RSP Technical Specification

Version 3.0

19th October 2022

This Industry Specification is a Non-binding Permanent Reference Document of the GSMA

Security Classification: Non-confidential

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.

Contents

1 Introduction

1.1 Overview

This document provides a technical description of the GSMA's 'Remote SIM Provisioning (RSP) Architecture for consumer Devices' [4] that applies for eSIM Products.

1.2 Scope

This specification provides a technical description of:

- The eUICC Architecture;
- The interfaces used within the Remote SIM Provisioning Architecture; and
- The security functions used within the Remote SIM Provisioning Architecture.

1.3 Document Purpose

This document defines a technical solution for the remote provisioning and management of the eUICC in consumer Devices as defined in RSP Architecture [4]. The adoption of this technical solution will provide the basis for global interoperability between different Operator deployment scenarios, for example network equipment (e.g., Subscription Manager Data Preparation (SM-DP+)) and various eUICC platforms.

1.4 Intended Audience

Technical experts working for Operators, SIM solution providers, consumer Device vendors, standards organisations, network infrastructure vendors, Service Providers and other industry bodies, etc.

1.5 Definition of Terms

Term	Description
Access Rules	Information stored on the eUICC that defines whether applications on the Device are denied or allowed access to applications on the eUICC as specified by GlobalPlatform Secure Element Access Control [56].
Activation Code	Information issued by an Operator/Service Provider to an End User. It is used by the End User to request the download and installation of a Profile.
Activation Code Token	The part of the Activation Code information provided by the Operator/Service Provider that refers to a Subscription.
Alternative SM-DS	SM-DS used in cascade mode with a Root SM-DS to redirect Event Registration from an SM-DP+ to that Root SM-DS used directly to perform Event Registration from an SM-DP+ or for an installed Profile.
Application Protocol Data Unit	The communication unit between the Device and the eUICC, as defined by ISO/IEC 7816-4 [14].
Bound Profile Package	A Protected Profile Package that has been cryptographically linked to a particular eUICC.

(Public Key) Certificate	A certificate as defined in RFC 5280 **Error! Reference source not found.Error! Reference source not found.**[17].
Certificate Authority	A Certificate Authority is an entity that issues (Public Key) Certificates.
Certificate Issuer	An Entity that is Authorised to Issue digital certificates.
Certified eUICC	An eUICC meeting the GSMA requirements for Remote SIM Provisioning and certified according to the GSMA compliance programme defined in [64]. Note: Unless stated otherwise, the word eUICC in this specification refers to a Certified eUICC.
Command Port	eSIM Port on which a given ES10 command is sent. It may be identical to the Target Port.
Companion Device	A Device that relies on the capabilities of a Primary Device for the purpose of Remote SIM Provisioning.
Confirmation Code	A code entered by an End User required by the SM-DP+ to validate the request to download a Profile or proceed with the Device Change.
Confirmation Code Required Flag	A parameter to indicate whether the Confirmation Code is required.
Confirmation Level	Refers to the hierarchy of User Intent and Confirmation Request, where: • User Intent is the first and lowest level • Simple Confirmation is the second level • Strong Confirmation is the third and highest level
Confirmation Request	A request confirming User Intent by using either Simple Confirmation or Strong Confirmation.
CRL Issuer	Designates an RSP entity that manages and publishes Certificate revocation status by means of CRL(s).
Default SM-DP+	An SM-DP+ that is contacted by an LPA by means of a pre-provisioned address on the eUICC.
Device	User equipment used in conjunction with an eUICC to connect to a mobile network. E.g., a tablet, wearable, smartphone or handset.
Device Application	An application installed in a Device.
Device Baseband	A Device component that manages the cellular radio functions.
Device Change	Given that one or more Profiles related to their respective Subscription are installed in an old Device, the process for installing one or more Profiles related to the same Subscription(s) onto a new Device.
Device Change Configuration	Set of rules that apply to a Profile to support Device Change.
Device Information Code	A set of Device-related information used for Remote SIM Provisioning operations (e.g., for Profile preparation between the SM-DP+ and the operator).
Delete Notification for Device Change	A report containing an information about the deletion of the Profile for Device Change on the old Device.

Device Reset	An action that returns the Device to a state equivalent to a factory state.
Device Test Mode	A mode hidden from the End User that allows access to and use of Test Profiles.
Digital Letter Of Approval	A digital representation of a Letter of Approval, signed by a DLOA Authority.
Disabled Profile	A Profile in a state where all files and applications (e.g., NAA) present in the Profile are not selectable.
Discrete eUICC	An eUICC implemented on discrete hardware.
DLOA Authority	In the context of this document, an entity that provides a certification, evaluation, approval, qualification, or validation scheme that delivers Digital Letters of Approval.
DLOA Management System	Any authorised system interested in verifying the level of certification, evaluation, approval, qualification, or validation of an eUICC (e.g., an MNO backend system, an SM-DP+). NOTE: This was called 'Management System' in the prior versions of this document.
DLOA Registrar	A role that stores DLOAs and provides an interface to enable Management System to retrieve them.
Enabled Profile	A Profile in a state where all files and/or applications (e.g., NAA) are selectable.
End User	The person using the Device.
Enterprise	A business, organization or government entity that subscribes to mobile services to be utilised by its workforce in support of the business or activities of the Enterprise. The Enterprise as the Subscriber owns the relationship with the Service Provider(s).
Enterprise Capable Device	A Device that supports the installation and enforcement of Enterprise Rules.
Enterprise Profile	An Operational Profile for which the Subscriber is an Enterprise. This Profile may include restrictions on the End User of the Device.
Enterprise Rule	A rule stored in an Enterprise Profile that can be used by the Profile Owner to restrict End User controllability for enabling and installing Profiles on Enterprise Capable Devices.
eSIM CA	A GSMA Certificate Issuer or an Independent eSIM CA.
eSIM Port	A logical connection between an endpoint in the Device and an endpoint in the eUICC (i.e., the Enabled Profile and/or the ISD-R in the context of this specification).
eSIM Products	eUICCs, Devices, and RSP Servers implementing this specification.
eUICC	A removable or non-removable UICC which enables the remote and/or local management of Profiles in a secure way. NOTE: The term originates from "embedded UICC".
eUICC Certificate	A certificate issued by the EUM for a specific eUICC. This certificate can be verified using the EUM Certificate.

eUICC Manufacturer	Manufacturer of the eUICC.
eUICC Memory Reset	An action that returns the eUICC to a state equivalent to a factory state.
eUICC OS Update	A mechanism to correct existing features on the eUICC by the original OS manufacturer.
eUICC Test Memory Reset	An action that deletes all post-issuance Test Profiles on an eUICC.
EUM Certificate	A certificate issued by an eSIM CA to a GSMA accredited EUM which can be used to verify eUICC Certificates.
EUM Keyset	Keyset used by the EUM to update ECASD content.
EUM Public Key	Public key included in the EUM Certificate (called PK.EUM.SIG).
EUM Private Key	Private Key used by the EUM to sign eUICC Certificates (called SK.EUM.SIG).
Event	A request for a Profile download or an RPM Operation which is set by an SM-DP+ on behalf of an Operator, to be processed by a specific eUICC.
EventID	Unique identifier of an Event for a specific EID generated by the SM-DP+ / SM-DS.
Event Checking	A process for the LDS to query the SM-DS to determine the presence of Event Records registered for an eUICC.
EventCheckingID (ECID)	Unique identifier for a specific EID generated by the SM-DS for Event Checking.
Event Record	The set of information stored on the SM-DS for a specific Event, via the Event Registration procedure.
Event Registration	A process notifying an SM-DS of the availability of information on either a specific SM-DP+ or a specific SM-DS for a specific eUICC.
Field-Test eUICC	A pre-production eUICC whose functional or security certifications are not yet completed by the EUM.
GSMA Certificate Issuer	A Certificate Authority accredited by GSMA.
HRI Server	Server providing High Resolution Icons
Independent eSIM CA	A non-GSMA CI that issues Certificates for a specific region, company or group of companies for eSIM purposes.
Integrated Circuit Card Identifier	Unique number to identify a Profile in an eUICC as defined by ITU-T E.118 [21].
Integrated eUICC	An eUICC implemented on a Tamper Resistant Element (TRE) that is integrated into a System-on-Chip (SoC), optionally making use of remote volatile/non-volatile memory
Integrated TRE	A TRE implemented inside a larger System-on-Chip (SoC)
International Mobile Subscriber Identity	Unique identifier owned and issued by Operators as defined in 3GPP TS 23.003 [35] section 2.2.
Issuer Identifier Number	The first 8 digits of the EID identifying the EUM issuing the eUICC.

Issuer Security Domain	A Security Domain on the UICC as defined by GlobalPlatform Card Specification [8].
Letter of Approval (LOA)	A letter generated by a Process Certification Authority, identifying a platform that has completed a certification, evaluation, approval, qualification, or validation scheme.
Local Profile Assistant (LPA)	A functional element in the Device or in the eUICC that provides the Local Profile Download (LPD), Local Discovery Services (LDS) and Local User Interface (LUI) features. When the LPA is located in the Device, they are called LPAd, LPDd, LUId, LDSd. When the LPA is located in the eUICC, they are called LPAe, LPDe, LUIe, LDSe. Where LPA, LPD, LDS or LUI are used, they apply to the element independent of its location in the Device or in the eUICC.
Local Profile Management	Local Profile Management are operations that are locally initiated on the End User (ESeu) interface.
Local Profile Management Operation	Local Profile Management Operations include enable Profile, disable Profile, delete Profile, query Profile Metadata, eUICC Memory Reset, eUICC Test Memory Reset, set/edit Nickname, add Profile and edit Default SM-DP+ address.
Logical SE Interface	As defined in ETSI TS 102 221 [6].
LPA Proxy	An LPA role that establishes a Profile Content Management session and optionally provides progress information about that session to a Device Application.
Managing SM-DP+	An SM-DP+ that is authorised by the Profile Owner to perform RPM to the eUICC on which their Profile resides.
MatchingID	Reference data for an RSP Server which could be an Activation Code Token or the EventID.
MEP-Capable Device	A Device where more than one Enabled Profile can be used in parallel on the same eUICC.
MEP-Capable eUICC	An eUICC where more than one Profile can be in Enabled state at the same point in time.
Mobile Network Operator	An entity providing access capability and communication services to its End User through a mobile network infrastructure.
Mobile Network Operator Security Domain (MNO-SD)	Part of the Profile, owned by the Operator, providing the Secured Channel to the Operator's Over The Air (OTA) Platform. It is used to manage the content of a Profile once the Profile is enabled.
Mobile Virtual Network Operator	An entity providing access capability and communication services to its Subscribers through a mobile network infrastructure but which does not have an allocation of spectrum.
Network Access Application	Application residing in a Profile providing authorisation to access a network.
NFC Device	A Device compliant with GSMA TS.26 [40].
NFC Profile	A Profile containing contactless applications.
Non-Enterprise Capable Device	A Device that does not support the enforcement of Enterprise Rules.

Notification	A report about a Profile download, Local/remote Profile Management Operation or Remote eUICC Management operation processed by the eUICC or about the progress of such an operation.
Operational Profile	A combination of Operator data and applications to be provisioned on an eUICC for the purposes of providing services by the Operator. The Profile SHALL be in support of a Subscription with the relevant Operator and allow connectivity to a mobile network. Applications MAY be included to provide non-telecommunication services.
Operator	A Mobile Network Operator or Mobile Virtual Network Operator; a company providing wireless cellular network services. NOTE: For historic reasons, this document uses the expression "Operator" for the entity in the architecture and as endpoint for the specified interfaces, even though the more generic expression "Service Provider" or "Operator/Service Provider" might be more applicable. However, the exact work split between a Mobile Network Operator, a Mobile Virtual Network Operator and any other Service Provider is irrelevant for this specification. Their overall activities are covered by the architecture entity "Operator".
Operator Credentials	A set of credentials owned by the Operator, including Network Access Credentials, OTA Keys for Remote File/Application management, and authentication algorithm parameters.
OTA Keys	The credentials included in the Profile, used in conjunction with OTA Platforms.
OTA Platform	An Operator platform for remote management of UICCs and the content of Enabled Operator Profiles on eUICCs.
Other Notification	Any Notification other than a Profile Installation Result and a Load RPM Package Result.
PIX	Proprietary application Identifier extension, the value of which is part of the Application Identifier (AID).
Polling Address	An address configured in a Profile indicating where the Device needs to connect to retrieve RPM Events for this Profile.
Primary Device	A Device that can be used to provide some capabilities to a Companion Device for the purpose of Remote SIM Provisioning.
Process Certification Authority	The authority that provides the certification, evaluation, approval, qualification, and validation scheme defined in SGP.24 and that delivers Digital Letters of Approval accordingly. In the context of this document, the Process Certification Authority is the GSMA.
Profile	A combination of data and applications to be provisioned on an eUICC for the purpose of providing services.

