security Configuration

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Error: Unsupported Security Configuration* defined in section 4.4.21.2.2 TC_LPAe_InitiateAuthentication_ErrorCases where the LPAd plays the IPAe role

Test Sequence #03 Error: Unsupported SVN

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #03 Error: Unsupported SVN* defined in section 4.4.21.2.2 TC_LPAe_InitiateAuthentication_ErrorCases where the LPAd plays the IPAe role

Test Sequence #04 Error: Unavailable SM-DP+ Certificate

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #04 Error: Unavailable SM-DP+ Certificate* defined in section 4.4.21.2.2 TC_LPAe_InitiateAuthentication_ErrorCases where the LPAd plays the IPAe role

Test Sequence #05 Error: Invalid SM-DP+ Certificate

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #05 Error: Invalid SM-DP+ Certificate* defined in section 4.4.21.2.2 TC_LPAe_InitiateAuthentication_ErrorCases where the LPAd plays the IPAe role

Test Sequence #06 Error: Invalid SM-DP+ Signature

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #06 Error: Invalid SM-DP+ Signature* defined in section 4.4.21.2.2 TC_LPAe_InitiateAuthentication_ErrorCases where the LPAd plays the IPAe role

Test Sequence #07 Error: Invalid SM-DP+ Address sent by the SM-DP+

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #07 Error: Invalid SM-DP+ Address sent by the SM-DP+* defined in section 4.4.21.2.2 TC_LPAe_InitiateAuthentication_ErrorCases where the LPAd plays the IPAe role

Test Sequence #08 Error: Unsupported CI Key ID

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #08 Error: Unsupported CI Key ID* defined in section 4.4.21.2.2 TC_LPAe_InitiateAuthentication_ErrorCases where the LPAd plays the IPAe role

Test Sequence #09 Error: Invalid SM-DP+ OID

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #09 Error: Invalid SM-DP+ OID* defined in section 4.4.21.2.2 TC_LPAe_InitiateAuthentication_ErrorCases where the LPAd plays the IPAe role

4.3.2 ES9+ (IPAe -- SM-DP+): GetBoundProfilePackage

4.3.2.1 Conformance Requirements

ES9+ interface between the IPAe and SM-DP+ is identical to the LPAd and SM-DP+.

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.3.2.2 Test Cases

4.3.2.2.1 TC_LPAd_ES9+_GetBoundProfilePackage_Nominal

Test Sequence #01 Nominal: Get BPP using S-ENC and S-MAC without Confirmation Code

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: Get BPP using S-ENC and S-MAC without Confirmation Code* defined in section 4.4.22.2.1 TC_LPAd_ES9+_GetBoundProfilePackage_Nominal where the LPAd plays the IPAe role

Test Sequence #02 Nominal: Get BPP using S-ENC and S-MAC with Confirmation Code

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Nominal: Get BPP using S-ENC and S-MAC with Confirmation Code* defined in section 4.4.22.2.1 TC_LPAd_ES9+_GetBoundProfilePackage_Nominal where the LPAd plays the IPAe role

Test Sequence #03 Nominal: Get BPP using PPK-ENC and PPK-MAC without Confirmation Code

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #03 Nominal: Get BPP using PPK-ENC and PPK-MAC without Confirmation Code* defined in section 4.4.22.2.1 TC_LPAd_ES9+_GetBoundProfilePackage_Nominal where the LPAd plays the IPAe role

Test Sequence #04 Nominal: Get BPP using PPK-ENC and PPK-MAC with Confirmation Code

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #04 Nominal: Get BPP using PPK-ENC and PPK-MAC with Confirmation Code* defined in section 4.4.22.2.1 TC_LPAd_ES9+_GetBoundProfilePackage_Nominal where the LPAd plays the IPAe role

4.3.2.2.2 TC_LPAd_ES9+_GetBoundProfilePackage_Retry

Test Sequence #01 Nominal: Get BPP Retry after incorrect Confirmation Code

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: Get BPP Retry after incorrect Confirmation Code*defined in section 4.4.22.2.2 TC_LPAd_ES9+_GetBoundProfilePackage_Retry where the LPAd plays the IPAe role

4.3.2.2.3 TC_LPAd_ES9+_GetBoundProfilePackage_Error

Test Sequence #01 Error: Wrong eUICC Signature

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Error: Wrong eUICC Signature* defined in section 4.4.22.2.3 TC_LPAd_ES9+_GetBoundProfilePackage_Error where the LPAd plays the IPAe role

Test Sequence #02 Error: BPP Not Available

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Error: BPP Not Available* defined in section 4.4.22.2.3 TC_LPAd_ES9+_GetBoundProfilePackage_Error where the LPAd plays the IPAe role

Test Sequence #03 Error: Unknown TransactionID received by SM-DP+

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #03 Error: Unknown TransactionID received by SM-DP+* defined in section 4.4.22.2.3 TC_LPAd_ES9+_GetBoundProfilePackage_Error where the LPAd plays the IPAe role

Test Sequence #04 Error: Missing Confirmation Code

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #04 Error: Missing Confirmation Code received by SM-DP+* defined in section 4.4.22.2.3 TC_LPAd_ES9+_GetBoundProfilePackage_Error where the LPAd plays the IPAe role

Test Sequence #05 Error: Download Order Expired

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #05 Error: Download Order Expired* defined in section 4.4.22.2.3 TC_LPAd_ES9+_GetBoundProfilePackage_Error where the LPAd plays the IPAe role

Test Sequence #06 Error: Wrong Confirmation Code

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #06 Error: Wrong Confirmation Code* defined in section 4.4.22.2.3 TC_LPAd_ES9+_GetBoundProfilePackage_Error where the LPAd plays the IPAe role

Test Sequence #07 Error: Maximum number of Confirmation Code retries exceeded

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #07 Error: Maximum number of Confirmation Code retries exceeded* defined in section 4.4.22.2.3 TC_LPAd_ES9+_GetBoundProfilePackage_Error where the LPAd plays the IPAe role

4.3.3 ES9+ (IPAe -- SM-DP+): AuthenticateClient

4.3.3.1 Conformance Requirements

ES9+ interface between the IPAe and SM-DP+ is identical to the LPAd and SM-DP+.

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.3.3.2 Test Cases

4.3.3.2.1 TC_IPAe_AuthenticateClient_Nominal

Test Sequence #01 Nominal: Authenticate Client without Confirmation Code

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: Authenticate Client without Confirmation Code* defined in section 4.4.23.2.1 TC_LPAd_AuthenticateClient_Nominal where the LPAd plays the IPAe role

Test Sequence #02 Nominal: Authenticate Client with Confirmation Code

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Nominal: Authenticate Client with Confirmation Code* defined in section 4.4.23.2.1 TC_LPAd_AuthenticateClient_Nominal where the LPAd plays the IPAe role

Test Sequence #03 Nominal: Authenticate Client with Confirmation Code Retry

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #03 Nominal: Authenticate Client with Confirmation Code Retry* defined in section 4.4.23.2.1 TC_LPAd_AuthenticateClient_Nominal where the LPAd plays the IPAe role

4.3.3.2.2 TC_IPAe_AuthenticateClient_ErrorCases

Test Sequence #01 Error: Invalid EUM Certificate

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Error: Invalid EUM Certificate* defined in section 4.4.23.2. TC_LPAd_AuthenticateClient_ErrorCases where the LPAd plays the IPAe role

Test Sequence #02 Error: Expired EUM Certificate

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Error: Expired EUM Certificate* defined in section 4.4.23.2. TC_LPAd_AuthenticateClient_ErrorCases where the LPAd plays the IPAe role

Test Sequence #03 Error: Invalid eUICC Certificate

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #03 Error: Invalid eUICC Certificate* defined in section 4.4.23.2. TC_LPAd_AuthenticateClient_ErrorCases where the LPAd plays the IPAe role

Test Sequence #04 Error: Expired eUICC Certificate

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #04 Error: Expired eUICC Certificate* defined in section 4.4.23.2. TC_LPAd_AuthenticateClient_ErrorCases where the LPAd plays the IPAe role

Test Sequence #05 Error: Invalid eUICC Signature or serverChallenge

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #05 Error: Invalid eUICC Signature or serverChallenge* defined in section 4.4.23.2. TC_LPAd_AuthenticateClient_ErrorCases where the LPAd plays the IPAe role

Test Sequence #06 Error: Insufficient Memory

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #06 Error: Insufficient Memory* defined in section 4.4.23.2. TC_LPAd_AuthenticateClient_ErrorCases where the LPAd plays the IPAe role

Test Sequence #07 Error: Unknown CI Root Key

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #06 Error: Insufficient Memory* defined in section 4.4.23.2. TC_LPAd_AuthenticateClient_ErrorCases where the LPAd plays the IPAe role

Test Sequence #08 Error: Profile not Allowed (Not in 'released' State)

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #08 Error: Profile not Allowed (Not in 'released' State)* defined in section 4.4.23.2. TC_LPAd_AuthenticateClient_ErrorCases where the LPAd plays the IPAe role

Test Sequence #09 Error: Unknown TransactionID

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #09 Error: Unknown TransactionID* defined in section 4.4.23.2. TC_LPAd_AuthenticateClient_ErrorCases where the LPAd plays the IPAe role

Test Sequence #10 Error: Refused MatchingID

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #10 Error: Refused MatchingID* defined in section 4.4.23.2. TC_LPAd_AuthenticateClient_ErrorCases where the LPAd plays the IPAe role

Test Sequence #11 Error: Refused EID

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #11 Error: Refused EID* in section 4.4.23.2. TC_LPAd_AuthenticateClient_ErrorCases where the LPAd plays the IPAe role

Test Sequence #12 Error: No Eligible Profile for this eUICC/Device

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #12 Error: No Eligible Profile for this eUICC/Device* in section 4.4.23.2. TC_LPAd_AuthenticateClient_ErrorCases where the LPAd plays the IPAe role

Test Sequence #13 Error: Expired Download Order

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #13 Error: Expired Download Order* in section 4.4.23.2. TC_LPAd_AuthenticateClient_ErrorCases where the LPAd plays the IPAe role