Profile Component	A Profile Component is an element of the Profile, when installed in the eUICC, and MAY be one of the following: • An element of the file system like an MF, EF or DF; • An Application, including NAA and Security Domain; • Profile Metadata, including Profile Policy Rules; • An MNO-SD.
Profile Content Management	The update of an Enabled Profile by its Profile Owner, using an agent (the PCMAA) on the Device as an intermediary between the Profile and a remote server (the PCMP).
Profile Content Management Platform (PCMP)	Platform owned by the Profile Owner, used to manage the content of an enabled Profile. As this specification defines a generic mechanism for the PCMAA to request the next data from a different management platform, the term PCMP is used for all of these platforms and the term Delegated PCMP (DPCMP) is not used in this specification.
Profile Installation Result	A Notification that contains the result of a Profile installation.
Profile Management	A combination of local and remote management operations (enable Profile, disable Profile, delete Profile, list Profile information and query Profile Metadata).
Profile Management Operation	An operation related to the content and state update of a Profile in a dedicated ISD-P on the eUICC.
Profile Metadata	Information pertaining to a Profile used for the purpose of Local Profile Management and Remote Profile Management.
Profile Nickname	Alternative name of the Profile set by the End User.
Profile Owner	The entity that controls the operations that can be performed upon its Profile. With the exception of Test Profiles, this is always the Operator.
Profile Package	A personalised Profile using an interoperable description format that is transmitted to an eUICC to load and install a Profile.
Profile Policy Authorisation Rule	A set of data that governs the ability of a Profile Owner to make use of a Profile Policy Rule in a Profile.
Profile Policy Enabler	The functional element within eUICC that interprets and enforces Profile Policy Rules.
Profile Policy Management	A policy control system that allows the Service Provider to implement, manage and enforce its subscription terms and conditions associated with the installed Profile.
Profile Policy Rule	Defines a qualification for or enforcement of an action to be performed on a Profile when a certain condition occurs.
Profile Recovery	The process for re-installing a Profile related to the Subscription onto the old Device under condition that the Profile Installation onto the new Device for Device Change is failed due to a permanent error.
Profile Type	Operator specific defined type of Profile. This is equivalent to the "Profile Description ID" as described in Annex B of SGP.21 [4]

Protected Profile Package	A Profile Package which has been cryptographically protected for storage but not linked to a particular eUICC.
Provisioning Profile	A combination of Operator data and applications to be provisioned on an eUICC for the purposes of providing connectivity to a mobile network solely for the purpose of the provisioning of Profiles on the eUICC. NOTE: Use of Provisioning Profiles for other system services in version 3 of this specification may require modifications of this definition.
Public CA	A Certificate Authority, commonly used to issue certificates for public Internet purposes, which is not subject to the GSMA Policy Authority as defined in SGP.14 [45].
Push Service	A service provided by a combination of a push server and a push client on the Device, able to send a push notification from an SM-DS to an LDS.
Push Token	An identifier that is generated when an LDS registered a Push Service to a push server.
Reference Enterprise Rule	The Enterprise Rule that is currently being enforced by the eUICC.
Remote Profile Management	Profile Management operations performed by a Managing SM-DP+ at the request of the Profile Owner.
Remote SIM Provisioning	The downloading, installing, enabling, disabling, and deleting of a Profile on an eUICC.
Roles	Roles are representing a logical grouping of functions.
Root SM-DS	A central access point for finding Events from one or more SM-DP+(s).
Rules Authorisation Table	A set of Profile Policy Authorisation Rules that, together, determines the ability of a Profile Owner to make use of a set of Profile Policy Rules in a Profile.
RSP Server	Either an SM-DS or SM-DP+.
RSP Session	Sequence of interlinked data exchanges between the eUICC, the LPA and an RSP Server, starting with the generation of a challenge on the eUICC and ending with a successful operation (Profile installation, RPM execution or Event List verification) on the eUICC or by explicit or implicit cancellation.
Service Provider	The organization through which the End User obtains PLMN telecommunication services. This is usually the network operator or possibly a separate body. See also the definition for "Operator".
Simple Confirmation	A secure and non-interceptable mechanism by which the End User confirms their action, e.g., by selecting Yes/No, OK/Cancel.
SM-DP+ Certificate	A Certificate issued to a GSMA accredited SM-DP+, which chains to an eSIM CA RootCA Certificate.
SM-DS Certificate	A Certificate issued to a GSMA accredited SM-DS, which chains to an eSIM CA RootCA Certificate.

SM-DP+ OID	Identifier of the SM-DP+ that is globally unique and is included as part of the SM-DP+ Certificate.
SM-DS OID	Identifier of the SM-DS that is globally unique and is included as part of the SM-DS Certificate.
Strong Confirmation	A secure and non-interceptable mechanism to guarantee a higher level of User Intent than Simple Confirmation by which the End User confirms their action, e.g., by inputting PIN or fingerprint, repeating Simple Confirmation, entering Confirmation Code, etc.
SubCA	A CA whose Certificate is signed by another CA.
Subscriber	An entity (associated with one or more End Users) that is engaged in a Subscription with a Operator/Service Provider.
Subscription	Describes the commercial relationship between the Subscriber and the Operator/Service Provider.
Subscription Manager Data Preparation+ (SM-DP+)	Prepares Profile Packages, secures them with a Profile Protection Key, stores Profile Protection Keys in a secure manner and the Protected Profile Packages in a Profile Package repository, and allocates the Protected Profile Packages to specified EIDs. The SM-DP+ binds Protected Profile Packages to the respective EID and securely downloads these Bound Profile Packages to the LPA of the respective eUICC. The SM-DP+ also performs Remote Profile Management and Remote eUICC Management.
Subscription Manager Discovery Service (SM-DS)	This is responsible for providing addresses of one or more SM-DP+(s) to a LDS.
Tamper Resistant Element	A security module consisting of hardware and low-level software providing resistance against software and hardware attacks, capable of securely hosting operating systems together with applications and their confidential and cryptographic data.
Target Port	eSIM Port which is explicitly affected by an ES10c.EnableProfile or ES10c.DisableProfile command.
Target Profile	Profile that is explicitly affected by an ES10 command. NOTE: An ES10c.EnableProfile command may implicitly affect another Profile by implicitly disabling it.
Test Profile	A combination of data and applications to be provisioned on an eUICC to provide connectivity to test equipment for the purpose of testing the Device and the eUICC. A test profile is not intended to store any Operator Credentials.
User Intent	Describes the direct, real time acquisition and validation of the manual End User input on the LUI to trigger locally a Profile download or Profile Management Operation. As defined in SGP.21 [4].

1.6 Abbreviations and Notations

Abbreviation	Description
AES	Advanced Encryption Standard
AID	Application Identifier
APDU	Application Protocol Data Unit
API	Application Programming Interface
ASN.1	Abstract Syntax Notation One
BPP	Bound Profile Package
BSP	BPP Security Protocol
CA	Certificate Authority
CASD	Controlling Authority Security Domain
CAT	Card Application Toolkit
CI	Certificate Issuer
CMAC	Cipher-based MAC
CP	Command Port
CRL	Certificate Revocation List
CRT	Control Reference Template
DH	Diffie-Hellman
DEV-IC	Device Information Code
DLOA	Digital Letter Of Approval
DPI	Delegated Platform Identifier
E4E	E4 ENVELOPE (ENVELOPE command with tag 'E4')
ECASD	eUICC Controlling Authority Security Domain
ECC	Elliptic Curve Cryptography
ECDHE	Elliptic Curve Diffie-Hellman using Ephemeral keys
ECDSA	Elliptic Curve cryptography Digital Signature Algorithm
ECID	Event Checking Identifier
ECKA	Elliptic Curve cryptography Key Agreement algorithm
EID	eUICC identifier
EIN	EUM Identification Number
ESIN	EUM Specific Identification Number
ETSI	European Telecommunications Standards Institute
EUM	eUICC Manufacturer
FCI	File Control Information
FFS	For Further Study
FQDN	Fully Qualified Domain Name
GID1	Group Identifier 1, as defined in 3GPP TS 31.102 [54]
GID2	Group Identifier 2, as defined in 3GPP TS 31.102 [54]
GP	GlobalPlatform

GSMA	GSM Association
GSMA CI	GSM Association Certificate Issuer
HLR	Home Location Register
HRI	High Resolution Icon
ICCID	Integrated Circuit Card ID
ICV	Initial Chaining Vector
IIN	Issuer Identifier Number
IMEI	International Mobile Equipment Identity
IMSI	International Mobile Subscriber Identity
ISD	Issuer Security Domain
ISD-P	Issuer Security Domain Profile
ISD-R	Issuer Security Domain Root
ISO	International Standards Organisation
ITU	International Telecommunications Union
KA	Key Agreement
LDS	Local Discovery Service
LDSd	Local Discovery Service when LPA is in the Device
LDSe	Local Discovery Service when LPA is in the eUICC
LOA	Letter Of Approval
LPA	Local Profile Assistant
LPAd	Local Profile Assistant when LPA is in the Device
LPAe	Local Profile Assistant when LPA is in the eUICC
LPD	Local Profile Download
LPDd	Local Profile Download when LPA is in the Device
LPDe	Local Profile Download when LPA is in the eUICC
LPM	Local Profile Management
LPRd	LPA Proxy when LPA is in the Device
LSI	Logical SE Interface
LTE	Long Term Evolution
LUI	Local User Interface
LUId	Local User Interface when LPA is in the Device
LUIe	Local User Interface when LPA is in the eUICC
M4M	Mifare4Mobile^{TM}
MAC	Message Authentication Code
MEP	Multiple Enabled Profiles
MNO	Mobile Network Operator
MNO-SD	Mobile Network Operator - Security Domain
MOC	Mandatory, Optional or Conditional

MXP	Message eXchange Pattern Note: In version 2.X of this specification, this was abbreviated as MEP.
NAA	Network Access Application
OS	Operating System
OTA	Over The Air
PCM	Profile Content Management
PCMAA	PCM Admin Agent
PCMP	Profile Content Management Platform
PE	Profile Element
PIX	Proprietary application Identifier eXtension
PKI	Public Key Infrastructure
POS	Point Of Sale
PPAR	Profile Policy Authorisation Rule
PPE	Profile Policy Enabler
PPK	Profile Protection Key
PPK-ENC	Optional Profile Protection Key randomly generated by the SM-DP+ and used for encryption/decryption of a Protected Profile Package
PPK-MAC	Optional Profile Protection Key randomly generated by the SM-DP+ and used for MAC generation/verification of a Protected Profile Package
PPP	Protected Profile Package
PPR	Profile Policy Rule
RAT	Rules Authorisation Table
RFU	Reserved for Future Use
RPM	Remote Profile Management
RSA	Rivest / Shamir / Adleman asymmetric algorithm
RSP	Remote SIM Provisioning
SAS	Security Accreditation Scheme
SBPP	Segmented Bound Profile Package
SCP	Secure Channel Protocol
SCWS	Smartcard Web Server
SD	Security Domain
SEAC	Secure Element Access Control
SEP	Single Enabled Profile
SIM	Subscriber Identity Module
SVN	SGP.22 Specification Version Number (referred to as 'eSVN' in SGP.21 [4]).
SM-DP+	Subscription Manager Data Preparation (Enhanced compared to the SM-DP in SGP.02 [2])

SM-DS	Subscription Manager Discovery Service
S-ENC	Session key resulting from the key agreement between the SM-DP+ and the eUICC and used for encryption of parts of a Bound Profile Package
S-MAC	Session Key resulting from the key agreement between the SM-DP+ and the eUICC and used for MAC computation of parts of a Bound Profile Package
TAC	Type Allocation Code
TAR	Toolkit Application Reference
TLS	Transport Layer Security
TLV	Tag-Length-Value
TP	Target Port
TRE	Tamper Resistant Element
UI	User Interface
UIM	User Interface Module for LPAe
UPP	Unprotected Profile Package
URI	Uniform Resource Identifier
URL	Uniform Resource locator
USIM	Universal Subscriber Identity Module
W3C	World Wide Web Consortium

1.7 References

1.7.1 Normative References

Ref	Document Number	Title
[1]	Void	Void
[2]	SGP.02	GSMA "Remote Provisioning of Embedded UICC Technical specification" V3.1
[3]	Void	Void
[4]	SGP.21	RSP Architecture V3.0
[5]	eUICC Profile Package	Trusted Connectivity Alliance (TCA) eUICC Profile Package: Interoperable Format Technical Specification V3.2
[6]	ETSI TS 102 221	Smart Cards; UICC-Terminal interface; Release 17
[7]	OMA-TS-Smartcard_Web_Server-V1_2_1-20130913-A	Open Mobile Alliance: Smartcard-Web-Server, Version 1.2.1 – 13 Sep 2013
[8]	GPC_SPE_034	GlobalPlatform Card Specification v2.3
[9]	GPC_SPE_007	GlobalPlatform Card Specification v2.3 Amendment A: Confidential Card Content Management v1.1

[10]	GPC_SPE_025	GlobalPlatform Card Specification v2.3 Amendment C: Contactless Services v1.2
[11]	GPC_SPE_014	GlobalPlatform Card Specification v2.2 Amendment D: Secure Channel Protocol '03' v1.1.1
[12]	GPC_SPE_042	GlobalPlatform Card Specification v2.2 Amendment E: Security Upgrade for Card Content Management v1.0
[13]	GPC_SPE_093	GlobalPlatform Card Specification v2.2 Amendment F: Secure Channel Protocol '11' v1.3
[14]	ISO/IEC 7816-4:2013	Identification cards – Integrated circuit cards - Part 4: Organization, security and commands for interchange
[15]	ISO/IEC 18004:2015	Information technology -- Automatic identification and data capture techniques -- QR Code bar code symbology specification
[16]	RFC 5246	The TLS Protocol – Version 1.2
[17]	RFC 5280	Internet X.509 PKI Certificate and CRL Profile
[18]	RFC 5639	Elliptic Curve Cryptography (ECC) Brainpool Standard Curves and Curve Generation
[19]	RFC 793	Transmission Control Protocol, DARPA Internet Program, Protocol specification, Sept 1981
[20]	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
[21]	ITU E.118	The international telecommunication charge card
[22]	GSMA Security Principles Related to Handset Theft	GSMA Doc Reference: Security Principles Related to Handset Theft 3.0.0 EICTA CCIG Doc Reference: EICTA Doc: 04cc100
[23]	FS.08	GSMA SAS Standard for Subscription Manager Roles Version 3.0 - 31 March 2017
[24]	ITU-T X.520	ITU-T X.520 Information technology – Open Systems Interconnection – The Directory: Selected attribute types
[25]	RFC 5758	RFC 5758 Internet X.509 Public Key Infrastructure: Additional Algorithms and Identifiers for DSA and ECDSA
[26]	RFC 5759	RFC 5759 Suite B Certificate and Certificate Revocation List (CRL) Profile
[27]	RFC 5480	RFC 5480 Elliptic Curve Cryptography Subject Public Key Information
[28]	RFC 4519	Lightweight Directory Access Protocol (LDAP)
[29]	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
[30]	ITU E.212	The international identification plan for public networks and Subscriptions
[31]	ETSI TS 102 223	Smart Cards; Card Application Toolkit (CAT); Release 17