Test Sequence #14 Error: Maximum Number of Retries Exceeded

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #14 Error: Maximum Number of Retries Exceeded* in section 4.4.23.2. TC_LPAd_AuthenticateClient_ErrorCases where the LPAd plays the IPAe role

Test Sequence #15 Error: Invalid SM-DP+(pb) certificate

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #15 Error: Invalid SM-DP+(pb) certificate* in section 4.4.23.2. TC_LPAd_AuthenticateClient_ErrorCases where the LPAd plays the IPAe role

Test Sequence #16 Error: Different OID for SM-DP+ Certificates (CERT.DPpb.ECDSA and CERT.DPauth.ECDSA not belonging to the same entity)

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #16 Error: Different OID for SM-DP+ Certificates (CERT.DPpb.ECDSA and CERT.DPauth.ECDSA not belonging to the same entity)* in section 4.4.23.2. TC_LPAd_AuthenticateClient_ErrorCases where the LPAd plays the IPAe role

Test Sequence #17 Error: Invalid SM-DP+ signature (smdpSignature2)

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #17 Error: Invalid SM-DP+ signature (smdpSignature2)* defined in section 4.4.23.2. TC_LPAd_AuthenticateClient_ErrorCases where the LPAd plays the IPAe role

Test Sequence #18 Error: Invalid TransactionID sent by SM-DP+

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #18 Error: Invalid TransactionID sent by SM-DP+* defined in section 4.4.23.2. TC_LPAd_AuthenticateClient_ErrorCases where the LPAd plays the IPAe role

4.3.4 ES9+ (IPAe – SM-DP+): HandleNotification

4.3.4.1 Conformance Requirements

ES9+ interface between the IPAe and SM-DP+ is identical to the LPAd and SM-DP+.

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.3.4.2 Test Cases

4.3.4.2.1 TC_IPAe_ES9+_HandleNotification_Nominal

Throughout all the 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.

Test Sequence #01 Nominal: Successful PIR and Install Notifications to the Same SM-DP+ Address

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: Successful PIR and Install Notifications to the Same SM-DP+ Address* in section4.4.24.2.1 TC_LPAd_ES9+_HandleNotification_Nominal where the LPAd plays the IPAe role

Test Sequence #02 Nominal: Successful PIR and Enable Notifications to the Same SM-DP+ Address

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Nominal: Successful PIR and Enable Notifications to the Same SM-DP+ Address* in section4.4.24.2.1 TC_LPAd_ES9+_HandleNotification_Nominal where the LPAd plays the IPAe role

Test Sequence #03 Nominal: Disable and Delete Notifications to the Same SM-DP+ Address

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #03 Nominal: Disable and Delete Notifications to the Same SM-DP+ Address* in section4.4.24.2.1 TC_LPAd_ES9+_HandleNotification_Nominal where the LPAd plays the IPAe role

Test Sequence #04 Nominal: Enable and Disable Notifications with Different SM-DP+ Addresses

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #04 Nominal: Enable and Disable Notifications with Different SM-DP+ Addresses* in section4.4.24.2.1 TC_LPAd_ES9+_HandleNotification_Nomina lwhere the LPAd plays the IPAe role

Test Sequence #05 Nominal: Different SM-DP+ Addresses in PIR and Install Notifications

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #05 Nominal: Different SM-DP+ Addresses in PIR and Install Notifications* in section 4.4.24.2.1 TC_LPAd_ES9+_HandleNotification_Nominal where the LPAd plays the IPAe role

Test Sequence #06 Nominal: Profile Download with PIR Failed

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #06 Nominal: Profile Download with PIR Failed* in section 4.4.24.2.1 TC_LPAd_ES9+_HandleNotification_Nominal where the LPAd plays the IPAe role

Test Sequence #07 Nominal: Successful PIR and Install Notifications after Connectivity Interruption

This Test Sequence is FFS as indicated in SGP.23 [8].

Test Sequence #08 Nominal: No Acknowledge for Successful PIR results in No Further Notifications

The purpose of this test case is to verify that the next Notification of a group is not sent until LPA receives a successful response from the SM-DP+ for the previous Notification.

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #08 Nominal: No Acknowledge for Successful PIR results in No Further Notifications* in section 4.4.24.2.1 TC_LPAd_ES9+_HandleNotification_Nominal where the LPAd plays the IPAe role

Test Sequence #09 Nominal: Disable and Delete Notifications to the Same SM-DP+ Address using Delete Operation

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #09 Nominal: Disable and Delete Notifications to the Same SM-DP+ Address using Delete Operation* in section 4.4.24.2.1 TC_LPAd_ES9+_HandleNotification_Nominal where the LPAd plays the IPAe role

Test Sequence #10 Nominal: Disable and Delete Notifications with Different SM-DP+ Addresses

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #10 Nominal: Disable and Delete Notifications with Different SM-DP+ Addresses* in section 4.4.24.2.1 TC_LPAd_ES9+_HandleNotification_Nominal where the LPAd plays the IPAe role

4.3.5 ES9+ (IPAe – SM-DP+): CancelSession

4.3.5.1 Conformance Requirements

ES9+ interface between the IPAe and SM-DP+ is identical to the LPAd and SM-DP+.

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.3.5.2 Test Cases

4.3.5.2.1 TC_IPAe_ES9+_CancelSession_Nominal

Test Sequence #01 Nominal: Profile Download with PPR1 not allowed due to Operational Profile already present

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: Profile Download with PPR1 not allowed due to Operational Profile already present* in section 4.4.25.2.1 TC_LPAd_ES9+_CancelSession_Nominal where the LPAd plays the IPAe role

Test Sequence #02 Nominal: End User rejection

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Nominal: End User rejection* in section 4.4.25.2.1 TC_LPAd_ES9+_CancelSession_Nominal where the LPAd plays the IPAe role

Test Sequence #03 Nominal: Load BPP Error

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #03 Nominal: Load BPP Error* in section 4.4.25.2.1 TC_LPAd_ES9+_CancelSession_Nominal where the LPAd plays the IPAe role

Test Sequence #04 Nominal: End User Timeout

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #04 Nominal: End User Timeout* in section 4.4.25.2.1 TC_LPAd_ES9+_CancelSession_Nominal where the LPAd plays the IPAe role

Test Sequence #05 Nominal: Load BPP Error due to unknown TAG

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #05 Nominal: Load BPP Error due to unknown TAG* in section 4.4.25.2.1 TC_LPAd_ES9+_CancelSession_Nominal where the LPAd plays the IPAe role

4.3.5.2.2 TC_IPAe_ES9+_CancelSession_EndUserPostponed_Nominal

Test Sequence #01 Nominal: End User Postponed

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: End User Postponed* in section 4.4.25.2.2 TC_LPAd_ES9+_CancelSession_EndUserPostponed_Nominal where the LPAd plays the IPAe role

4.3.5.2.3 TC_IPAe_ES9+_CancelSession_Error

Test Sequence #01 Error: Unknown TransactionID after End User Rejection/Postpone

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Error: Unknown TransactionID after End User Rejection/Postpone* in section 4.4.25.2.3 TC_LPAd_ES9+_CancelSession_Error where the LPAd plays the IPAe role

Test Sequence #02 Error: Invalid eUICC Signature after End User Rejection/Postpone

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Error: Invalid eUICC Signature after End User Rejection/Postpone* in section 4.4.25.2.3 TC_LPAd_ES9+_CancelSession_Error where the LPAd plays the IPAe role

Test Sequence #03 Error: Invalid SM-DP+ OID after End User Rejection/Postpone

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #03 Error: Invalid SM-DP+ OID after End User Rejection/Postpone* in section 4.4.25.2.3 TC_LPAd_ES9+_CancelSession_Error where the LPAd plays the IPAe role

4.3.5.2.4 TC_IPAe_ES9+_CancelSession_PPRs

Test Sequence #01 Nominal: End User rejection/postpone after PPR1 consent requested

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: End User rejection/postpone after PPR1 consent requested* in section 4.4.25.2.4 TC_LPAd_ES9+_CancelSession_PPRs where the LPAd plays the IPAe role

Test Sequence #02 Nominal: End User rejection/posptone after PPR2 consent requested

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Nominal: End User rejection/posptone after PPR2 consent requested* in section 4.4.25.2.4 TC_LPAd_ES9+_CancelSession_PPRs where the LPAd plays the IPAe role

4.3.6 ES9+ (IPAe – SM-DP+): HTTPS

4.3.6.1 Conformance Requirements

ES9+ interface between the IPAe and SM-DP+ is identical to the LPAd and SM-DP+.

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.3.6.2 Test Cases

4.3.6.2.1 TC_IPAe_HTTPS_Nominal

Test Sequence #01 Nominal: HTTPS Session Establishment

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: HTTPS Session Establishment* in section 4.4.26.2.1TC_LPAd_HTTPS_Nominal where the LPAd plays the IPAe role

Test Sequence #02 Nominal: non-reuse of session keys

The purpose of this test sequence is to verify that the IPAe is not reusing ephemeral keys from the previous session.

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Nominal: non-reuse of session keys* in section 4.4.26.2.1 TC_LPAd_HTTPS_Nominal where the LPAd plays the IPAe role

4.3.6.2.2 TC_IPAe_HTTPS_ErrorCases

Test Sequence #01 Error: Invalid (SM-DP+) TLS Certificate signature

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Error: Invalid (SM-DP+) TLS Certificate signature* in section 4.4.26.2.2 TC_LPAd_HTTPS_ErrorCases where the LPAd plays the IPAe role

Test Sequence #02 Error: Expired TLS Certificate

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Error: Expired TLS Certificate* in section 4.4.26.2.2TC_LPAd_HTTPS_ErrorCases where the LPAd plays the IPAe role

Test Sequence #03 Error: VOIDTest Sequence #04 Error: VOID

Test Sequence #05 Error: VOID

Test Sequence #06 Error: VOID

Test Sequence #07 Error: Invalid TLS Certificate based on Invalid CI (Invalid Curve)

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #07 Error: Invalid TLS Certificate based on Invalid CI (Invalid Curve)* in section 4.4.26.2.2 TC_LPAd_HTTPS_ErrorCases where the LPAd plays the IPAe role

4.3.7 ES11 (IPAe – SM-DS): InitiateAuthentication

4.3.7.1 Conformance Requirements

ES11 interface between the IPAe and SM-DS is identical to the LPAd and SM-DS.