[32]	3GPP TS 24.008	Digital cellular telecommunications system (Phase 2+);
		Universal Mobile Telecommunications System (UMTS);
		LTE;
		Mobile radio interface Layer 3 specification;
		Core network protocols;
		Stage 3

[33]	ETSI TS 101 220	Smart Cards; ETSI numbering system for telecommunication application providers
[34]	RFC 768	User Datagram Protocol, Aug 1980.
[35]	3GPP TS 23.003	Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); Numbering, addressing and identification
[36]	3GPP2 S.R0048-A	3GPP2 - 3G Mobile Equipment Identifier (MEID)
[37]	ISO/IEC 7812-1:2015	Identification cards -- Identification of issuers -- Part 1: Numbering system
[38]	ETSI TS 102 225	Secured packet structure for UICC based applications; Release 12
[39]	ETSI TS 102 226	Remote APDU structure for UICC based applications; Release 9
[40]	TS.26	GSMA NFC Handset Requirements V9.0
[41]	BSI TR-03111	BSI Technical Guideline, TR-03111; Elliptic Curve Cryptography; Version 2.10
[42]	TLS 1.3	The Transport Layer Security (TLS) Protocol Version 1.3
[43]	RFC 2986	PKCS #10: Certification Request Syntax Specification
[44]	Void	Void
[45]	SGP.14	GSMA eUICC PKI Certificate Policy V1.1
[46]	RFC 5289	TLS Elliptic Curve Cipher Suites with SHA-256/384 and AES Galois Counter Mode (GCM)
[47]	RFC 4279	Pre-Shared Key Cipher suites for Transport Layer Security (TLS)
[48]	Void	Void
[49]	ITU-T X.680 (11/2008)	Abstract Syntax Notation One (ASN.1): Specification of basic notation including Corrigendum 1 and 2
[50]	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
[51]	3GPP TS 35.231	Specification of the TUAK Algorithm Set; Document 1: Algorithm Specification
[52]	3GPP TS 35.205	Specification of the MILENAGE Algorithm Set; Document 1: General
[53]	ETSI TS 102 241	Smart cards; UICC Application Programming Interface (UICC API) for Java Card™; Release 17
[54]	3GPP TS 31.102	Characteristics of the Universal Subscriber Identity Module (USIM) application

[55]	SGP.03	GSMA NFC UICC Requirements Specification V6.1
[56]	GPD_SPE_013	GlobalPlatform Device Technology – Secure Element Access Control - Version 1.1
[57]	GPC_SPE_095	GlobalPlatform Card - Digital Letter of Approval - Version 1.0
[58]	M4M	MIFARE4Mobile Architecture – V 2.1.1
[59]	ISO/IEC 10646:2014	Information technology — Universal Coded Character Set (UCS)
[60]	RFC 6066	Transport Layer Security (TLS) Extensions: Extension Definitions
[61]	RFC 2119	Key words for use in RFCs to Indicate Requirement Levels, S. Bradner http://www.ietf.org/rfc/rfc2119.txt
[62]	3GPP TS 34.108	Common test environments for User Equipment (UE); Conformance testing
[63]	3GPP TS 29.002	Mobile Application Part (MAP) specification
[64]	SGP.24	RSP Compliance Process
[65]	RFC 8422	Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS) Versions 1.2 and Earlier
[66]	RFC 7027	Elliptic Curve Cryptography (ECC) Brainpool Curves for Transport Layer Security (TLS)
[67]	RFC 2818	HTTP Over TLS
[68]	RFC 7159	IETF - The JavaScript Object Notation (JSON) Data Interchange Format
[69]	GPD_SPE_075	GlobalPlatform Technology Open Mobile API Specification V3.3
[70]	RFC 5646	Tags for Identifying Languages
[71]	RFC 4648	The Base16, Base32, and Base64 Data Encodings
[72]	RFC 3986	Uniform Resource Identifier (URI): Generic Syntax
[73]	WebIDL	WebIDL Level 1, W3C Recommendation, 15 December 2016
[74]	GPD_SPE_008	GlobalPlatform Device Technology – Secure Element Remote Application Management NOTE: The current version 1.0.1 of this GlobalPlatform specification does not cover all requirements from SGP.21 v3.0 [4]. The version will be updated once the new specification is available from GlobalPlatform.
[75]	RFC 1341	MIME (Multipurpose Internet Mail Extensions): Mechanisms for Specifying and Describing the Format of Internet Message Bodies
[76]	3GPP TS 31.111	3rd Generation Partnership Project; Technical Specification Group Core Network and Terminals; Universal Subscriber Identity Module (USIM) Application Toolkit (USAT)
[77]	FS.04	GSMA SAS Standard for UICC Production Version 8.0 - 31 March 2017
[78]	BSI TR-02102-1, Version 2018-01	BSI Technische Richtlinie TR-02102-1: Kryptographische Verfahren: Empfehlungen und Schlüssellängen

[79]	NIST SP 800-90A Rev. 1	Recommendation for Random Number Generation Using Deterministic Random Bit Generators, June 2015
[80]	RFC 3490	Internationalizing Domain Names in Applications (IDNA)
[81]	RFC 7230	Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing
[82]	RFC 7231	Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content
[83]	NIST SP 800-38A	Recommendation for Block Cipher Modes of Operation: Methods and Techniques, 2001.
[84]	NIST SP 800-38B	NIST, Special Publication 800-38B, "Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication", May 2005.
[85]	TS.48	GSMA Generic eUICC Test Profile for Device Testing
[86]	NIST SP 800-57	NIST Special Publication 800-57 Part 1 Rev. 4, Recommendation for Key Management
[87]	3GPP TS 33.501	3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Security architecture and procedures for 5G system
[88]	3GPP TS 31.130	3rd Generation Partnership Project; Technical Specification Group Core Network and Terminals; (U)SIM Application Programming Interface (API); (U)SIM API for Java™ Card;
[89]	SGP.29	GSMA EID Definition and Assignment Process V1.0
[90]	RFC 8410	Algorithm Identifiers for Ed25519, Ed448, X25519, and X448 for Use in the Internet X.509 Public Key Infrastructure
[91]	RFC 7905	ChaCha20-Poly1305 Cipher Suites for Transport Layer Security (TLS)
[92]	SGP.25	GSMA Embedded UICC for Consumer Devices – Protection Profile
[93]	SM2 algorithm	ISO/IEC 14888-3:2018 IT Security techniques – Digital signatures with appendix – Part 3: Discrete logarithm based mechanisms
[94]	SM3 algorithm	ISO/IEC 10118-3:2018 IT Security techniques – Hash-functions – Part 3: Dedicated hash-functions
[95]	SM4 algorithm	ISO/IEC 18033-3:2010/AMD1:2021 Information technology – Security techniques – Encryption algorithms – Part 3: Block ciphers – Amendment 1: SM4
[96]	RFC8998	RFC 8998 ShangMi (SM) Cipher Suites for TLS 1.3

1.7.2 Informative References

Ref	Document Number	Title
[i1]	TS.43	GSMA VoWiFi and VoLTE Entitlement Configuration V4.0
[i2]	OCF Easy Setup Specification	OCF Easy Setup Specification v2.2.2: https://openconnectivity.org/specs/OCF_Easy_Setup_Specification_v2.2.2.pdf

1.8 Conventions

The key words "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", and "MAY" in this document SHALL be interpreted as described in RFC 2119 [61].

1.9 Feature Support

This specification indicates features and data elements that were added starting from v3.0.0 by acronyms encapsulated within hash symbols. This allows the sender of information to limit the content of its messages to what the receiver indicated to support.

The following acronyms are used:

#SupportedOnlyBeforeV3.0.0#
These data elements were used in pre-3.0.0 versions of the specification and SHALL NOT be sent to a receiver indicating support for v3.0.0 or higher.

#SupportedFromV2.X.Y#
These data elements relate to a feature introduced in v2.X.Y.

#DeviceInfoExtensibilitySupported#
These data elements relate to the support of Device Information extensibility. These SHOULD only be sent to an eUICC indicating support for the extensibility (see section 4.3).

#SupportedFromV3.0.0#
These data elements relate to a feature introduced in v3.0.0. These SHALL NOT be sent to an eUICC indicating support for a version lower than v3.0.0.

#SupportedForRpmV3.0.0#
These data elements relate to the support of the Remote Profile Management, a feature introduced in v3.0.0. These SHALL only be sent to an eUICC indicating support for this feature.

#SupportedForEnterpriseV3.0.0#
These data elements relate to the support of Enterprise Profiles, an optional feature introduced in v3.0.0. These SHALL only be sent to an eUICC indicating support for this feature.

#SupportedForLpaProxyV3.0.0#
These data elements relate to the support of the LPA Proxy/Profile Content Management, an optional feature introduced in v3.0.0. These SHALL only be sent to an eUICC indicating support for this feature.

#SupportedForDcV3.0.0#
These data elements relate to the support of the Device Change, an optional feature introduced in v3.0.0.

#SupportedForEventCheckingV3.0.0#
These data elements relate to the support of the Event Checking, an optional feature introduced in v3.0.0.

#SupportedForPushServiceV3.0.0#
These data elements relate to the support of the Push Service, an optional feature introduced in v3.0.0.

#SupportedForMEPV3.0.0#
These data elements relate to the support of the Multiple Enabled Profiles, an optional feature introduced in v3.0.0.

#MandatoryFromV3.0.0#
These data elements were optional or not defined in pre-3.0.0 versions and SHALL always be provided in v3.0.0 or higher.

Backward compatibility with version 1 of this specification is not supported.

1.10 Editorial Guidelines

This specification is built according to these editorial guidelines:

```
ASN.1 code
```

`ASN.1 references used as character style`

```
JSON definitions
```

`Input Data, Output Data`

Status code "Subject - Reason"

Names and phrases that are defined in section 1.5 or referenced in the definition of acronyms in section 1.6 are capitalised. Words or phrases used according to their commonly understood meanings are not capitalised.

In addition, within section titles, table captions, and figure captions, the first word and every other word with length greater than three characters is capitalised.

Octet values and octet string values are represented by uppercase hexadecimal digits between single quotation marks, optionally with spaces after each pair of digits. For example, 'BF12' or '12 34 56 78'.

ASN.1 code is represented by lines containing no more than 80 characters. Where necessary to conform to this guideline, long definitions are represented by multiple consecutive lines.

Comments within ASN.1 code use only the single-line format defined in section 12.6.3 of [49]. The following additional considerations apply:

- Comments, which begin with a pair of hyphens (i.e., `--`), are always terminated by the end of the line (i.e., never by a second pair of hyphens).
- Comments that would result in lines longer than 80 characters are represented by multiple consecutive single-line comments, each less than 80 characters in width.

2 General Architecture

This section contains a technical description and architecture of the Remote SIM Provisioning System for consumer Devices. The statements in this section define the basic characteristics that need to be taken into account when reviewing this specification.

2.1 General Architecture Diagram

This section further specifies the Roles and interfaces associated with the Remote SIM Provisioning and Management of the eUICC for consumer Devices.

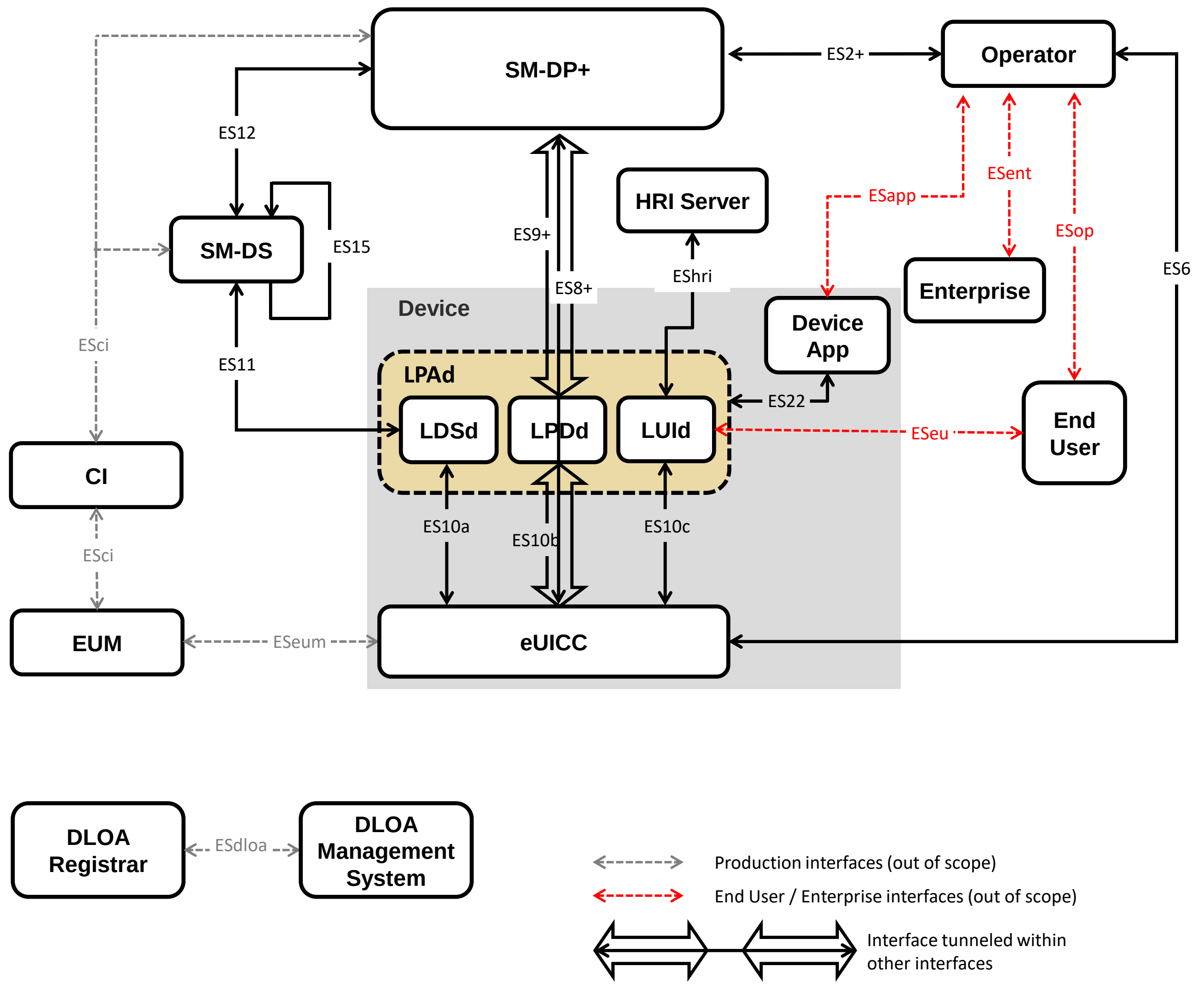


Figure 1: Remote SIM Provisioning System, LPA in the Device

Figure 1 shows the entities required for Profile Download and RPM. Figure 1a shows the additional entities required for Profile Content Management using an LPA Proxy (LPRd) and a PCM Admin Agent. The PCM Admin Agent MAY be external to the LPAd.