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.3.7.2 Test Cases

4.3.7.2.1 TC_IPAe_ES11_InitiateAuthentication_Nominal

Test Sequence #01 Nominal: Initiate Authentication

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: Initiate Authentication* in section 4.4.27.2.1 TC_LPAd_ES11_InitiateAuthentication_Nominal where the LPAd plays the IPAe role

4.3.7.2.2 TC_IPAe_ES11_InitiateAuthentication_ErrorCases

Test Sequence #01 Error: Invalid SM-DS Address

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: Initiate Authentication* in section 4.4.27.2.2 TC_LPAd_ES11_InitiateAuthentication_ErrorCases where the LPAd plays the IPAe role

Test Sequence #02 Error: Unsupported Security Configuration

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Error: Unsupported Security Configuration* in section 4.4.27.2.2 TC_LPAd_ES11_InitiateAuthentication_ErrorCases where the LPAd plays the IPAe role

Test Sequence #03 Error: Unsupported SVN

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #03 Error: Unsupported SVN* in section 4.4.27.2.2 TC_LPAd_ES11_InitiateAuthentication_ErrorCases where the LPAd plays the IPAe role

Test Sequence #04 Error: Unavailable SM-DS Certificate

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #04 Error: Unavailable SM-DS Certificate* in section 4.4.27.2.2 TC_LPAd_ES11_InitiateAuthentication_ErrorCases where the LPAd plays the IPAe role

Test Sequence #05 Error: Invalid SM-DS Certificate

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #05 Error: Invalid SM-DS Certificate* in section 4.4.27.2.2 TC_LPAd_ES11_InitiateAuthentication_ErrorCases where the LPAd plays the IPAe role

Test Sequence #06 Error: Invalid SM-DS Signature

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #06 Error: Invalid SM-DS Signature* in section 4.4.27.2.2 TC_LPAd_ES11_InitiateAuthentication_ErrorCases where the LPAd plays the IPAe role

Test Sequence #07 Error: Invalid SM-DS Address sent by the SM-DS

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #07 Error: Invalid SM-DS Address sent by the SM-DS* in section 4.4.27.2.2 TC_LPAd_ES11_InitiateAuthentication_ErrorCases where the LPAd plays the IPAe role

Test Sequence #08 Error: Unsupported CI Key ID

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #08 Error: Unsupported CI Key ID* in section 4.4.27.2.2 TC_LPAd_ES11_InitiateAuthentication_ErrorCases where the LPAd plays the IPAe role

4.3.8 ES11 (IPAe – SM-DS): AuthenticateClient

4.3.8.1 Conformance Requirements

ES11 interface between the IPAe and SM-DS is identical to the LPAd and SM-DS.

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.3.8.2 Test Cases

4.3.8.2.1 TC_IPAe_ES11_AuthenticateClient_Nominal

Test Sequence #01 Nominal: Authenticate Client with empty MatchingID

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: Authenticate Client with empty MatchingID* in section 4.4.27.2.2 LPAd_ES11_AuthenticateClient_Nominal where the LPAd plays the IPAe role

Test Sequence #02 Nominal: Authenticate Client with MatchingID set to EventID

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: Authenticate Client with empty MatchingID* section 4.4.27.2.2 LPAd_ES11_AuthenticateClient_Nominal where the LPAd plays the IPAe role

4.3.8.2.2 TC_IPAe_ES11_AuthenticateClient_ErrorCases

Test Sequence #01 Error: Invalid EUM Certificate

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Error: Invalid EUM Certificate* section 4.4.28.2.2 TC_LPAd_ES11_AuthenticateClient_ErrorCases where the LPAd plays the IPAe role

Test Sequence #02 Error: Expired EUM Certificate

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Error: Expired EUM Certificate* section 4.4.28.2.2 TC_LPAd_ES11_AuthenticateClient_ErrorCases where the LPAd plays the IPAe role

Test Sequence #03 Error: Invalid eUICC Certificate

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #03 Error: Invalid eUICC Certificate* section 4.4.28.2.2 TC_LPAd_ES11_AuthenticateClient_ErrorCases where the LPAd plays the IPAe role

Test Sequence #04 Error: Expired eUICC Certificate

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #04 Error: Expired eUICC Certificate* section 4.4.28.2.2 TC_LPAd_ES11_AuthenticateClient_ErrorCases where the LPAd plays the IPAe role

Test Sequence #05 Error: Invalid eUICC signature or serverChallenge

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #04 Error: Expired eUICC Certificate* section 4.4.28.2.2 TC_LPAd_ES11_AuthenticateClient_ErrorCases where the LPAd plays the IPAe role

Test Sequence #06 Error: Unknown TransactionID

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #04 Error: Expired eUICC Certificate* section 4.4.28.2.2 TC_LPAd_ES11_AuthenticateClient_ErrorCases where the LPAd plays the IPAe role

Test Sequence #07 Error: Unknown Event Record

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #07 Error: Unknown Event Record* section 4.4.28.2.2 TC_LPAd_ES11_AuthenticateClient_ErrorCases where the LPAd plays the IPAe role

4.3.9 ES11 (IPAe -- SM-DS): HTTPS

4.3.9.1 Conformance Requirements

ES11 interface between the IPAe and SM-DS is identical to the LPAd and SM-DS.

References

GSMA RSP Technical Specification [4] and eSIM IoT Technical Specification [3]

4.3.9.2 Test Cases

4.3.9.2.1 TC_IPAe_ES11_HTTPS_Nominal

Test Sequence #01 Nominal: HTTPS Session Establishment

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Nominal: HTTPS Session Establishment* section 4.4.29.2.1 TC_LPAd_ES11_HTTPS_Nominal where the LPAd plays the IPAe role

Test Sequence #02 Nominal: non-reuse of session keys

The purpose of this test sequence is to verify that the IPAe is not reusing ephemeral keys from the previous session.

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Nominal: non-reuse of session keys* section 4.4.28.2.2 TC_LPAd_ES11_AuthenticateClient_ErrorCases where the LPAd plays the IPAe role

4.3.9.2.2 TC_IPAe_ES11_HTTPS_Error

Test Sequence #01 Error: Invalid (SM-DS) TLS Certificate signature

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #01 Error: Invalid (SM-DS) TLS Certificate signature* section 4.4.28.2.2 TC_LPAd_ES11_AuthenticateClient_ErrorCases where the LPAd plays the IPAe role

Test Sequence #02 Error: Expired TLS Certificate

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #02 Error: Expired TLS Certificate* section 4.4.28.2.2 TC_LPAd_ES11_AuthenticateClient_ErrorCases where the LPAd plays the IPAe role

Test Sequence #03 Error: VOID

Test Sequence #04 Error: VOID

Test Sequence #05 Error: VOID

Test Sequence #06 Error: VOID

Test Sequence #07 Error: Invalid TLS Certificate based on Invalid CI (Invalid Curve)

This test sequence is the same as defined in SGP.23 [8] - the *Test Sequence #07 Error: Invalid TLS Certificate based on Invalid CI (Invalid Curve)* section 4.4.28.2.2 TC_LPAd_ES11_AuthenticateClient_ErrorCases where the LPAd plays the IPAe role

5 Procedure - Behaviour Testing

5.1 General Overview

NOTE: Those test cases related to eUICC Behaviour are FFS.

7 External Test Specifications

Some test specifications related to the RSP ecosystem have been developed by external organisations (e.g. TCA (formerly SIMalliance)). These organisations defined their own requirements for test benches, test applicability and pass criteria.

This section lists the test specifications that relate to SGP.22 [4] and SGP.32 [3].

7.1 TCA eUICC Profile Package Test Specification

The TCA eUICC Profile Package: Interoperable Format Test Specification [23] SHALL be executed on the eUICC in order to check its compliance with the eUICC Profile Package Specification [4].

Test cases are applicable according to the eUICC Profile Package Specification [4] version and the additional eUICC Profile Package Specification [4] versions (if any) supported by the eUICC, in conjunction with the applicability table of the referred Test Specification [23].

The table below describes the versions of the eUICC Profile Package Specification [4] allowed depending on the SGP.22 and SGP.32 versions supported by the eUICC:

SGP.22 version	SGP.32 version	eUICC Profile Package Specification [4] versions required for the given SGP.22 version	Allowed values for #IUT_EUICC_ADD_PP_VERSIONS
2.5	1.0	2.1 or 2.2 or 2.3.1	3.1 or 3.2 or 3.3.1

Moreover, eUICC Manufacturers SHALL declare that the following options (as defined in [23]) are supported by the eUICC:

- O_MILENAGE
- O_TUAK_128
- O_JAVACARD

Annex A Constants

A.1 Generic Constants

Name	Content
AUTO_ENABLE_FLAG	NULL
EID1	-- as defined in SGP.23 [8]
EID2	-- as defined in SGP.23 [8]

EIM_CONFIG_DATA_NO_EIM_ID	EimConfiguraitonData ::= {
	  counterValue 1
	}

EIM_CONFIG_DATA1	EimConfigurationData ::= {
	  eimId #TEST_EIM_ID1,
	  eimIdType 3, -- proprietary
	  counterValue 1,
	  eimPublicKeyData #EIM_PUBLIC_KEY_DATA_PK
	}

EIM_CONFIG_DATA1_UPDATE_COUNTER	EimConfiguraitonData ::= {
	  eimId #TEST_EIM_ID1,
	  counterValue 99
	}

EIM_CONFIG_DATA1_UPDATE_COUNTER_OUT_OF_RANGE	EimConfigurationData ::= {
	  eimId #TEST_EIM_ID1,
	  counterValue #EIM_COUNTER_OUT_OF_RANGE
	}

EIM_CONFIG_DATA1_UPDATE_LOWER_COUNTER	EimConfigurationData ::= {
	  eimId #TEST_EIM_ID1,
	  counterValue 0
	}

EIM_CONFIG_DATA1_UPDATE_UKNOWN_EUICC_CI_PKID	EimConfigurationData ::= {
	  eimId #TEST_EIM_ID1,
	  euiccCiPKId #CI_PKI_RANDOM
	}

EIM_CONFIG_DATA2	EimConfigurationData ::= {
	  eimId #TEST_EIM_ID2,
	  eimIdType 3, -- proprietary
	  counterValue 1,
	  eimPublicKeyData #EIM_PUBLIC_KEY_DATA_PK
	}

Name	Content

EIM_CONFIG_DATA2_ADD_COUNTER_OUT_OF_RANGE	EimConfigurationData ::= {
	  eimId #TEST_EIM_ID2,
	  eimIdType 3, -- proprietary
	  counterValue #EIM_COUNTER_OUT_OF_RANGE
	  eimPublicKeyData #EIM_PUBLIC_KEY_DATA_PK
	}

EIM_CONFIG_DATA2_ADD_UKNOWN_EUICC_CI_PKID	EimConfigurationData ::= {
	  eimId #TEST_EIM_ID2,
	  eimIdType 3, -- proprietary
	  counterValue 1,
	  eimPublicKeyData #EIM_PUBLIC_KEY_DATA_PK
	  euiccCiPKId #CI_PKI_RANDOM
	}

EIM_CONFIG_DATA2_UPDATE_COUNTER	EimConfiguraitonData ::= {
	  eimId #TEST_EIM_ID2,
	  counterValue 99
	}

EIM_CONFIG_DATA2_EIM_ID_129_CHARACTERS	EimConfiguraitonData ::= {
	  eimId #TEST_EIM_ID129,
	  counterValue 1
	}

EIM_COUNTER_OUT_OF_RANGE	8388608 (0x800000)
EIM_PUBLIC_KEY_DATA_PK	eimPublicKey #PK_S_EIMsign_ECDSA
ICCID_OP_PROF1	-- as defined in SGP.23 [8]
ICCID_OP_PROF2	-- as defined in SGP.23 [8]
ICCID_OP_PROF3	-- as defined in SGP.23 [8]

Name	Content
ICCID_OP_PROFX	`-- as defined in SGP.23 [8]`
ICON_OP_PROF1	`-- as defined in SGP.23 [8]`
ICON_OP_PROF2	`-- as defined in SGP.23 [8]`
ICON_OP_PROF3	`-- as defined in SGP.23 [8]`

LIST_EIM_RESULT	`{`
	`  {eimId #TEST_EIM_ID1},`
	`  {eimId #TEST_EIM_ID2}`
	`}`

NAME_OP_PROF1	`-- as defined in SGP.23 [8]`
NAME_OP_PROF2	`-- as defined in SGP.23 [8]`
NAME_OP_PROF3	`-- as defined in SGP.23 [8]`
PROFILE_TAG_INVALID	`0xAA`
SP_NAME1	`-- as defined in SGP.23 [8]`
SP_NAME2	`-- as defined in SGP.23 [8]`
SP_NAME3	`-- as defined in SGP.23 [8]`
S_SM_DP+_OID1	`-- as defined in SGP.23 [8]`
S_SM_DP+_OID2	`-- as defined in SGP.23 [8]`
SMDP_ADDRESS1	`0x 73 6D 64 70 33 33 2E 67 73 6D 61 2E 63 6F 6D`
TAGS_AID_PROFILENAME	`-- as defined in SGP.23 [8]`
TEST_EIM_ID1	`testeim1`
TEST_EIM_ID2	`testeim2`
TEST_EIM_ID129 (129 characters)	`testeim1234567890abcdefghijklmnopqrstuvwxyz1234567890abcdefghijklmnopqrstuvwxyz1234567890abcdefghijklmnopqrstuvwxyz1234567890abcd`

A.2 Test Certificates and Test Keys

All ECC certificates and keys described below are based on either:

- NIST P-256 curve, defined in Digital Signature Standard [11]

- brainpoolP256r1 curve, defined in RFC 5639 [8]

NOTE: SGP.26 [25] contains test keys, valid test certificates and instructions for how to generate invalid certificates. Unless specified differently, the test keys and test certificates used in the present document are bundled with SGP.26 [25].

Name	Description
CERT_EUICC_SIG	-- as CERT_EUICC_ECDSA defined in SGP.23 [8].
CERT_EUM_SIG	-- as CERT_EUM_ECDSA defined in SGP.23 [8]
CERT_S_EIMsign_ ECDSA	Certificate of S EIM
PK_EUICC_SIG	-- as PK_EUICC_ECDSA defined in SGP.23 [8].
PK_EUM_SIG	Public Key of the eUICC, contained within #CERT_EUM_SIG.
PK_S_EIMsign_ECDSA	Public Key of S_EIM contained within #CERT_S_EIMsign_ ECDSA

Annex B Dynamic Content

1)

Variable	Description
CI_KEY_ID_BP256	Subject Key Identifier of the PK CI Key ID based on BrainpoolP256r1 curve
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_CI_PK_ID_TO_BE_USED	-- as defined in SGP.23 [XX]
EUICC_SIGN_EPR	EuiccPackageResult signature computed as defined in SGP.32 [X].
ISD_P_AID1	-- as defined in SGP.23 [8]
ISD_P_AID2	-- as defined in SGP.23 [8]
ISD_P_AID3	-- as defined in SGP.23 [8]
S_COUNTER_VALUE	Counter used by the eIM for replay protection
S_EIM_SIGNATURE	Signature computed as defined in SGP.32 [X].
S_EIM_SIGN_RANDOM	Random eIM signature with a size corresponding to a valid one as defined in SGP.32 [X].
S_LOW_CNTR_VALUE	Any value lower than <S_COUNTER_VALUE>
S_TRANSACTION_ID	The transactionId which allows the eIM to link a received eUICC Package Result to a sent eUICC Package.
SEQ_NUMBER	The sequnce number used to return either a Notification or an eUICC Package Result

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_ADD_INITIAL_EIM_SINGLE
Description	Generate the ASN.1 AddInitialEimRequest structure with a single EIM.
Parameter(s)	• paramAddInitialEimData: list of eimConfigurationData data

Details	`AddInitialEimRequest ::= {`
	`    eimConfigurationDataList {`
	`        paramAddInitialEimData`
	`    }`

Method	MTD_EUICC_PACKAGE_REQUEST_DISABLE
Description	Generate the ASN.1 EuiccPackageRequest structure according to the input parameters.
Parameter(s)	• paramEIMId : identifier of the eIM that issues the eUICC Package. • paramEidValue: value of EID of the targeted eUICC • paramCounterValue: used by the eIM for replay protection • paramTransactionId: optional, used to uniquely identify the RSP session • paramIccidValue: The ICCID of the Profile to Enable. • paramEimSignature: Signature computed as defined in SGP.32 [X].

Details	IF TransactionId is provided Then
	req EuiccPackageRequest ::= {
	  euiccPackageSigned {
	    eimId paramEIMId,
	    eidValue paramEidValue,
	    counterValue paramCounterValue,
	    transactionId paramTransactionId,
	    euiccPackage psmoList : {
	      disable : {
	        iccid paramIccidValue
	      }
	    }
	  },
	  eimSignature paramEimSignature
	}
	End if

	IF TransactionId is not provided Then
	req EuiccPackageRequest ::= {
	  euiccPackageSigned {
	    eimId paramEIMId,
	    eidValue paramEidValue,
	    counterValue paramCounterValue,
	    euiccPackage psmoList : {
	      disable : {
	        iccid paramIccidValue
	      }
	    }
	  },
	  eimSignature paramEimSignature
	}
	 End if

Method	MTD_CONFIGURE_AUTOMATIC_PROFILE_ENABLING
Description	Generate the ASN.1 ConfigureAutomaticProfileEnabling structure according to the input parameters.
Parameter(s)	• autoEnableFlag: optional, this flag indicates whether the automatic enable should be active or not. • paramSmdpOid: optional, value of smdpOid for the default SM-DP+ • paramSmdpAddress: optional, value of smdpAddress for the default SM-DP+

Details	IF autoEnableFlag, paramSmdpOid and paramSmdpAddress are provided then
	
	req configureAutoEnable : {
	    autoEnableFlag NULL,
	    smdpOid paramSmdpOid,
	    smdpAddress paramSmdpAddress
	}
	 End if
	
	IF autoEnableFlag is not present, paramSmdpOid and paramSmdpAddress are provided then
	req configureAutoEnable : {
	    smdpOid paramSmdpOid,
	    smdpAddress paramSmdpAddress
	}
	End if

	MTD_ENABLE_USING_DD
Description	Generate the ASN.1 EnableUsingDDRequest structure.
Parameter(s)	no parameters

Details	req EnableUsingDDRequest ::= {
	}

Method	MTD_EUICC_PACKAGE_REQUEST_ADD_EIM
Description	Generate the ASN.1 EuiccPackageRequest structure according to the input parameters.
Parameter(s)	• paramEIMId : identifier of the eIM that issues the eUICC Package • paramEidValue: value of EID of the targeted eUICC • paramCounterValue: used by the eIM for replay protection • paramEimTransactionId: optional, used by the eIM to uniquely identify an RSP session • paramEimConfigData: eIM Configuration Data to be added • paramEimSignature: Signature computed as defined in SGP.32 [X]

Details	IF paramEimTransactionId is provided THEN
	  req EuiccPackageRequest ::= {
	    euiccPackageSigned {
	      eimId paramEIMId,
	      eidValue paramEidValue,
	      counterValue paramCounterValue,
	      eimTransactionId paramEimTransactionId,
	      euiccPackage ecoList : {
	        addEim paramEimConfigData
	      }
	    },
	    eimSignature paramEimSignature
	  }
	ELSE THEN
	  req EuiccPackageRequest ::= {
	    euiccPackageSigned {
	      eimId paramEIMId,
	      eidValue paramEidValue,
	      counterValue paramCounterValue,
	      euiccPackage ecoList : {
	        addEim paramEimConfigData
	      }
	    },
	    eimSignature paramEimSignature
	  }
	END IF

Method	MTD_EUICC_PACKAGE_GET_RAT
Description	Generate the ASN.1 EuiccPackageRequest structure according to the input parameters.
Parameter(s)	• paramEIMId : identifier of the eIM that issues the eUICC Package. • paramEidValue: value of EID of the targeted eUICC • paramCounterValue: used by the eIM for replay protection

Details	req EuiccPackageRequest ::= {
	euiccPackageSigned {
	eimId paramEIMId,
	eidValue paramEidValue,
	counterValue paramCounterValue,
	euiccPackage psmoList : {
	getRAT : {
	}
	}
	},
	eimSignature paramEimSignature
	} End if

Method	MTD_EUICC_PACKAGE_REQUEST_DELETE_EIM
Description	Generate the ASN.1 EuiccPackageRequest structure according to the input parameters.
Parameter(s)	• paramEIMId : identifier of the eIM that issues the eUICC Package. • paramEidValue: value of EID of the targeted eUICC • paramCounterValue: used by the eIM for replay protection • paramEimTransactionId: optional, used by the eIM to uniquely identify an RSP session • paramTargetEIMId: identifier of the eIM of which eIM Configuration Data is to be deleted • paramEimSignature: Signature computed as defined in SGP.32 [X].