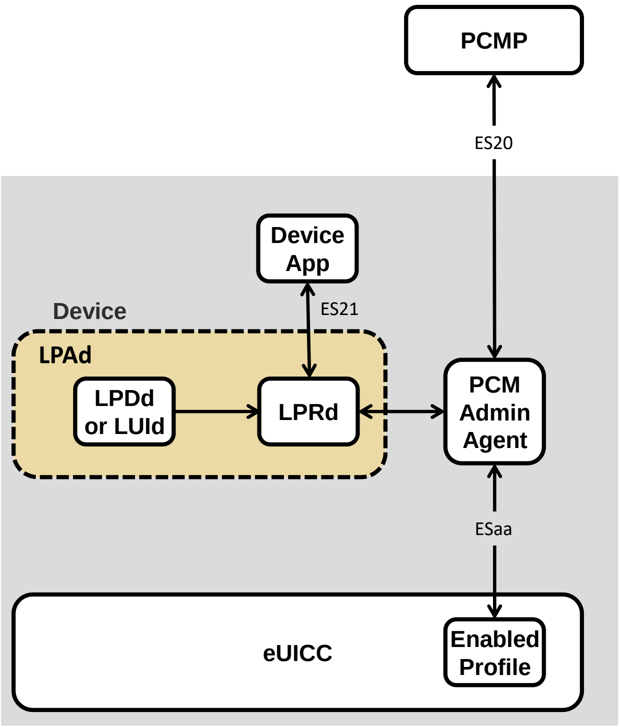


Figure 1a: Entities for Profile content management, LPA in the Device

PCM enables the exchange of APDUs between the PCMP and the Enabled Profile. This includes using all kind of APDU based Secure Channel Protocols, e.g., SCP03 [11] and SCP11 [13] defined by GlobalPlatform.

The LPRd triggers the PCM Admin Agent to start a PCM session and optionally relays progress information from the PCM Admin Agent to a Device Application. (See section 3.9.)

A Device with an LPAd SHOULD support PCM.

An eUICC that supports both the LPA Proxy and Remote Profile Management SHALL support the 'Contact PCMP' command.

A Device compliant with this specification SHALL implement at least one of the following:

- The LPAd, or
- The requirements for one of the options for the LPAe (section 5.11).

A Device that supports a non-removable eUICC without an LPAe SHALL provide an LPAd.

An eUICC compliant with this specification SHALL satisfy the following:

- The eUICC MAY implement the LPAe.
- A removable eUICC SHALL implement the LPA Services.
- A non-removable eUICC in a Device containing an LPAd SHALL implement the LPA Services.

A Device supporting both the LPAd and the LPAe SHALL implement an appropriate mechanism that sets the LPA to be used.

The above figure provides the complete description of the consumer Remote SIM Provisioning and Management system, when LPA is in the Device (LPAd).

The Remote SIM Provisioning and Management system also allows to have the LPA in the eUICC (LPAe). This architecture is shown in the following figure.

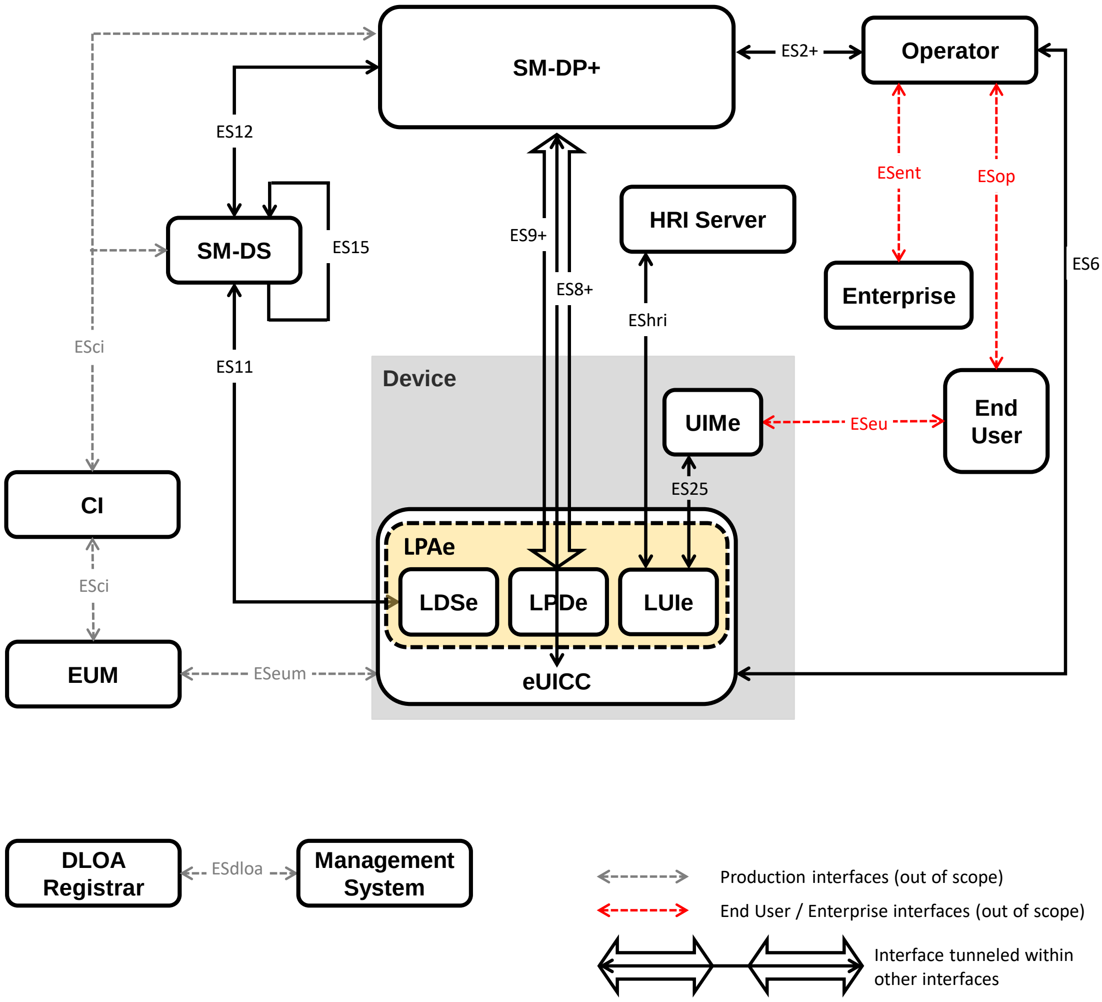


Figure 2: Remote SIM Provisioning System, LPA in the eUICC

NOTE: LPR is not defined for LPAe. This may be added in a future version of this specification.

2.2 Roles

Roles are defined within SGP.21 [4] Architecture Specification section 3.

The DLOA Registrar is a role that stores DLOAs and provides an interface to enable authorised DLOA Management System to retrieve them. In the context of RSP, a well-known DLOA Registrar SHALL be defined (i.e., a well-known URL SHALL be defined), containing all the valid (not expired and not revoked) DLOAs delivered by the DLOA Authority. This DLOA Registrar MAY also contain additional DLOAs delivered by other authorities. DLOAs delivered by the DLOA Authority MAY be provided to other DLOA Registrars. The DLOA Registrar is defined in GlobalPlatform DLOA [57].

A DLOA Management System is any authorised system (e.g., a MNO backend system, an SM-DP+) interested in verifying the level of certification, evaluation, approval, qualification, or validation of a component (e.g., eUICC platform).

2.3 Interfaces

The following table provides information about the interfaces within the architecture.

Interface	Between		Description
ES2+	Operator	SM-DP+	Used by the Operator to order Profiles for specific eUICCs as well as other administrative functions.
ES6	Operator	eUICC	Used by the Operator for the management of Operator services via OTA services.
ES8+	SM-DP+	eUICC	Provides a secure end-to-end channel between the SM-DP+ and the eUICC for the administration of the ISD-P and the associated Profile during download and installation. It provides Perfect Forward Secrecy.
ES9+	SM-DP+	LPD	Used to provide a secure transport between the SM-DP+ and the LPA (LPD) for Profile Download, RPM, etc.
ES10a	LDSd	eUICC	Used between the LDSd and the LPA Services to handle a Profile discovery.
ES10b	LPDd	eUICC	Used between the LPDd and the LPA services to transfer a Bound Profile Package to the eUICC. This interface plays no role in the decryption of Profile Packages.
ES10c	LUId	eUICC	Used between the LUId and the LPA services for Local Profile Management by the End User.
ES11	LDS	SM-DS	Used by the LDS to retrieve Event Records for the respective eUICC.
ES12	SM-DP+	SM-DS	Used by the SM-DP+ to register Event Records with the SM-DS and to delete its Event Records.
ES15	SM-DS	SM-DS	Used in the case of deployments of cascaded SM-DSs to connect those SM-DSs.
ES20	PCMP	PCMAA	Used to send APDU scripts enhanced by additional information.
ES21	Device Application	LPRd	Used to trigger the PCMAA and to send PCM notifications to the Device Application.
ES22	Device Application	LPAd	Used by a Device Application to interwork with the LPA (out of scope of this specification).
ES25	UIMe	LUIe	Used between the UIMe and the LUIe to transfer End User related interaction.
ESaa	PCMAA	enabled Profile	Used to select an application of the Enabled Profile and to exchange APDUs subsequently. Identical to the interface between the Admin Agent and the Secure Element

Interface	Between		Description
			defined in GP SERAM [74] and not further described in this specification.
ESapp	Operator	Device Application	Interface between the Operator and the Device Application (out of scope of this specification). NOTE: an example of a protocol that can be used on ESapp can be found in [i1].
EShri	LUI	HRI Server	Used to retrieve High Resolution Icons.
ESop	Operator	End User	Business interface between Operator and End user (out of scope of this specification).
ESent	Operator	Enterprise	Interface between the Operator and the Enterprise (out of scope of this specification).
ESeu	End User	LUI	Interface to initiate local profile management functions (out of scope of this specification).
ESeum	eUICC	EUM	Administrative interface between the eUICC vendor (EUM) and the eUICC (out of scope of this specification).
ESci	CI	SM-DP+ SM-DS EUM	This interface is used by the SM-DP+, SM-DS and EUM to request a certificate and retrieve certificate revocation status. Any other relying party MAY retrieve certificate revocation status. The interface for Certificate Signing Request is defined in SGP.14 [45] section 5.1. The interface for CRL retrieval is defined in the present document, section 4.5.2.1.2 "Extension CRL Distribution Points".
ESdloa	DLOA Registrar	Management System	This interface is defined in GlobalPlatform DLOA [57] section 5.
ESdsps	SM-DS	push server	Interface between SM-DS and push server (out of scope of this specification).
ESps	push server	push client	Interface between push server and push client (out of scope of this specification).
ESpc	push client	LPAd	Interface between push client and LPAd (out of scope of this specification).

Table 1: Interfaces

2.4 eUICC Architecture

2.4.1 eUICC Overview

This section describes the internal high-level architecture of the eUICC. It should be noted that the eUICC architecture is very similar to that used in the GSMA Remote SIM Provisioning of Embedded UICC Technical specification [2]. Operator Profiles are stored inside Security Domains within the eUICC and are implemented using GlobalPlatform standards. These ensure that it is impossible for any Profile to access the applications or data of any other Profile stored on the eUICC. The same mechanism is currently in use within SIM cards to ensure payment applications are kept secure.