Details	IF paramEimTransactionId is provided THEN
	req EuiccPackageRequest ::= {
	euiccPackageSigned {
	eimId paramEIMId,
	eidValue paramEidValue,
	counterValue paramCounterValue,
	eimTransactionId paramEimTransactionId,
	euiccPackage ecoList : {
	deleteEim paramTargetEIMId
	}
	},
	eimSignature paramEimSignature
	}

	ELSE THEN
	  req EuiccPackageRequest ::= {
	    euiccPackageSigned {
	      eimId paramEIMId,
	      eidValue paramEidValue,
	      counterValue paramCounterValue,
	      euiccPackage ecoList : {
	        deleteEim paramTargetEIMId
	      }
	    },
	    eimSignature paramEimSignature
	  }
	END IF

Method	MTD_EUICC_PACKAGE_REQUEST_LIST_EIM
Description	Generate the ASN.1 EuiccPackageRequest structure according to the input parameters.
Parameter(s)	• paramEIMId : identifier of the eIM that issues the eUICC Package. • paramEidValue: value of EID of the targeted eUICC • paramCounterValue: used by the eIM for replay protection • paramEimTransactionId: optional, used by the eIM to uniquely identify an RSP session • paramEimSignature: Signature computed as defined in SGP.32 [X].

Details	IF paramEimTransactionId is provided THEN
	  req EuiccPackageRequest ::= {
	    euiccPackageSigned {
	      eimId paramEIMId,
	      eidValue paramEidValue,
	      counterValue paramCounterValue,
	      eimTransactionId paramEimTransactionId,
	      euiccPackage ecoList : {
	        listEim { }
	      }
	    },
	    eimSignature paramEimSignature
	  }
	ELSE THEN
	  req EuiccPackageRequest ::= {
	    euiccPackageSigned {
	      eimId paramEIMId,
	      eidValue paramEidValue,
	      counterValue paramCounterValue,
	      euiccPackage ecoList : {
	        listEim : { }
	      }
	    },
	    eimSignature paramEimSignature

	}
	END IF

Method	MTD_EUICC_PACKAGE_REQUEST_LIST_PRO_FILE_INFO
Description	Generate the ASN.1 EuiccPackageRequest structure according to the input parameters.
Parameter(s)	• paramEIMId: identifier of the eIM that issues the eUICC Package. • paramEidValue: value of EID of the targeted eUICC • paramCounterValue: used by the eIM for replay protection • paramIccidValue: optional ICCID of target Profile. • paramIsdpAid: optional AID of the ISD-P • paramProfileClass: optional Profile Class • paramTagList: optional tagList • paramEimSignature: Signature computed as defined in SGP.32 [X].

Details	IF all optional parameters are not provided Then
	
	req EuiccPackageRequest ::= {
	  euiccPackageSigned {
	    eimId paramEIMId ,
	    eidValue paramEidValue,
	    counterValue paramCounterValue,
	    euiccPackage psmoList : {
	      listProfileInfo : {
	      }
	    }
	  },
	  eimSignature paramEimSignature
	} End if
	
	IF paramIsdpAid is provided and the other parameters are not providedThen
	
	req EuiccPackageRequest ::= {
	  euiccPackageSigned {
	    eimId paramEIMId ,
	    eidValue paramEidValue,
	    counterValue paramCounterValue,
	    euiccPackage psmoList : {
	      listProfileInfo : {
	        searchCriteria isdpAid : paramIsdpAid
	      }
	    }
	  },
	  eimSignature paramEimSignature
	} End if

	IF paramIccid is provided and the other parameters are not providedThen
	
	req EuiccPackageRequest ::= {
	  euiccPackageSigned {
	    eimId paramEIMId ,
	    eidValue paramEidValue,
	    counterValue paramCounterValue,
	    euiccPackage psmoList : {
	      listProfileInfo : {
	        searchCriteria iccid : paramIccidValue
	      }
	    }
	  },
	  eimSignature paramEimSignature
	} End if
	
	IF paramProfileClass is provided and the other parameters are not providedThen
	
	req EuiccPackageRequest ::= {
	  euiccPackageSigned {
	    eimId paramEIMId,
	    eidValue paramEidValue,
	    counterValue paramCounterValue,
	    euiccPackage psmoList : {
	      listProfileInfo : {
	        searchCriteria profileClass: paramProfileClass
	      }
	    }
	  },
	  eimSignature paramEimSignature
	} End if
	
	IF tagList is provided and the other parameters are not providedThen
	
	req EuiccPackageRequest ::= {
	  euiccPackageSigned {
	    eimId paramEIMId,
	    eidValue paramEidValue,
	    counterValue paramCounterValue,
	    euiccPackage psmoList : {
	      listProfileInfo : {
	        tagList tagList
	      }
	    }
	  },
	  eimSignature paramEimSignature

	} End if

Method	MTD LIST_PROFILE_INFO CHANGE_ONGOING_ERROR
Description	Generate the ASN.1 EuiccPackageRequest structure according to the input parameters.
Parameter(s)	• paramEIMId : identifier of the eIM that issues the eUICC Package. • paramEidValue: value of EID of the targeted eUICC • paramCounterValue: used by the eIM for replay protection • paramEimSignature: Signature computed as defined in SGP.32 [X].

Details	req EuiccPackageRequest ::= {
	euiccPackageSigned {
	eimId paramEIMId,
	eidValue paramEidValue,
	counterValue paramCounterValue,
	euiccPackage psmoList : {
	enable : {
	iccid ICCID_OP_PROF2
	},
	listProfileInfo : {
	searchCriteria iccid : ICCID_OP_PROF2
	}
	}
	},
	eimSignature paramEimSignature
	} End if

Method	MTD_EUICC_PACKAGE_REQUEST_UPDATE_EIM
Description	Generate the ASN.1 EuiccPackageRequest structure according to the input parameters.
Parameter(s)	• paramEIMId : identifier of the eIM that issues the eUICC Package. • paramEidValue: value of EID of the targeted eUICC • paramCounterValue: used by the eIM for replay protection • paramEimTransactionId: optional, used by the eIM to uniquely identify an RSP session • paramEimConfigData: eIM Configuration Data to be updated • paramEimSignature: Signature computed as defined in SGP.32 [X].

Details	IF paramEimTransactionId is provided THEN
	`req EuiccPackageRequest ::= {`
	`euiccPackageSigned {`
	`eimId paramEIMId,`
	`eidValue paramEidValue,`
	`counterValue paramCounterValue,`
	`eimTransactionId paramEimTransactionId,`

	euiccPackage ecoList : {
	updateEim paramEimConfigData
	}
	},
	eimSignature paramEimSignature
	}
	ELSE THEN
	req EuiccPackageRequest ::= {
	euiccPackageSigned {
	eimId paramEIMId,
	eidValue paramEidValue,
	counterValue paramCounterValue,
	euiccPackage ecoList : {
	updateEim : paramEimConfigData
	}
	},
	eimSignature paramEimSignature
	}
	END IF

Method	MTD_EUICC_PACKAGE_REQUEST_ENABLE
Description	Generate the ASN.1 EuiccPackageRequest structure according to the input parameters.
Parameter(s)	• paramEIMId : identifier of the eIM that issues the eUICC Package. • paramEidValue: value of EID of the targeted eUICC • paramCounterValue: used by the eIM for replay protection • paramTransactionId: optional, used to uniquely identify the RSP session • paramIccidValue: The ICCID of the Profile to Enable. • paramRollbackFlag: optional rollback condition • paramEimSignature: Signature computed as defined in SGP.32 [X].

Details	IF TransactionId and rollbackFlag are provided Then
	
	req EuiccPackageRequest ::= {
	euiccPackageSigned {
	eimId paramEIMId ,
	eidValue paramEidValue,
	counterValue paramCounterValue,
	transactionId paramTransactionId,
	euiccPackage psmoList : {
	enable : {
	iccid paramIccidValue,
	rollbackFlag paramRollbackFlag
	}

```
    }
  },
  eimSignature paramEimSignature
}
 End if

IF TransactionId is provided and rollbackFlag is not provided Then

req EuiccPackageRequest ::= {
  euiccPackageSigned {
    eimId paramEIMId  ,
    eidValue paramEidValue,
    counterValue paramCounterValue,
    transactionId paramTransactionId,
    euiccPackage psmoList : {
      enable : {
        iccid paramIccidValue,
      }
    }
  },
  eimSignature paramEimSignature
}
 End if

IF TransactionId is not provided and rollbackFlag  is provided Then

req EuiccPackageRequest ::= {
  euiccPackageSigned {
    eimId paramEIMId  ,
    eidValue paramEidValue,
    counterValue paramCounterValue,
    euiccPackage psmoList : {
      enable : {
        iccid paramIccidValue,
        rollbackFlag paramRollbackFlag
      }
    }
  },
  eimSignature paramEimSignature
} End if

IF TransactionId and rollbackFlag are not provided Then

req EuiccPackageRequest ::= {
  euiccPackageSigned {
    eimId paramEIMId  ,
```

	eidValue paramEidValue,
	counterValue paramCounterValue,
	euiccPackage psmoList : {
	enable : {
	iccid paramIccidValue,
	}
	}
	},
	eimSignature paramEimSignature
	} End if

Method	MTD_EUICC_PACKAGE_REQUEST_DELETE
Description	Generate the ASN.1 EuiccPackageRequest structure according to the input parameters for the delete.
Parameter(s)	• paramEIMId : identifier of the eIM that issues the eUICC Package. • paramEidValue: value of EID of the targeted eUICC • paramCounterValue: used by the eIM for replay protection • paramTransactionId: optional, used to uniquely identify the RSP session • paramIccidValue: The ICCID of the Profile to Enable. paramEimSignature: Signature computed as defined in SGP.32 [X].

Details	IF TransactionId is provided
	
	req EuiccPackageRequest ::= {
	euiccPackageSigned {
	eimId paramEIMId ,
	eidValue paramEidValue,
	counterValue paramCounterValue,
	euiccPackage psmoList : {
	delete : {
	iccid paramIccidValue
	}
	}
	},
	eimSignature paramEimSignature
	}
	Else
	req EuiccPackageRequest ::= {
	euiccPackageSigned {
	eimId paramEIMId ,
	eidValue paramEidValue,
	counterValue paramCounterValue,
	transactionId paramTransactionId,
	euiccPackage psmoList : {
	delete : {
	iccid paramIccidValue

	}
	   }
	  },
	  eimSignature paramEimSignature
	}
	End If

Method	MTD_EUICC_PKG_ERROR_UNSIGNED
Description	Generate the ASN.1 EuiccPackageResult structure for euiccPackageErrorUnsigned
Parameter(s)	- paramEIMId: eIMId to be returned

Details	`resp EuiccPackageResult ::=`
	`euiccPackageErrorUnsigned : {`
	`  eimId [0] paramEIMId`
	`}`

Method	MTD_EUICC_PKG_ ERROR_SIGNED
Description	Generate the ASN.1 EuiccPackageResult structure euiccPackageErrorSigned.
Parameter(s)	- paramEuiccPackageResultErrorCode: The Package Result Error code (Mandatory)

Details	`resp EuiccPackageResult ::=`
	`euiccPackageErrorSigned : {`
	`  euiccPackageErrorDataSigned {`
	`    eimId [0] <S_EIM_ID>,`
	`    counterValue [1] <S_COUNTER_VALUE>,`
	`    transactionId[2] <S_TRANSACTION_ID>,`
	`    euiccPackageErrorCode paramEuiccPackageResultErrorCode`
	`  },`
	`  euiccSignEPR <EUICC_SIGNATURE>`
	`}`

Method	MTD_GET_CERTS_REQUEST
Description	Generate the ASN.1 GetCerts structure
Parameter(s)	- paramEuiccCiPkId: Id of the targeted eUICC CI PKI (Optional)

Details	`  GetCertsRequest ::= {`
	`       euiccCiPKId paramEuiccCiPkiId`
	`       }`

Method	MTD_GET_EIM
Description	Generate the ASN.1 GetEimRequest structure.
Parameter(s)	-
Details	`  GetEimConfigurationDataRequest ::= {`

	}

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
	req ProfileInfoListRequest::= {
	searchCriteria isdpAid: paramIsdpAidValue
	}
	End If

Method	MTD_PROFILE_ROLLBACK_REQUEST
Description	Generate the ASN.1 ProfileRollbackRequest according to the input parameters.
Parameter(s)	• paramRefreshFlag: indicating whether REFRESH is required

Details	ProfileRollbackRequest ::= {
	refreshFlag paramRefreshFlag
	}

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_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_STORE_DATA
Description	Generates the STORE DATA command (Case 4) as defined in GlobalPlatform Card Specification [X].
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

C.2 Procedures

	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_OPEN_LOGICAL_CHANNEL_AND_SELECT_ISDR	
	Description	The LPAd opens a logical channel and selects the ISD-R.	
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_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

Annex D Commands And Responses

D.1 ES10x Requests And Responses

D.1.1 ES10x Requests

Name	Content
GET_PROFILES_INFO_ALL	`request ProfileInfoListRequest ::= { }`

LOAD_EUICC_PKG_ INVALID_ECO	`{`
	`  ecoList : {`
	`    update_Eim,`
	`    #EIM_CONFIG_DATA1_UPDATE_COUNTER`
	`  }`
	`}`

LOAD_EUICC_PKG_ INVALID_PSMO	`{`
	`  psmoList : {`
	`      remove : {`
	`          iccid #ICCID_OP_PROF1`
	`      }`
	`  }`
	`}`

D.1.2 ES10x Responses

Name	Content

ADD_INITIAL_EIM_RES_OK	`addInitialEimOk {`
	`        addOk`
	`}`

ADD_INITIAL_EIM_ ERR_EIM_CONF	`addInitialEimError {`
	`        unsignedEimConfigDisallowed`
	`}`

ADD_INITIAL_EIM_ERR_UNKNOWN_CI	`addInitialEimError {`
	`        ciPKUnknown`
	`}`

ADD_INITIAL_EIM_ERR_CNTR_OUT_OF_RANGE	`addInitialEimError {`
	`        counterValueOutOfRange`
	`}`

CONFIG_AUTO_ENABLE_RES_OK	`response ConfigureAutoProfileEnablingResponse ::=`
	`{`

	`  configAutoEnableResult 0`
	`}`

CONFIG_AUTO_ENABLE_RES_CONFIG_DISALLOWED	`response ConfigureAutoProfileEnablingResponse ::= {`
	`  configAutoEnableResult 2`
	`}`

DELETE_PROFILE_ICCID_NOT_FOUND	`  euiccPackageResultDataSigned {`
	`    eimId paramEIMId,`
	`    counterValue paramCounterValue,`
	`    seqNumber SEQ_NUMBER,`
	`    euiccResult {`
	`      deleteResult : 1`
	`    }`
	`  },`
	`  euiccSignEPR paramEimSignature`

DELETE_PROFILE_ICCID_NOT_IN_DISABLE_STATE	`  euiccPackageResultDataSigned {`
	`    eimId paramEIMId,`
	`    counterValue paramCounterValue,`
	`    seqNumber SEQ_NUMBER,`
	`    euiccResult {`
	`      deleteResult : 2`
	`    }`
	`  },`
	`  euiccSignEPR paramEimSignature`

DELETE_RES_OK_1	`  euiccPackageResultDataSigned {`
	`    eimId paramEIMId,`
	`    counterValue paramCounterValue,`
	`    seqNumber SEQ_NUMBER,`
	`    euiccResult {`
	`      deleteResult: 0`
	`    }`
	`  },`
	`  euiccSignEPR paramEimSignature`

DELETE_RES_OK_2	`euiccPackageResultDataSigned {`
	`    eimId paramEIMId,`
	`    counterValue paramCounterValue,`
	`    transactionId S_TRANSACTION_ID,`
	`    seqNumber SEQ_NUMBER,`
	`    euiccResult {`
	`      deleteResult: 0`
	`    }`
	`  },`
	`  euiccSignEPR paramEimSignature`

DISABLE_RES_OK_1	euiccPackageResultDataSigned {
	eimId paramEIMId,
	counterValue paramCounterValue,
	seqNumber SEQ_NUMBER,
	euiccResult {
	disableResult : 0
	}
	},
	euiccSignEPR EUICC_SIGN_EPR

DISABLE_RES_OK_2	euiccPackageResultDataSigned {
	eimId paramEIMId,
	counterValue paramCounterValue,
	transactionId paramTransactionId
	seqNumber SEQ_NUMBER,
	euiccResult {
	disableResult : 0
	}
	},
	euiccSignEPR EUICC_SIGN_EPR

DISABLE_RES_PROFILE_ICCID_NOT_FOUND	euiccPackageResultDataSigned {
	eimId paramEIMId,
	counterValue paramCounterValue,
	seqNumber SEQ_NUMBER,
	euiccResult {
	disableResult : 1
	}
	},
	euiccSignEPR EUICC_SIGN_EPR

DISABLE_RES_PROFILE_NOT_IN_ENABLED_STATE	euiccPackageResultDataSigned {
	eimId paramEIMId,
	counterValue paramCounterValue,
	seqNumber SEQ_NUMBER,
	euiccResult {
	disableResult : 2
	}
	},
	euiccSignEPR EUICC_SIGN_EPR

ENABLE_RES_OK_1	euiccPackageResultDataSigned {
	eimId paramEIMId,
	counterValue paramCounterValue,
	seqNumber SEQ_NUMBER,
	euiccResult {
	enableResult : 0
	}
	},
	euiccSignEPR EUICC_SIGN_EPR

ENABLE_RES_OK_2	euiccPackageResultDataSigned {
	eimId paramEIMId,
	counterValue paramCounterValue,
	transactionId S_TRANSACTION_ID,
	seqNumber SEQ_NUMBER,
	euiccResult {
	enableResult : 0
	}
	},
	euiccSignEPR EUICC_SIGN_EPR

ENABLE_RES_PROFILE_ICCID_NOT_FOUND	euiccPackageResultDataSigned {
	eimId paramEIMId,
	counterValue paramCounterValue,
	seqNumber SEQ_NUMBER,
	euiccResult {
	enableResult : 1
	}
	},
	euiccSignEPR EUICC_SIGN_EPR

ENABLE_RES_PROFILE_NOT_IN_DISABLE_STATE	euiccPackageResultDataSigned {
	eimId paramEIMId,
	counterValue paramCounterValue,
	seqNumber SEQ_NUMBER,
	euiccResult {
	enableResult : 2
	}
	},
	euiccSignEPR EUICC_SIGN_EPR