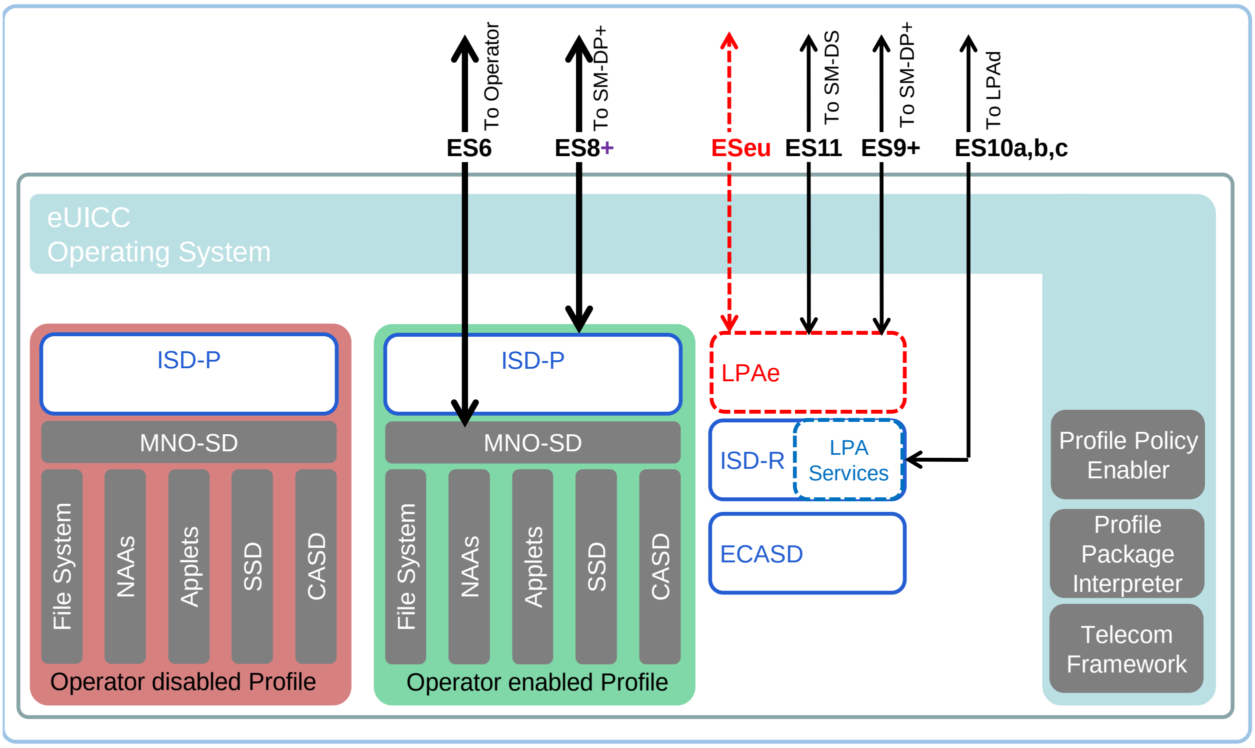


Figure 3: Schematic Representation of the eUICC

2.4.2 ECASD

The Embedded UICC Controlling Authority Security Domain (ECASD) is responsible for secure storage of credentials required to support the required Security Domains on the eUICC.

There SHALL be only one ECASD on an eUICC. The ECASD SHALL be installed and personalized by the EUM (eUICC Manufacturer) during the eUICC manufacturing. After eUICC manufacturing, the ECASD SHALL be in life-cycle state PERSONALIZED as defined in GlobalPlatform Card Specification [8] section 5.3.

The AID of the ECASD SHALL follow SGP.02 [2].

The ECASD SHALL contain:

- The eUICC's Private Key(s) (SK.EUICC.SIG) for creating digital signatures
- The eUICC's Certificate(s) for eUICC authentication (CERT.EUICC.SIG) containing the eUICC's public key(s) (PK.EUICC.SIG)
- The eSIM Certificate Issuer's (CI) RootCA Public Key(s) (PK.CI.SIG) for verifying off-card entities certificates (e.g., SM-DP+) and Certificate Revocation List (CRL). ECASD MAY contain several public keys belonging to the same eSIM CA or different eSIM CAs. Each PK.CI.SIG SHALL be stored with information coming from the CERT.CI.SIG the key is included in, at least:
  - eSIM Certificate Issuer OID
  - Subject Key Identifier: required to verify the Certificate chain of the off-card entity

- The Certificate(s) of the EUM (CERT.EUM.SIG), and, optionally, the Certificate(s) of the EUM SubCA (CERT.EUMSubCA.SIG)

The ECASD SHOULD also contain:

- eUICC Manufacturer's (EUMs) keyset for key/certificate renewal, which is used in one or more of the following functions:
    - Renew eUICC's Private Key(s) and Certificate(s)
    - Renew EUM Certificate(s) and, optionally, EUM SubCA Certificate(s)
    - Renew eSIM CA RootCA public key(s)
    - Add new eUICC Private Key(s), eUICC Certificate(s), EUM Certificate(s) and, optionally, EUM SubCA Certificate(s)
    - Remove eUICC's Certificate(s)
    - Remove EUM Certificate(s) and related eUICC's Certificate(s)
    - Add new eSIM CA RootCA Public Key(s)
    - Remove eSIM CA RootCA Public Key(s)
    - Generate a new Public/Private Key pair to support a new curve

The means by which the EUM SHOULD perform key/certificate renewal/generation/addition/removal is out of scope of this specification but, if provided, it SHALL be a GlobalPlatform [8] mechanism with a minimum security level corresponding to the security level of the key/Certificate on which the operation is performed (e.g., AES algorithm using a minimum key length of 128 bits for a renewal of ECC keys of 256 bits). The EUM MAY also revoke an eSIM CA RootCA Certificate on the eUICC (e.g., by deleting the related public key).

The ECASD SHALL provide the following services to the ISD-R:

- eUICC signature creation on material provided by an ISD-R
- Verification of the off-card entities certificates (e.g., SM-DP+), provided by an ISD-R, with the eSIM CA RootCA public key (PK.CI.SIG)

Personalisation of the ECASD SHALL be done in a certified 'GSMA SAS-UP environment' according to the SAS UP specification [77].

2.4.3 ISD-R

The ISD-R is responsible for the creation of new ISD-Ps and lifecycle management of all ISD-Ps.

There SHALL be only one ISD-R on an eUICC.

The ISD-R SHALL be installed and personalized by the EUM during eUICC manufacturing. The ISD-R SHALL be associated with itself. The ISD-R privileges SHALL be granted according to Annex A.

The ISD-R cannot be deleted or disabled.

2.4.4 ISD-P

The ISD-P is the on-card representative of the SM-DP+ and is a secure container (Security Domain) for the hosting of a Profile. The ISD-P is used for the Profile download and installation in collaboration with the Profile Package Interpreter for the decoding/interpretation of the received Profile Package.

An ISD-P hosts a unique Profile.

No component outside the ISD-P SHALL have visibility or access to any Profile Component with the exception of the ISD-R, which SHALL have access to Profile Metadata.

A Profile Component SHALL NOT have any visibility of, or access to, components outside its ISD-P. An ISD-P SHALL NOT have any visibility of, or access to, any other ISD-P.

Deletion of a Profile SHALL remove the containing ISD-P and all Profile Components of the Profile.

2.4.5 Profile

A Profile consists of Profile Components:

- One MNO-SD
- Supplementary Security Domains (SSD) and a CASD
- Applets
- Applications, e.g., NFC applications
- NAAs
- Other elements of the File System
- Profile Metadata, including Profile Policy Rules

The MNO-SD is the on-card representative of the Operator. It contains the Operator’s Over-The-Air (OTA) keys and provides a secure OTA channel.

All Security Domains of a Profile SHALL be located in the hierarchy of the MNO-SD or an SD extradited to itself.

The behaviour of an eUICC with an Enabled Profile SHALL be equivalent to a UICC. This applies especially for the NAAs and applets contained in the Profile.

When a Profile is Disabled, the eUICC SHALL ensure that:

- Remote management of any Profile Component is not possible via the ES6 interface.
- The file system within the Profile cannot be selected by the Device or any application on the eUICC.
- The applications (including NAAs and Security Domains) within the Profile cannot be selected, triggered or individually deleted.
- For an eUICC compliant with M4M [58], no M4M Virtual Card inside that Profile is visible nor accessible through any interface.

2.4.5.1 Operational Profile

An Operational Profile SHALL have its Profile Class set to 'operational' in its Profile Metadata to indicate to the LPA and the eUICC that it SHALL be handled in the manner that is appropriate for an Operational Profile.

2.4.5.2 Provisioning Profile

A Provisioning Profile SHALL have its Profile Class set to 'provisioning' in its Profile Metadata to indicate to the LPA and the eUICC that it SHALL be handled in the manner that is appropriate for a Provisioning Profile. In every other respect, a Provisioning Profile SHALL have the same format structure as any other Profile.

Provisioning Profile Impact on LUI

Provisioning Profiles and their associated Profile Metadata SHALL NOT be visible to the End User in the LUI. As a result, Provisioning Profiles SHALL NOT be selectable by the End User nor deletable through any End User action, including eUICC Memory Reset.

Provisioning Profile and Operational Profile Policies

Provisioning Profiles SHALL still be usable, even if the currently enabled Operational Profile is subject to Profile Policy Rule 'ppr1'. In the case where a Provisioning Profile needs to be enabled, the LPA SHALL directly enable the Provisioning Profile, without first explicitly disabling the currently enabled Operational Profile; the eUICC SHALL allow this operation and implicitly disable the currently enabled Operational Profile regardless of the Profile Policy Rule.

2.4.5.3 Test Profile

An eUICC MAY support Test Profiles.

A Test Profile SHALL have its Profile Class set to 'test' in its Profile Metadata to indicate to the LPA and the eUICC that it SHALL be handled in the manner that is appropriate for a Test Profile. To ensure that a Test Profile is not used as an Operational Profile, the value of its key(s) for network authentication SHALL comply with one of the following:

1. All bits except the lowest 32 bits set to zero.
2. '00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F' (default K value of Test USIM as defined in Section 8.2 of 3GPP TS 34.108 [62]).
3. Any arbitrary value, if the network authentication algorithm is the Test Algorithm as defined in Section 8.1.2 of 3GPP TS 34.108 [62] or the IMSI value complies with the Test USIM IMSI defined in Section 8.3.2.2 of 3GPP TS 34.108 [62].

When a Test Profile is downloaded, the eUICC SHALL accept keys compliant with (1) and SHOULD accept keys compliant with (2) or (3).

Preloaded Test Profiles SHALL comply with one of these conditions.

In every other respect, a Test Profile SHALL have the same format structure as any other Profile.

NOTE: A live commercial network would never use a key with so little entropy.

Test Profiles and their associated Profile Metadata SHALL be visible in the LUI when the Device is in Device Test Mode.

The only indication the eUICC receives about the Device entering Test Mode is the initial enabling of a Test Profile. Other Test Profiles may be enabled subsequently. The explicit disabling of the current Test Profile or the enabling of a non-Test Profile indicates to the eUICC that Device Test Mode was left. If an Operational Profile was in Enabled state before the (first) Test Profile was enabled, the eUICC will ensure that this Operational Profile is re-enabled when the Test Profile is disabled. This ensures that the Profile Policy Rule "Disabling not allowed" is handled consistently. (See sections 3.2.1 and 3.2.2 for details.)

Even though Profile Policy Rules do not apply for MEP, the same re-enabling mechanism is specified for MEP. While a Test Profile is enabled on an eSIM Port, the LPAd SHALL not enable the temporarily disabled Operational Profile on a different eSIM Port. The behaviour of the eUICC for such attempts or for any other exceptional action by the LPAd is neither specified nor are such cases covered in the procedures in sections 3.2.1 and 3.2.2.

NOTE: While the LPAd can receive an implementation specific notification when the Device enters/leaves Test Mode, such a mechanism is not available for the LPAe. To avoid extra messages, the LPAe MAY implement a protected option in its user interface for activating Test Mode.

Even if configured in its Metadata, the eUICC will never generate Notifications related to the Enabling or Disabling of a Test Profile. In addition, no Notifications will be generated for an Operational Profile when switching to or from a Test Profile.

NOTE: The GSMA Generic Test Profile [85] is an example of a Test Profile that supports a wide range of development, certification, and repair/refurbishment testing activities.

2.4.6 Telecom Framework

The Telecom Framework is an Operating System service that provides standardised network authentication algorithms to the NAAs hosted in the ISD-Ps. Furthermore, it provides the capabilities to configure the algorithm with the necessary parameters in the Enabled Profile.

2.4.7 Profile Package Interpreter

The Profile Package Interpreter is an eUICC Operating System service that translates the Profile Package data as defined in eUICC Profile Package Specification [5] into an installed Profile using the specific internal format of the target eUICC.

NOTE: According to the eUICC Profile Package Specification [5], the eUICC also attempts to process Profile Packages indicating a different minor version.

2.4.8 LPAe

The LPAe is a functional element that provides the LPDe, LDSe and LUIe features. These features are similar to the features of an LPAd.

LPAe is optional.

The technical implementation of LPAe is up to the EUM. For example, the LPAe MAY be a feature of the ISD-R.

2.4.9 LPA Services

This role provides the necessary access to the services and data required by LPAd functions. These services include:

- Provide the address(es) of the Root SM-DS(s) and the Default SM-DP+, if configured
- Transfer Bound Profile Package from the LPAd to the ISD-P
- Provide list of installed Profiles and their Profile Metadata
- Retrieve EID
- Provide Local Profile Management Operations
- Provide Remote Profile Management
- Provide authentication and integrity verification for Event Retrieval from an SM-DS
- Provide pending Notifications

The conditions under which LPA Services SHALL be present are described in section 2.1.

2.4.10 Hardware Characteristics of the eUICC

The following requirements apply:

- The eUICC SHALL be based on Tamper Resistant Element.
- The eUICC SHALL be either a Discrete eUICC or an Integrated eUICC.
- A Discrete eUICC MAY either be removable or non-removable. A removable eUICC SHALL be packaged in a form factor specified in ETSI TS 102 221 [6].

2.4.11 Platform Characteristics of the eUICC

In compliance with SGP.21, the following requirements apply:

- The eUICC SHALL support SHA-1.
- The eUICC SHALL support TUAK [51].
- The eUICC SHALL support Milenage [52].
- All cryptographic functions SHALL be implemented in a tamper-resistant way and SHALL resist side-channel attacks.

2.4.11.1 Java Card packages

An eUICC supporting Java Card^{TM} SHALL support the Java Packages listed below. The implementation of each Package SHALL as a minimum be according to the given Package version and Specification version.