ENABLE_RES_PROFILE_ UNDEFINED_ERROR	euiccPackageResultDataSigned {
	eimId paramEIMId,
	counterValue paramCounterValue,
	seqNumber SEQ_NUMBER,
	euiccResult {
	enableResult : 127
	}
	},
	euiccSignEPR EUICC_SIGN_EPR

ENABLE_USING_DD_RESULT_OK	enableUsingDDResult 0
ENABLE_USING_DD_RESULT_AUTO_ENABLE_NOT_AVAILABLE	enableUsingDDResult 1
ENABLE_USING_DD_RESULT_NO_SESSION_CONTEXT	enableUsingDDResult 4

GET_CERTS_RES_OK	certs {
	eumCertificate #CERT_EUM_SIG,
	euiccCertificate #CERT_EUICC_SIG
	}

GET_CERTS_ERR_UNKNOWN_CI	getCertsError {

	`invalidCiPKId`
	`}`

GET_EIM_CONFIG_DATA_LIST_EMPTY	`eimConfigurationDataList {}`

GET_EIM_CONFIG_DATA_LIST1	`eimConfigurationDataList {`
	`{`
	`eimId #TEST_EIM_ID1,`
	`eimIdType 3, -- proprietary`
	`eimSupportedProtocol 1, --`
	`proprietary`
	`eimPublicKeyData`
	`#EIM_PUBLIC_KEY_DATA_PK`
	`}`
	}

GET_EIM_CONFIG_DATA_LIST2	`eimConfigurationDataList {`
	`{`
	`eimId #TEST_EIM_ID1,`
	`eimIdType 3, -- proprietary`
	`eimSupportedProtocol 1, --`
	`proprietary`
	`eimPublicKeyData`
	`#EIM_PUBLIC_KEY_DATA_PK`
	`},`
	`{`
	`eimId #TEST_EIM_ID2,`
	`eimIdType 3, -- proprietary`
	`eimSupportedProtocol 1, --`
	`proprietary`
	`eimPublicKeyData`
	`#EIM_PUBLIC_KEY_DATA_PK`
	`}`
	}

LOAD_EUICC_PKG_ ERR_EID	`{ invalidEid }`
LOAD_EUICC_PKG_ ERR_LOW_CNTR	`{ counterValueOutOfRange }`

LOAD_EUICC_PKG_ ERR_ UNKNOWN_CMD	`euiccResult : {`
	`processingTerminated {`
	`unknownOrDamagedCommand`
	`}`
	`}`

PACKAGE_RES_ERROR_CODE_REPLAY_ERROR	`euiccPackageErrorDataSigned {`
	`eimId paramEIMId,`
	`counterValue paramCounterValue,`
	`euiccPackageErrorCode 4`
	`},`
	`euiccSignEPE EUICC_SIGN_EPR`

PPRS_ALLOWED (ProfilePolicyAuthorisationRule)	`-- as defined in SGP.23[XX]`

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_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_INFO2 (ProfileInfo)	-- as defined in SGP.23 [XX]
PROFILE_INFO3 (ProfileInfo)	-- as defined in SGP.23 [XX]

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_ROLLBACK_RES_OK	cmdResult : 0,
	eUICCPackageResult {
	euiccPackageResultDataSigned {

	eimId paramEIMId,
	counterValue paramCounterValue,
	seqNumber SEQ_NUMBER,
	euiccResult {
	rollbackResult : 0
	}
	},
	euiccSignEPR EUICC_SIGN_EPR
	}

PROFILE_ROLLBACK_RES_ROLLBACK_NOT_ALLOWED	cmdResult : 1
PROFILE_ROLLBACK_RES_COMMAND_ERROR	cmdResult : 7

PROFILE_INFO_TAG_LIST	listProfileInfoResult :
	profileInfoListOk : {
	{
	isdpAid ISD_P_AID1,
	profileName NAME_OP_PROF1
	},
	{
	isdpAid ISD_P_AID2,
	profileName NAME_OP_PROF2
	}
	{
	isdpAid ISD_P_AID3,
	profileName NAME_OP_PROF3
	}
	}

D.2 APDU

D.2.1 APDU Commands

Name	Content
GET_MNO_SD	- CLA = 80, INS = F2, P1 = 80, P2 = 02, LC = <L> - Data = 4F 00 - LE = 00
MANAGE_CHANNEL_OPEN	- CLA = 00, INS = 70, P1 = 00, P2 = 00, LE = 01
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_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

D.2.2 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.3 ESep Requests And Responses

D.3.1 ESep Responses

Name	Content

ADD_EIM_RES_ERR_CI_PK_UNKNOWN	`euiccPackageResultDataSigned {`
	`  eimId paramEIMId,`
	`  counterValue paramCounterValue,`
	`  seqNumber SEQ_NUMBER,`
	`  euiccResult : {`
	`    addEimResult : 3`
	`  }`
	`},`
	`euiccSignEPR EUICC_SIGN_EPR`

ADD_EIM_RES_ERR_COMMAND_ERROR	`euiccPackageResultDataSigned {`
	`  eimId paramEIMId,`
	`  counterValue paramCounterValue,`
	`  seqNumber SEQ_NUMBER,`
	`  euiccResult : {`
	`    addEimResult : 7`
	`  }`
	`},`
	`euiccSignEPR EUICC_SIGN_EPR`

ADD_EIM_RES_ERR_COUNTER_OUT_OF_RANGE	`euiccPackageResultDataSigned {`
	`  eimId paramEIMId,`
	`  counterValue paramCounterValue,`
	`  seqNumber SEQ_NUMBER,`
	`  euiccResult : {`
	`    addEimResult : 6`
	`  }`
	`},`
	`euiccSignEPR EUICC_SIGN_EPR`

ADD_EIM_RES_OK_1	`euiccPackageResultDataSigned {`
	`  eimId paramEIMId,`
	`  counterValue paramCounterValue,`
	`  seqNumber SEQ_NUMBER,`
	`  euiccResult : {`
	`    addEimResult : 0`
	`  }`
	`},`
	`euiccSignEPR EUICC_SIGN_EPR`

ADD_EIM_RES_OK_2	`euiccPackageResultDataSigned {`
	`  eimId paramEIMId,`
	`  counterValue paramCounterValue,`
	`  eimEimTransactionId S_EIM_TRANSACTION_ID,`
	`  seqNumber SEQ_NUMBER,`
	`  euiccResult : {`
	`    addEimResult : 0`
	`  }`
	`},`
	`euiccSignEPR EUICC_SIGN_EPR`

DELETE_EIM_RES_ERR_COMMAND_ERROR	euiccPackageResultDataSigned {
	  eimId paramEIMId,
	  counterValue paramCounterValue,
	  seqNumber SEQ_NUMBER,
	  euiccResult : {
	    deleteEimResult : 7
	  }
	},
	euiccSignEPR EUICC_SIGN_EPR

DELETE_EIM_RES_ERR_EIM_NOT_FOUND	euiccPackageResultDataSigned {
	  eimId paramEIMId,
	  counterValue paramCounterValue,
	  seqNumber SEQ_NUMBER,
	  euiccResult : {
	    deleteEimResult : 1
	  }
	},
	euiccSignEPR EUICC_SIGN_EPR

DELETE_EIM_RES_ERR_LAST_EIM_DELETED	euiccPackageResultDataSigned {
	  eimId paramEIMId,
	  counterValue paramCounterValue,
	  seqNumber SEQ_NUMBER,
	  euiccResult : {
	    deleteEimResult : 2
	  }
	},
	euiccSignEPR EUICC_SIGN_EPR

DELETE_EIM_RES_OK_1	euiccPackageResultDataSigned {
	  eimId paramEIMId,
	  counterValue paramCounterValue,
	  seqNumber SEQ_NUMBER,
	  euiccResult : {
	    deleteEimResult : 0
	  }
	},
	euiccSignEPR EUICC_SIGN_EPR

DELETE_EIM_RES_OK_2	euiccPackageResultDataSigned {
	  eimId paramEIMId,
	  counterValue paramCounterValue,
	  eimEimTransactionId S_EIM_TRANSACTION_ID,
	  seqNumber SEQ_NUMBER,
	  euiccResult : {
	    deleteEimResult : 0
	  }
	},
	euiccSignEPR EUICC_SIGN_EPR

LIST_EIM_RES_OK_1	euiccPackageResultDataSigned {
	  eimId paramEIMId,
	  counterValue paramCounterValue,
	  seqNumber SEQ_NUMBER,
	  euiccResult : {
	    listEim : { LIST_EIM_RESULT }
	  }
	},
	euiccSignEPR EUICC_SIGN_EPR

LIST_EIM_RES_OK_2	euiccPackageResultDataSigned {
	  eimId paramEIMId,
	  counterValue paramCounterValue,
	  eimEimTransactionId S_EIM_TRANSACTION_ID,
	  seqNumber SEQ_NUMBER,
	  euiccResult : {
	    listEim : { LIST_EIM_RESULT }
	  }
	},
	euiccSignEPR EUICC_SIGN_EPR

UPDATE_EIM_RES_ERR_COMMAND_ERROR	euiccPackageResultDataSigned {
	  eimId paramEIMId,
	  counterValue paramCounterValue,
	  seqNumber SEQ_NUMBER,
	  euiccResult : {
	    updateEimResult : 7
	  }
	},
	euiccSignEPR EUICC_SIGN_EPR

UPDATE_EIM_RES_ERR_COUNTER_OUT_OF_RANGE	euiccPackageResultDataSigned {
	  eimId paramEIMId,
	  counterValue paramCounterValue,
	  seqNumber SEQ_NUMBER,
	  euiccResult : {
	    updateEimResult : 6
	  }
	},
	euiccSignEPR EUICC_SIGN_EPR

UPDATE_EIM_RES_ERR_EIM_NOT_FOUND	euiccPackageResultDataSigned {
	  eimId paramEIMId,
	  counterValue paramCounterValue,
	  seqNumber SEQ_NUMBER,
	  euiccResult : {
	    updateEimResult : 1
	  }
	},
	euiccSignEPR EUICC_SIGN_EPR

UPDATE_EIM_RES_ERR_UNKNOWN_EUICC_CI_PKID	euiccPackageResultDataSigned {
	  eimId paramEIMId,
	  counterValue paramCounterValue,
	  seqNumber SEQ_NUMBER,
	  euiccResult : {
	    updateEimResult : 3
	  }
	},
	euiccSignEPR EUICC_SIGN_EPR

UPDATE_EIM_RES_OK_1	euiccPackageResultDataSigned {
	  eimId paramEIMId,
	  counterValue paramCounterValue,
	  seqNumber SEQ_NUMBER,
	  euiccResult : {
	    updateEimResult : 0
	  }
	},
	euiccSignEPR EUICC_SIGN_EPR

UPDATE_EIM_RES_OK_2	euiccPackageResultDataSigned {
	  eimId paramEIMId,
	  counterValue paramCounterValue,
	  eimEimTransactionId S_EIM_TRANSACTION_ID,
	  seqNumber SEQ_NUMBER,
	  euiccResult : {
	    updateEimResult : 0
	  }
	},
	euiccSignEPR EUICC_SIGN_EPR

Annex E Profiles

Those profiles are defined in SGP.23 [8].