Package	Description	Package Version	Spec Version
java.* (java.lang, java.io)	**java.io** defines a subset of the java.io package in the standard Java programming language.	1.0	Java Card^{TM} 3.0.5 Classic Edition
	java.lang Provides classes that are fundamental to the design of the Java	1.0	

	Card technology subset of the Java programming language.		
javacard.* (javacard.framework, javacard.security)	**javacard.framework** Provides a framework of classes and interfaces for building, communicating with and working with Java Card™ technology based applets.	1.6	Java Card™ 3.0.5 Classic Edition
	javacard.security Provides classes and interfaces that contain publicly available functionality for implementing a security and cryptography framework on the Java Card™ platform	1.6	
javacardx.* (javacardx.crypto)	**javacardx.crypto** Extension package that contains functionality, which may be subject to export controls, for implementing a security and cryptography framework on the Java Card™ platform.	1.6	Java Card™ 3.0.5 Classic Edition
uicc.* (uicc.access, uicc.system, uicc.toolkit)	**uicc.access** This package provides the means to the applets for accessing the UICC file system defined in the TS 102 221 [6] specification.	1.2	TS 102 241 [53] Rel-9
	uicc.system This package provides the means to the applets for accessing system wide services of the UICC platform.	1.0	
	uicc.toolkit This package provides the means for the toolkit applets to register to the events of the common application toolkit (CAT) framework, to handle TLV information and to send proactive commands according to the ETSI TS 102 223 [31] specification.	1.6	
uicc.usim.* (uicc.usim.access, uicc.usim.toolkit)	**uicc.usim.access** This package provides the means for applets to get access to the files defined in the USIM, SIM and ISIM specification.	1.0	3GPP TS 31.130 Rel-9
	uicc.usim.toolkit This package provides the means for the toolkit applets to register to the events defined in the USAT and STK specification to handle TLV information and to send proactive command according to the ETSI 3GPP TS 31.111 and 3GPP TS 51.014 specification.	1.8	
org.globalplatform	Provides a framework of classes and interfaces related to core services defined for smart cards based on GlobalPlatform specifications	1.6	GP 2.3 [8]

Table 2: Java Card Packages

The following additional Java Card packages SHALL be supported by an eUICC supporting NFC:

Package	Description	Package Version	Spec Version
javacardx.* (javacardx.external, javacardx.framework.tlv, javacardx.framework.util.intx)	**javacardx.external** Extension package that provides mechanisms to access memory subsystems which are not directly addressable by the Java Card™ runtime environment (Java Card™ RE) on the Java Card™ platform.	1.0	Java Card™ 3.0.5 Classic Edition
	javacardx.framework.tlv Extension package that contains functionality, for managing storage for BER TLV formatted data, based on the ASN.1 BER encoding rules of ISO/IEC 8825-1:2002, as well as parsing and editing BER TLV formatted data in I/O buffers.	1.0	
	javacardx.framework.util.intx Extension package that contains common utility functions for using `int` components.	1.0	
uicc.hci.* (uicc.hci.framework, uicc.hci.services.cardemulation, uicc.hci.services.connectivity, uicc.hci.services.readermode)	**uicc.hci.framework** This package defines the basic interfaces for the HCI protocol.	1.1	TS 102 705 Rel-9
	uicc.hci.services.cardemulation This package defines the interfaces for the card emulation mode of the HCI protocol.	1.0	
	uicc.hci.services.connectivity This package defines the interfaces for the functionality of the connectivity gate defined in the HCI protocol.	1.0	
	uicc.hci.services.readermode This package defines the interfaces for the reader emulation mode of the HCI protocol.	1.0	
org.globalplatform.contactless	Provides a framework of classes and interfaces related to contactless services defined for smart cards based on GlobalPlatform specifications	1.3	Amd C 1.2 [10]

Table 3: Java Card Packages

2.4.12 Profile Policy Enabler

The Profile Policy Enabler is described in detail in section 2.9.3.

2.4.13 eUICC OS Update

An eUICC SHOULD support a mechanism for eUICC OS Update. If such a mechanism is supported, it SHALL be secure. The eUICC OS Update capability allows the OS manufacturer to correct errors in existing features on the eUICC by updating the eUICC OS.

The process and mechanisms used to perform an eUICC OS Update are Device manufacturer and EUM implementation-specific and therefore out of scope of this version of the specification. However, the eUICC OS Update mechanism SHALL fulfil the requirements defined in SGP.21 [4].

This operation MAY affect information contained within `euiccInfo1` and `euiccInfo2`.

Following any eUICC OS Update, the eUICC SHALL remain compliant according to the RSP Certification process as specified in SGP.24 [64].

2.4a ASN.1

The description of some data objects in this specification is based on ASN.1 specified in ITU-T X.680 [49] and encoded in TLV structures using DER (Distinguished Encoding Rule) encoding as specified in ITU-T X.690 [50]. This provides a flexible description of those data objects.

The Remote SIM Provisioning format is defined in a single, self-contained, ASN.1 definition module called `RSPDefinitions`, with an ISO Object Identifier in the GSMA namespace.

```
-- ASN1START
RSPDefinitions {joint-iso-itu-t(2) international-organizations(23) gsma(146) rsp(1)
spec-version(1) version-three(3)}
DEFINITIONS
AUTOMATIC TAGS
EXTENSIBILITY IMPLIED ::=
BEGIN
-- ASN1STOP
```

Two encoding/decoding attributes are defined:

- AUTOMATIC TAGS means that the tags are defined automatically using the encoding rules unless a tag notation is present in a data object format definition.
- EXTENSIBILITY IMPLIED means that types MAY have elements that are not defined in this specification. This means that decoders SHALL be able to handle values with unspecified tags, either by processing them if they know their value content, or ignoring them silently (without reporting an error) if they do not know them. This is useful when processing data definitions from a newer specification and to handle proprietary tag values.

As the eUICC cannot implement an off-the-shelf standard decoder, the requirement on extensibility SHALL NOT apply to the eUICC. In some cases the eUICC is even mandated to report undefined tags, see e.g., sections 3.1.5 and 5.7.6.

The following general rules SHALL apply for the compliance of entities according to this specification:

- A sender of data defined in this specification SHALL comply with the exact definition of the data structure.
- A receiver is only mandated to check incoming data where such a check is explicitly required by this specification. Otherwise, a receiver is free to tolerate deviations (e.g., strings longer than the defined maximum) or reject them. If the receiver does not understand a field, the receiver SHALL silently ignore it without reporting an error (i.e., SHALL NOT parse or process the value content). If the receiver is requested to sign or

verify the signature of a data object containing those unknown field(s), the receiver SHALL perform the request on the entire data object.

NOTE: Unless otherwise specified in this specification, an optional field should be present only when the field has a value, i.e., it should not be present with empty content.

This specification relies on X.509 data objects imported from libraries defined in RFC 5280 [17]:

```
-- ASN1START
IMPORTS Certificate, CertificateList, Time FROM PKIX1Explicit88 {iso(1) identified-
organization(3) dod(6) internet(1) security(5) mechanisms(5) pkix(7) id-mod(0) id-
pkix1-explicit(18)}
SubjectKeyIdentifier FROM PKIX1Implicit88 {iso(1) identified-organization(3) dod(6)
internet(1) security(5) mechanisms(5) pkix(7) id-mod(0) id-pkix1-implicit(19)}
UICCCapability FROM PEDefinitions {joint-iso-itu-t(2) international-
organizations(23) tca(143) euicc-profile(1) spec-version(1) version-three(3)};
-- The UICCCapability import module version is defined in section 4.3
-- ASN1STOP
```

2.4a.1 Common ASN.1 data types

NOTE: Other common data types may be added here in future versions.

2.4a.1.0 Data type: Miscellaneous

The following constant, types and objects are used in this specification.

```
-- ASN1START
id-rsp OBJECT IDENTIFIER ::= {joint-iso-itu-t(2) international-organizations(23)
gsma(146) rsp(1)}

-- Basic types, for size constraints
Octet1 ::= OCTET STRING(SIZE(1))
Octet4 ::= OCTET STRING (SIZE(4))
Octet8 ::= OCTET STRING (SIZE(8))
Octet16 ::= OCTET STRING (SIZE(16))
OctetTo16 ::= OCTET STRING (SIZE(1..16))
Octet32 ::= OCTET STRING (SIZE(32))

VersionType ::= OCTET STRING(SIZE(3)) -- major/minor/revision version are coded as
binary value on byte 1/2/3, e.g., '02 00 0C' for v2.0.12.
-- If revision is not used (e.g., v2.1), byte 3 SHALL be set to '00'.
Iccid ::= [APPLICATION 26] OCTET STRING (SIZE(10)) -- ICCID as coded in EFiccid,
corresponding tag is '5A'
RemoteOpId ::= [2] INTEGER {installBoundProfilePackage(1)}
TransactionId ::= OCTET STRING (SIZE(1..16))

-- Definition of OIDs
id-rsp-cert-objects OBJECT IDENTIFIER ::= {id-rsp 2}
   -- value 0 in id-rsp-cert-objects was assigned in SGP.22 v2.x
   -- #SupportedOnlyBeforeV3.0.0#

id-rspRole OBJECT IDENTIFIER ::= {id-rsp-cert-objects 1}

-- Definition of OIDs for role identification in certificates
id-rspRole-ci OBJECT IDENTIFIER ::= {id-rspRole 0}
id-rspRole-ciSubCa OBJECT IDENTIFIER ::= {id-rspRole-ci 0}

id-rspRole-eum OBJECT IDENTIFIER ::= {id-rspRole-ciSubCa 0}
id-rspRole-eumSubCa OBJECT IDENTIFIER ::= {id-rspRole-eum 0}
id-rspRole-euicc OBJECT IDENTIFIER ::= {id-rspRole-eumSubCa 0}

id-rspRole-dpSubCa OBJECT IDENTIFIER ::= {id-rspRole-ciSubCa 1}
id-rspRole-dp-tls OBJECT IDENTIFIER ::= {id-rspRole-dpSubCa 0}
```

```
id-rspRole-dp-auth OBJECT IDENTIFIER ::= {id-rspRole-dpSubCa 1}
id-rspRole-dp-pb OBJECT IDENTIFIER ::= {id-rspRole-dpSubCa 2}

id-rspRole-dsSubCa OBJECT IDENTIFIER ::= {id-rspRole-ciSubCa 2}
id-rspRole-ds-tls OBJECT IDENTIFIER ::= {id-rspRole-dsSubCa 0}
id-rspRole-ds-auth OBJECT IDENTIFIER ::= {id-rspRole-dsSubCa 1}

-- The following OIDs are used in Variant O and OO Certificates
id-rspRole-euicc-v2 OBJECT IDENTIFIER ::= {id-rspRole 1}
id-rspRole-eum-v2 OBJECT IDENTIFIER ::= {id-rspRole 2}
id-rspRole-dp-tls-v2 OBJECT IDENTIFIER ::= {id-rspRole 3}
id-rspRole-dp-auth-v2 OBJECT IDENTIFIER ::= {id-rspRole 4}
id-rspRole-dp-pb-v2 OBJECT IDENTIFIER ::= {id-rspRole 5}
id-rspRole-ds-tls-v2 OBJECT IDENTIFIER ::= {id-rspRole 6}
id-rspRole-ds-auth-v2 OBJECT IDENTIFIER ::= {id-rspRole 7}

-- Definition of OIDs for RSP-specific extensions in Certificates
-- #SupportedFromV3.0.0#
id-rsp-extensions OBJECT IDENTIFIER ::= {id-rsp-cert-objects 2}
id-rsp-extension-permitted-eins OBJECT IDENTIFIER ::= { id-rsp-extensions 0}

-- The extnValue field of the id-rsp-extension-permitted-eins extension SHALL be of
type PermittedEins:
PermittedEins ::= SEQUENCE OF PrintableString

-- ASN1STOP
```

2.4a.1.1 Data type: PprIds

The data type `PprIds` codes the identifiers for Profile Policy Rules defined in this document.

```
-- ASN1START
PprIds ::= BIT STRING {-- Definition of Profile Policy Rules identifiers
   pprUpdateControl(0), -- defines how to update PPRs via ES6
   ppr1(1), -- Indicator for PPR1 'Disabling of this Profile is not allowed'
   ppr2(2) -- Indicator for PPR2 'Deletion of this Profile is not allowed'
}
-- ASN1STOP
```

For pprX: a bit set to '1' indicates that the corresponding PPR is set.

Further versions of this specification MAY introduce new Profile Policy Rule identifiers

2.4a.1.2 Data type: OperatorId

The data type `OperatorId` contains the identification of an Operator. This type is used to identify a Profile Owner.

```
-- ASN1START
OperatorId ::= SEQUENCE {
   mccMnc OCTET STRING (SIZE(3)), -- MCC&MNC coded as 3GPP TS 24.008
   gid1 OCTET STRING OPTIONAL, -- referring to content of EF GID1 (file identifier
'6F3E') in 3GPP TS 31.102 [54]
   gid2 OCTET STRING OPTIONAL  -- referring to content of EF GID2 (file identifier
'6F3F') in 3GPP TS 31.102 [54]
}
-- ASN1STOP
```

Coding of mccMnc: contains MCC (3 digits) and MNC (2 or 3 digits) on 3 bytes coded as in 3GPP TS 24.008 [32]. For instance, an Operator identified by 246 for the MCC and 81 for the MNC will be coded as bytes 1 to 3: '42' 'F6' '18'.

Coding of `gid1` and `gid2`: both are optional. If provided, their content SHALL be coded as defined in 3GPP TS 31.102 [54].

In case the Profile contains multiple USIM applications, the `OperatorId` SHALL reflect the values of one of the USIM applications.

NOTE: Additional mechanism for identifying MVNO/Service Providers is for further study.

2.4a.1.3 Data type: RpmConfiguration

The data type `RpmConfiguration` defines the configuration on RPM support of a Profile.

```
-- ASN1START
RpmConfiguration ::= SEQUENCE { -- #SupportedForRpmV3.0.0#
   managingDpList [0] SEQUENCE OF SEQUENCE {
      managingDpOid [0] OBJECT IDENTIFIER, -- Managing SM-DP+ OID
      rpmType [1] RpmType OPTIONAL,
      tagList [APPLICATION 28] OCTET STRING OPTIONAL
   },
   pollingAddress [1] UTF8String OPTIONAL, -- Tag '81'
   allowedCiPKId [2] SubjectKeyIdentifier OPTIONAL, -- eSIM CA RootCA PKID that is
allowed for managing SM-DP+s
   profileOwnerOid [3] OBJECT IDENTIFIER
}

RpmType ::= BIT STRING{
   enable(0), disable(1), delete(2), listProfileInfo(3), contactPcmp(4)
}
-- ASN1STOP
```

`managingDpList` contains a list of Managing SM-DP+s identified by their OIDs. Multiple bits can be set in `rpmType`, where a bit set to '1' indicates that the Managing SM-DP+ is allowed to perform the corresponding RPM Command, except for 'Update Metadata'. If a Managing SM-DP+ is allowed to perform 'Update Metadata', then the list of data objects that the Managing SM-DP+ can update SHALL be specified in the `tagList` (defined in section 5.7.15).

`pollingAddress` contains the address of either SM-DP+ or SM-DS as an FQDN.

`allowedCiPKId` specifies the only eSIM CA RootCA public key that is allowed for authentication of the Managing SM-DP+(s) specified by `managingDpList`. If `allowedCiPKId` is absent then any eSIM CA RootCA public key may be used.

`profileOwnerOid` contains the OID of the Profile Owner.

2.4a.1.4 Data type: LocalisedTextMessage

The data type `LocalisedTextMessage` contains a text string and the language associated with it.

```
-- ASN1START
LocalisedTextMessage ::= SEQUENCE { -- #SupportedFromV3.0.0#
   languageTag UTF8String DEFAULT "en", -- language tag as defined by RFC 5646
   message UTF8String
}
-- ASN1STOP
```

2.4a.1.5 Data type: LprConfiguration

The data type `LprConfiguration` defines the configuration for the LPRd.

```
-- ASN1START
LprConfiguration ::= SEQUENCE { -- #SupportedForLpaProxyV3.0.0#
   pcmpAddress [0] UTF8String,
   dpiEnable [1] UTF8String OPTIONAL,
   triggerLprOnEnableProfile [2] NULL OPTIONAL
}
-- ASN1STOP
```

`pcmpAddress` contains the FQDN of the PCMP server.

`dpiEnable` contains the DPI which if present is appended to the `pcmpAddress` if the LPR is triggered upon local enabling.

If `triggerLprOnEnableProfile` is present, then the LPA SHALL trigger the LPR session after local enabling the Profile.

2.4a.1.6 Data type: CertificateChain

The data type `CertificateChain` contains a chain of Certificates.

```
-- ASN1START
CertificateChain ::= SEQUENCE OF Certificate -- #SupportedFromV3.0.0#
-- ASN1STOP
```

Each subsequent Certificate in the list SHALL certify the one preceding it. The eSIM CA RootCA Certificate SHALL be omitted.

2.4a.1.7 Data type: EnterpriseConfiguration

The data type `EnterpriseConfiguration` defines the configuration of an Enterprise Profile.

```
-- ASN1START
EnterpriseConfiguration ::= SEQUENCE { -- #SupportedForEnterpriseV3.0.0#
   enterpriseOid [0] OBJECT IDENTIFIER,
   enterpriseName [1] UTF8String (SIZE(0..64)),
   enterpriseRules [2] SEQUENCE {
      enterpriseRuleBits [0] BIT STRING {
         referenceEnterpriseRule (0),
         priorityEnterpriseProfile (1),
         onlyEnterpriseProfilesCanBeInstalled (2)
      },
      numberOfNonEnterpriseProfiles [1] INTEGER -- that can be Enabled
   } OPTIONAL
}
-- ASN1STOP
```

`enterpriseOid` contains the OID of the Enterprise. This field SHALL NOT be modifiable after the installation of an Enterprise Profile.

`enterpriseName` contains the name of the Enterprise.

`enterpriseRules` defines the Enterprise Rules of the Enterprise Profile as follows.

- `referenceEnterpriseRule` indicates that the Enterprise Rule of this Enterprise Profile is the Reference Enterprise Rule. The next bits and `numberOfNonEnterpriseProfiles` described hereunder affect the Enterprise Capable Device only when this bit is set. The eUICC of an Enterprise Owned Device SHALL ensure that at most one Enterprise Profile has this bit set among the installed Enterprise Profiles.
- `priorityEnterpriseProfile` indicates that the End User can enable only this Enterprise Profile or has to enable this Enterprise Profile before being able to enable any other Profile.
- `onlyEnterpriseProfilesCanBeInstalled` indicates that the End User can install only Enterprise Profiles.

- numberOfNonEnterpriseProfiles defines the maximum number of non-Enterprise Profiles that can be Enabled on an eUICC.

NOTE: This parameter may even be higher than the supported number of Profiles being Enabled in parallel. Dependent on the latter and the parameter priorityEnterpriseProfile, the actual number of non-Enterprise Profiles that can be Enabled may be lower.

2.4a.1.8 Data type: VendorSpecificExtension

The data type VendorSpecificExtension is used to send additional extensions that are not defined in this specification.

```
-- ASN1START
OPENTYPE ::= CLASS {
   &typeId OBJECT IDENTIFIER,
   &Type
}

VendorSpecificExtension ::= SEQUENCE OF SEQUENCE { -- #SupportedFromV2.4.0#
   vendorOid [0] OPENTYPE.&typeId, -- OID of the vendor who defined this specific
extension
   vendorSpecificData [1] OPENTYPE.&Type
}
-- ASN1STOP
```

2.4a.1.9 Data type: DeviceChangeConfiguration

The data type DeviceChangeConfiguration defines the configurations for the LPAd to support the Device Change of the Profile.

```
-- ASN1START
DeviceChangeConfiguration ::= CHOICE { -- #SupportedForDcV3.0.0#
   requestToDp [0] SEQUENCE {
      smdpAddressForDc UTF8String, -- SM-DP+ address that processes the Device
Change
      allowedCiPKId SubjectKeyIdentifier OPTIONAL, -- PKID allowed for the SM-DP+
address that processes the Device Change
      eidRequired NULL OPTIONAL, -- the EID of the new Device is required
      tacRequired NULL OPTIONAL -- the TAC of the new Device is required
   },
   usingStoredAc [1] SEQUENCE {
      activationCodeForDc UTF8String (SIZE(0..255)), -- Activation Code for Device
Change of this Profile
      deleteOldProfile NULL OPTIONAL -- deletion of this Profile is required before
providing the Activation code to the new Device
   }
}
-- ASN1STOP
```

The requestToDp consists of the following data objects for the LPAd to support the Device Change of the Profile by requesting to the SM-DP+:

- smdpAddressForDc contains the SM-DP+ address that SHALL process the Device Change of the Profile.
- allowedCiPKId specifies the only eSIM CA RootCA public key that is allowed for authentication of the SM-DP+ in the smdpAddressForDc. If allowedCiPKId is absent, then any eSIM CA RootCA public key may be used.

- `eidRequired` indicates that the EID of the new Device is required for the Device Change of the Profile. If `eidRequired` is present, then the LPA SHALL retrieve the EID from the new Device.
- `tacRequired` indicates that the TAC of the new Device is required for the Device Change of the Profile. If `tacRequired` is present, then the LPA SHALL retrieve the TAC from the new Device.

The `usingStoredAc` consists of the following data objects for the LPAd to support the Device Change of the Profile by using the Activation Code stored in the Profile Metadata:

- `activationCodeForDc` contains the Activation Code to be provided to the new Device to download the Profile for the Device Change.
- `deleteOldProfile` indicates that the Profile has to be deleted before the Activation Code is made available for the new Device.

2.4a.2 ASN.1 data type UTF8String

The size limits for UTF-8 strings apply to the number of UTF-8 characters, each coded on 1 to 4 bytes, see ISO/IEC 10646 [59]. Thus the length of the TLV object counted in bytes can be up to 4 times the number of characters.

The eUICC is not mandated to analyse the character structure of UTF-8 strings provided in a command. The LPA and a Managing SM-DP+ SHOULD be able to receive a string with 4 times as many UTF-8 characters as the ASN.1 size limit if such a string was previously stored in the eUICC.

2.5 Profile Protection and Delivery

This section describes how an Operator's Profile is protected within a Profile Package prior to being downloaded to the eUICC. This also applies when the Profile Package is protected during transmission between Roles within the system.

2.5.1 Profile Package Types Overview

From generation to download, the Profile Package will take different formats. This specification uses the following terms:

- Unprotected Profile Package (UPP): Raw eUICC Profile Package TLV sequence.
- Protected Profile Package (PPP): Segmented and protected in BSP payload TLVs.
- Bound Profile Package (BPP): Prepended with session key agreement info, key replacement package, ISD-P creation and configuration info.
- Segmented Bound Profile Package (SBPP): BPP segmented into STORE DATA APDU script for loading into eUICC. This step is performed by the LPD when LPD is in the Device.

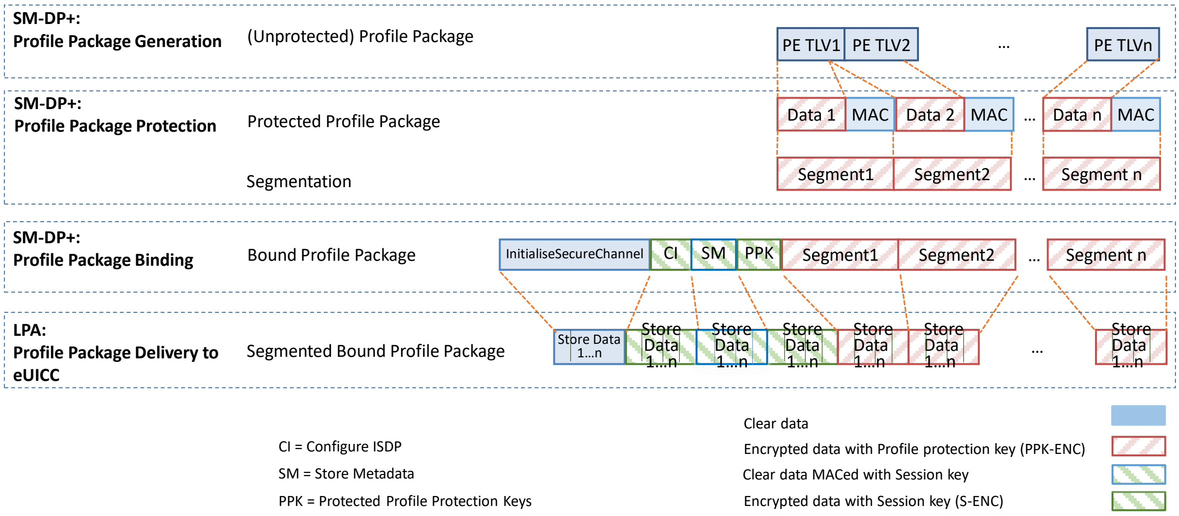


Figure 4: Profile Package stage Description

The above diagram describes the case where Profile Package is protected with keys different from the session keys established during the key agreement with the eUICC (S-ENC, S-MAC). It MAY also be possible to have a Profile Package protected with the session keys; in that case the 'Profile Protection Keys' block SHALL NOT be present.

2.5.2 Unprotected Profile Package

The Unprotected Profile Package (UPP) is generated by the SM-DP+, within the Profile Package Generation function. The Profile Package Generation takes as input the profile specification established with the Operator and input data provided by the Operator. The processes of profile specification and input data acquisition are out of scope of this specification.

The Unprotected Profile Package consists of a sequence of Profile Element (PE) TLVs according to the eUICC Profile Package specification [5].

2.5.3 Protected Profile Package

The Protected Profile Package (PPP) is generated by the SM-DP+, within the Profile Package Protection function.

The PPP SHALL be protected by the BSP. Command TLV encryption and MACing follows section 2.6.4.4. During this step the internal UPP structure is not considered, and rather seen as a unique block of data. That block of data is split into segments of a maximum size of 1020 bytes (including the tag, length field and MAC). The eUICC SHALL support receiving data segments of at least up to this size.

NOTE: When the BSP uses AES-CBC-128/AES-CMAC-128 or SM4-CBC/SM4-CMAC, the following applies: From the 1020 bytes of each data segment, only 1008 bytes are usable for payload (deducted the 1 byte tag, 3 bytes length field and 8 bytes MAC). Considering the necessary padding during encryption (16 bytes length block encryption and necessary '80' byte padding), then each data segment can only contain 1007 bytes of the PPP data block.

Profile protection SHALL be performed using either:

- Session keys (S-ENC, S-MAC) resulting from the key agreement with eUICC.
- Or random keys per Profile (denoted PPK-ENC and PPK-MAC), generated by the SM-DP+.

The SM-DP+ SHALL support the random key mode and SHOULD support Session keys. It is the SM-DP+ choice (in line with the agreement with the Operator) as to which mode is selected.

The eUICC SHALL support both modes.

The random key mode allows performing Profile Package Protection in advance without having any eUICC knowledge. It may help to provide a better SM-DP+ scalability. If this mode is selected by the SM-DP+, the initial MAC chaining value to be used for the first segment of the PPP is provided together with the random keys (section 5.5.4) and the data block counter for ICV calculation is reset to its initial state (i.e., the value on 16 bytes is '00…01'). Otherwise, the MAC chaining method defined in section 2.6.4 SHALL be applied (i.e., the output of the previous segment SHALL be used as MAC chaining value).

S-ENC, S-MAC, PPK-ENC and PPK-MAC SHALL follow the key length defined in section 2.6.5.

Each data segment of the PPP is identified by the tag '86'.

In case the random key mode is used, the PPP is not bound to any particular eUICC or ISD-P AID value at this stage.

Session keys and, if used, the random keys SHALL only be used in the Profile download process. They SHALL be deleted on the eUICC latest at the end of the process.

The process of Profile creation is out of scope of this specification; however, the Operator MAY request the SM-DP+ to create multiple Profiles in advance. In this case the SM-DP+ SHALL create the Profiles in bulk, protect them using the random key mode, and store the resulting PPPs for later use.

2.5.4 Bound Profile Package

The Bound Profile Package (BPP) is generated by the SM-DP+, within the Profile Package Binding function. The purpose of this operation is to link a Protected Profile Package to a particular eUICC. This is done within a key agreement between the eUICC and the SM-DP+. See download and installation procedure (section 3.1.3).