Annex F IUT 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_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. This setting is only applicable for eUICCs supporting SGP.22 v2.3 or later. NOTE: 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, if provided, SHALL be either not present or: • other(0) • or basicEuicc(1) • or mediumEuicc(2) • or contactlessEuicc(3)
IUT_EUICC_FIRMWARE_VER	eUICC Firmware version coded as binary value (3 bytes representing major/minor/revision).
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_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_SAS_ACCREDITATION_NUMBER	SAS Accreditation Number, coded as ASN.1 UTF8String
IUT_SIMA_VERSION	Version of 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, 9.0.0 or higher). The support of Java Card is considered as mandatory in the scope of this specification. 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.

F.4 Common Settings

In order to execute the test cases defined in this document, the IUT provider SHALL deliver following settings:

IUT Setting name	Description
IUT_RSP_VERSION	Version of SGP.22 supported by the IUT encoded as a string of three integers separated with dots (for example: 2.1.0). In the scope of this specification, this value SHALL indicate one of the versions of SGP.22 for which this specification contains test cases, as specified in section 1.2.

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 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.1.1 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 default files system does not contain EF_ICCID.
- 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.
- 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.
    - 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).

The CI, identified as highest priority in euiccCiPKIdListForSigning, is also selectable in the euiccCiPKIdListForVerification (i.e. all EUM and eUICC Certificates lead to a Root CI certificate linked to a #PK_CI_ECDSA contained in the eUICC).

This CI corresponds to the SubjectKeyIdentifier of one of the #CERT_CI_ECDSA defined in sections G.2.2 and G.2.3.

The default RAT configuration defined in section G.2.4 applies for all test sequences except if the Test Case overrides it. Particular RAT configurations for those specific Test Cases are defined in section G.2.5.

G.1.2 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_ECDSA (for creating ECDSA signatures) based on NIST P-256 [11]
- The eUICC's Certificate #CERT_EUICC_ECDSA (for eUICC authentication) containing the eUICC's Public Key #PK_EUICC_ECDSA based on NIST P-256 [11]
- The GSMA Certificate Issuer's Public Key #PK_CI_ECDSA (for verifying off-card entities certificates) based on NIST P-256 [11]
- The Certificate of the EUM #CERT_EUM_ECDSA based on NIST P-256 [11]

Other Certificates and Keys MAY be present.

G.1.3 For eUICC supporting BrainpoolP256r1

If the eUICC supports O_E_BRP, the ECASD contains at least:

- The eUICC's Private Key #SK_EUICC_ECDSA (for creating ECDSA signatures) based on brainpoolP256r1 [8]
- The eUICC's Certificate #CERT_EUICC_ECDSA (for eUICC authentication) containing the eUICC's Public Key #PK_EUICC_ECDSA based on brainpoolP256r1 [8]
- The GSMA Certificate Issuer's Public Key #PK_CI_ECDSA (for verifying off-card entities certificates) based on brainpoolP256r1 [8]
- The Certificate of the EUM #CERT_EUM_ECDSA based on brainpoolP256r1 [8]
- Other Certificates and Keys MAY be present.

G.1.4 With default RAT configuration

The eUICC's RAT is configured as detailed in SGP.21 Annex H:

- Only one PPAR authorizing PPR1 and PPR2 for all MNOs with End User consent required i.e. #PPRS_ALLOWED

G.1.5 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

G.1.6 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.

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 [29] 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.

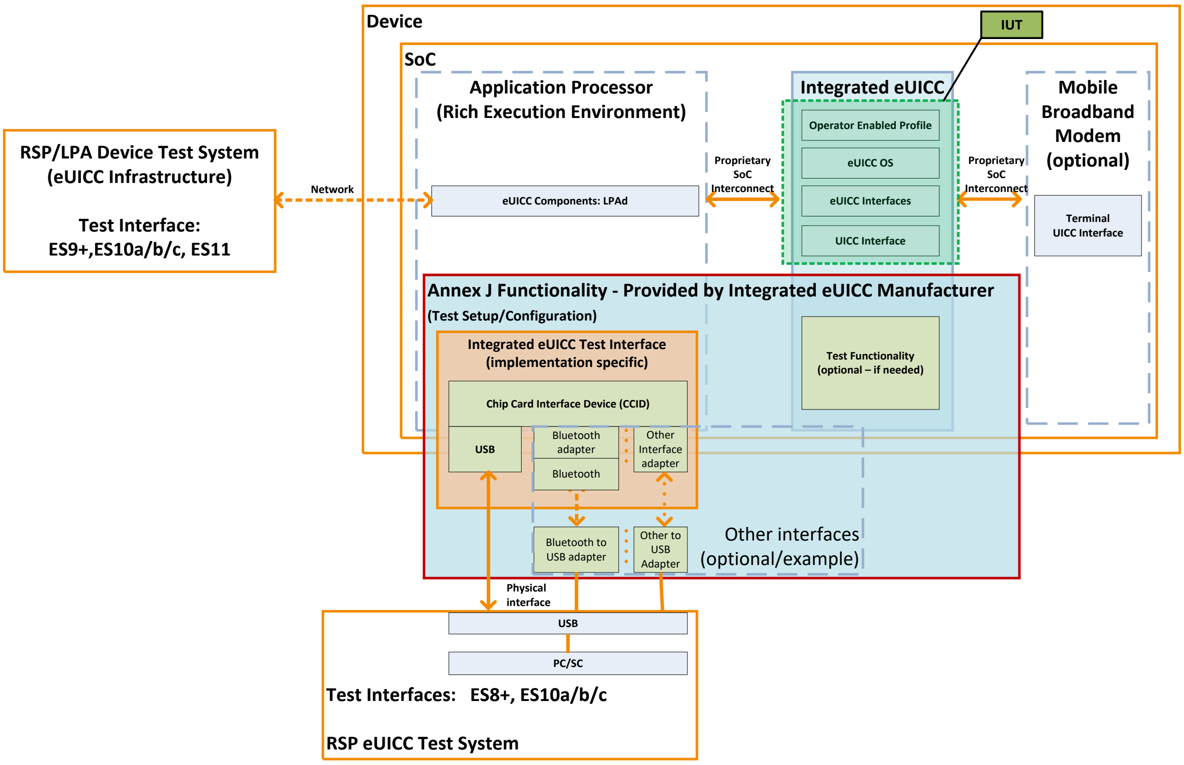


Figure 1 Integrated eUICC with USB CCID [29] Test Interface

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 [29] section 6 messages:

- [29] 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
- [29] 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 K Document Management

J.1 Document History

Version	Date	CR Number	Brief Description of Change	Approval Authority	Editor / Company
SGP.33-1 v1.0	26 January 2024	NA	Initial draft of SGP.33-1	ISAG	Yolanda Sanz, GSMA
		CR0002R00	UpdateEim test cases		
		CR0003R00	DeleteEim test cases		
		CR0004R00	ListEim test cases		
		CR0005R00	AddEim test cases		
		CR0006R01	Adding test cases to cover Enable function.		
		CR0007R01	Adding test cases to cover Disable function.		
		CR0008R02	Adding test cases to cover Delete function.		
		CR0009R01	Adding test cases to cover Disable Error Cases.		
		CR0010R01	Adding test cases to cover Delete Error Cases.		
		CR0011R01	Adding test cases to cover EnableError Cases.		
		CR0001R02	Constants and Methods for eUICC Package		
		CR00012R01	AddInitialEim test cases		
		CR00013R00	GetEimConfigData test cases		
		CR00014R01	ES10b_ProfileRollback		
		CR0016R01	Aligning AddEim test cases with PSMO test cases		
		CR000017R01	Aligning UpdateEim test cases with PSMO test cases		
		CR000018R01	Aligning DeleteEim test cases with PSMO test cases		
		CR000019R01	Aligning ListEim test cases with PSMO test cases		

		CR00020R03	LoadEUICCPackage test cases		
		CR00021R02	GetCerts test cases		
		CR0022R01	ESep_ListProfileInfo		
		CR0023R01	ESep_GetRat		
		CR0024R01	ES10_ConfigureAutomaticProfileEnabling		
		CR0025R01	ESep_Enable_ErrorCasesScenario5		
		CR0026R01	ES10bEnablingUsingDD		
		NA	Remove editor's note Delete Annex D commands and reponses table with empty values Fix the version of the SGP.xxx specifications according to the latest version published. Remove section 5 test cases as they are for further study.		
		CR0027	ApplicabilityTableForES10x		
		CR0028	IPAe testing ES9+ and ES11		
		NA	Mark eSIM – eUICC Test Caes Mandatory Mark eSIM – eIM Test Cases Mandatory Integreted eUICC Test cases – voidd and mark for FFS Section 5 Procedures deleted and mark as FFS. Adjust the Reference sections based on the versions agreed during eSIMWG3.101Bis Reference the test cases in SGP.23 for the eUICC memory reset nominal and error cases		

J.2 Other Information

Type	Description
Document Owner	Yolanda Sanz, GSMA
Editor / Company	Yolanda Sanz, 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.