The BPP comprises a sequence of TLV commands (in this order):

- TLV command for Key agreement in clear.
- Set of BSP payload TLVs (tag '87') containing TLV commands for ConfigureISDP
- Set of BSP payload TLVs (tag '88') containing TLV command for StoreMetadata
- Set of optional BSP payload TLVs (tag '87') containing TLV command for 'Profile Protection Keys'
- Followed by the BSP payload TLVs (tag '86') of the PPP

The encryption counter for ICV calculation is incremented each time a TLV with tag '86', '87' or '88' is received.

The data structure of the Bound Profile Package is as follows:

```
-- ASN1START
BoundProfilePackage ::= [54] SEQUENCE { -- Tag 'BF36'
   initialiseSecureChannelRequest [35] InitialiseSecureChannelRequest, -- Tag
'BF23'
   firstSequenceOf87 [0] SEQUENCE OF [7] OCTET STRING, -- sequence of '87' TLVs
   sequenceOf88 [1] SEQUENCE OF [8] OCTET STRING, -- sequence of '88' TLVs
   secondSequenceOf87 [2] SEQUENCE OF [7] OCTET STRING OPTIONAL, -- sequence of
'87' TLVs
   sequenceOf86 [3] SEQUENCE OF [6] OCTET STRING -- sequence of '86' TLVs
}
-- ASN1STOP
```

The following table describes the various sequences of '86', '87' and '88' TLV

Tag	Length	Value Description			MOC
'A0'	Var.	firstSequenceOf87			M
		'87'	Var.	BSP segment containing ConfigureISDP, protected with session keys resulting from the key agreement (S-ENC, S-CMAC) (section 2.6.4) Content: TLV for "ES8+.ConfigureISDP" function (section 5.5.2)	M
'A1'	Var	sequenceOf88			M
		'88'	Var.	BSP segment containing StoreMetadata, MAC protected with session keys resulting from the key agreement (S-CMAC) (See section 2.6.4) (i.e., not encrypted). Content: TLV for "ES8+.StoreMetadata" function (section 5.5.3)	M
		'88'	Var.	Additional BSP segment(s) containing the remainder of StoreMetadata, only present if one '88' TLV is not able to contain the whole data structure.	C
'A2'	Var.	secondSequenceOf87 SHALL be absent if no content			C
		'87'	Var.	BSP segment containing the Profile Protection Keys, protected with session keys resulting from the key agreement (S-ENC, S-CMAC) (section 2.6.4). Content: TLV for "ES8+.ReplaceSessionKeys" function (section 5.5.4)	O
'A3'	Var.	sequenceOf86			M
		'86'	Var.	BSP payload, segment b1 protected with Profile Protection Keys (PPK-ENC, PPK-MAC) or with session keys resulting from the key agreement (S-ENC, S-CMAC) (section 2.6.4).	M
		'86'	Var.	Subsequent BSP payload, segment b2…bn	O

Table 4: Profile Installation sequences of TLV

2.5.4.1 Description of 'InitialiseSecureChannel' Block

This block comprises the TLVs for opening a remote personalisation session with eUICC, including key agreement.

These TLVs, part of the function "ES8+.InitialiseSecureChannel", are listed and described in section 5.5.1. These TLVs SHALL NOT be encrypted. Integrity and authenticity are ensured by the signatures.

The execution of this function by the eUICC will result in the generation of the BSP session keys, denoted S-ENC, S-MAC and initial MAC chaining value, that will be used by the SM-DP+ to protect subsequent TLVs.

2.5.4.2 Description of 'ConfigureISDP' Block

This block comprises one TLV for ISD-P creation and configuration.

The TLV, part of the function "ES8+.ConfigureISDP", is listed and described in section 5.5.2 . This TLV SHALL be encrypted and MACed with the BSP session keys.

2.5.4.3 Description of 'StoreMetadata' Block

This block comprises one or more TLV(s) containing the Profile Metadata.

The TLV(s), part of the function "ES8+.StoreMetadata", are listed and described in section 5.5.3. These TLV(s) SHALL be MACed only with the BSP session keys.

2.5.4.4 Description of 'Profile Protection Keys' Block

The 'Profile Protection Keys' block contains the function "ES8+.ReplaceSessionKeys" to replace the session S-ENC and S-MAC keys resulting from key agreement, by the keys used for protecting the Protected Profile Package, PPK-ENC and PPK_MAC.

This function is protected by BSP with the S-ENC and S-MAC keys resulting from the key agreement.

This block is optional depending on the mode selected by the SM-DP+ to protect the Profile Package (section 2.5.3).

2.5.5 Segmented Bound Profile Package

The Segmented Bound Profile Package (SBPP) is generated by the LPAd, to transfer the Bound Profile Package to the eUICC using the local interface ES10b.

The segmentation SHALL be done according to the structure of the Bound Profile Package:

- Tag and length fields of the `BoundProfilePackage` TLV plus the `initialiseSecureChannelRequest` TLV
- Tag and length fields of the first `sequenceOf87` TLV plus the first '87' TLV
- Tag and length fields of the `sequenceOf88` TLV
- Each of the '88' TLVs
- Tag and length fields of the `sequenceOf87` TLV plus the first '87' TLV
- Tag and length fields of the `sequenceOf86` TLV
- Each of the '86' TLVs

Each segment of this list that is up to 255 bytes is transported in one APDU. Larger TLVs are sent in blocks of 255 bytes for the first blocks and a last block that MAY be shorter.

At the beginning of each segment the block number of the STORE DATA commands SHALL be reset.

2.5.6 Profile Installation Result

The Profile Installation Result SHALL be returned by the eUICC after the execution of the last TLVs of the BPP, or right after the first BPP's TLV executed with error. It MAY also be returned when the installation is interrupted.

The Profile Installation Result contains the following data:

- Notification Metadata: The Notification Metadata includes:
  - Sequence Number
  - Profile Management Operation
  - Recipient Address
  - ICCID (Not provided if the Notification reports an error that has happened before ICCID was known by the eUICC, otherwise it SHALL be present)
- Transaction ID: The Transaction Identifier given to the eUICC during the Profile "Download and Installation" procedure (section 3.1.3).
- Final Result: provides the final Profile installation status.
- SM-DP+ OID: The SM-DP+ OID as contained in CERT.DPpb.SIG used during the profile download and installation procedure (section 3.1.3).
- eUICC signature: A signature created by the eUICC ensuring the authenticity and the integrity of the Profile Installation Result.

The `notificationAddress` in the `profileInstallationResultData` SHALL be set to the `serverAddress` provided in "ES10b.AuthenticateServer".

Until the Profile Download and Installation process is completed or terminated, no Result is available for the LPA.

The Profile Installation Result SHALL be kept by the eUICC (which can hold one or several Profile Installation Results) until explicitly deleted by the LPA, after successfully delivered to the SM-DP+. Before being deleted the Profile Installation Result(s) MAY be retrieved at any time by the LPA.

When the eUICC needs to store a new Profile Installation Result, if there is not enough room the eUICC SHALL delete one or more of the previously stored Profile Installation Results in order of their Sequence Number, beginning with the lowest.

The Profile Installation Result SHALL be encoded in the ASN.1 data object as shown below. It SHALL include an eUICC signature data object computed as defined in section 2.6.9, using the eUICC private key SK.EUICC.SIG selected during the Profile Download and Installation procedure, across the data object `ProfileInstallationResultData` (tag 'BF 27').

```
-- ASN1START
-- Definition of Profile Installation Result
ProfileInstallationResult ::= [55] SEQUENCE { -- Tag 'BF37'
   profileInstallationResultData [39] ProfileInstallationResultData,
   euiccSignPIR EuiccSign
}

ProfileInstallationResultData ::= [39] SEQUENCE { -- Tag 'BF27'
   transactionId[0] TransactionId, -- The TransactionID generated by the SM-DP+
   notificationMetadata[47] NotificationMetadata,
   smdpOid OBJECT IDENTIFIER, -- SM-DP+ OID (value from CERT.DPpb.SIG)
   finalResult [2] CHOICE {
      successResult SuccessResult,
      errorResult ErrorResult
   }
}

EuiccSign ::= [APPLICATION 55] OCTET STRING -- Tag '5F37', eUICC’s signature

SuccessResult ::= SEQUENCE {
   aid [APPLICATION 15] OCTET STRING (SIZE (5..16)), -- AID of ISD-P
   ppiResponse OCTET STRING -- contains (multiple) 'EUICCResponse' of the Profile
Package Interpreter as defined in [5]
}

ErrorResult ::= SEQUENCE {
   bppCommandId BppCommandId,
   errorReason ErrorReason,
   ppiResponse OCTET STRING OPTIONAL -- contains (multiple) 'EUICCResponse' of the
Profile Package Interpreter as defined in [5]
}

BppCommandId ::= INTEGER {
   initialiseSecureChannel(0),
   configureISDP(1),
   storeMetadata(2),
   storeMetadata2(3),
   replaceSessionKeys(4),
   loadProfileElements(5)
}

ErrorReason ::= INTEGER {
   incorrectInputValues(1),
   invalidSignature(2),
   invalidTransactionId(3),
   unsupportedCrtValues(4),
   unsupportedRemoteOperationType(5),
   unsupportedProfileClass(6),
   bspStructureError(7),
   bspSecurityError(8),
   installFailedDueToIccidAlreadyExistsOnEuicc(9),
   installFailedDueToInsufficientMemoryForProfile(10),
   installFailedDueToInterruption(11),
   installFailedDueToPEProcessingError (12),
   installFailedDueToDataMismatch(13),
   testProfileInstallFailedDueToInvalidNaaKey(14),
   pprNotAllowed(15),
   enterpriseProfilesNotSupported(17), -- #SupportedForEnterpriseV3.0.0#
   enterpriseRulesNotAllowed(18), -- #SupportedForEnterpriseV3.0.0#
   enterpriseProfileNotAllowed(19), -- #SupportedForEnterpriseV3.0.0#
   enterpriseOidMismatch(20), -- #SupportedForEnterpriseV3.0.0#
   enterpriseRulesError(21), -- #SupportedForEnterpriseV3.0.0#
   enterpriseProfilesOnly(22), -- #SupportedForEnterpriseV3.0.0#
   lprNotSupported(23), -- #SupportedForLpaProxyV3.0.0#
   unknownTlvInMetadata(26), -- #SupportedFromV3.0.0#
   installFailedDueToUnknownError(127)
}
-- ASN1STOP
```

NOTE: Error reason values added since v3.0.0 are aligned with the cancel session reason values in section 5.7.14.

2.5.6.1 Profile Installation Result errors

ErrorReason data object contained in ErrorResult data object depends on the function that generated an error during processing of the BoundProfilePackage. The following table details authorised combinations:

ErrorReason in ErrorResult	ES8+ function				
	Initialise Secure Channel	Configure ISDP	Store Metadata	Replace Session Keys	Load Profile Elements
incorrectInputValues(1)	✓	✓	✓	✓	✓
invalidSignature(2)	✓				
invalidTransactionId(3)	✓				
unsupportedCrtValues(4)	✓				
unsupportedRemoteOperationType(5)	✓				
unsupportedProfileClass(6)			✓		
bspStructureError(7)		✓	✓	✓	✓
bspSecurityError(8)		✓	✓	✓	✓
installFailedDueToIccidAlreadyExistsOnEuicc(9)			✓		
installFailedDueToInsufficientMemoryForProfile(10)		✓	✓		✓
installFailedDueToInterruption(11)	✓	✓	✓	✓	✓
installFailedDueToPEProcessingError(12)					✓
installFailedDueToDataMismatch(13)					✓
testProfileInstallFailedDueToInvalidNaaKey(14)					✓
pprNotAllowed(15)			✓		
enterpriseProfilesNotSupported(17)			✓		
enterpriseRulesNotAllowed(18)			✓		
enterpriseProfileNotAllowed(19)			✓		
enterpriseOidMismatch(20)			✓		
enterpriseRulesError(21)			✓		
enterpriseProfilesOnly(22)			✓		
lprNotSupported(23)			✓		
unknownTlvInMetadata(26)			✓		
installFailedDueToUnknownError(127)	✓	✓	✓	✓	✓

Table 4a: Authorised combinations of ES8+ Errors

If an error is generated during the processing of a ProfileElement of the eUICC Profile Package, ppiResponse SHALL be set as provided by the Profile Package Interpreter, and a corresponding ErrorReason SHALL be set in the Profile Installation Result according to the following table:

eUICC Profile Package error *status* in *PEStatus*	ErrorReason in ErrorResult
pe-not-supported(1)	installFailedDueToPEProcessingError(12)
memory-failure(2)	installFailedDueToPEProcessingError(12)
bad-values(3)	installFailedDueToPEProcessingError(12)
not-enough-memory(4)	installFailedDueToInsufficientMemoryForProfile(10)
invalid-request-format(5)	installFailedDueToPEProcessingError(12)
invalid-parameter(6)	installFailedDueToPEProcessingError(12)
runtime-not-supported(7)	installFailedDueToPEProcessingError(12)
lib-not-supported(8)	installFailedDueToPEProcessingError(12)
template-not-supported(9)	installFailedDueToPEProcessingError(12)
feature-not-supported(10)	installFailedDueToPEProcessingError(12)
unsupported-profile-version(31)	installFailedDueToPEProcessingError(12)
Other eUICC Profile Package status codes except (0)	installFailedDueToPEProcessingError(12)

Table 4b: eUICC Profile Package error mapping to ErrorReason

For the following ErrorReason values, which indicate that Profile installation failed due to a requirement in this specification, any eUICC Profile Package status code MAY be set:

installFailedDueToDataMismatch(13)

testProfileInstallFailedDueToInvalidNaaKey(14)

The following ErrorReason values SHALL be treated by the SM-DP+ as temporary errors, where the SM-DP+ SHALL allow retries of the Profile download as long as the download retry limit is not exceeded:

installFailedDueToInsufficientMemoryForProfile(10)

installFailedDueToInterruption(11)

All other ErrorReason values SHALL be treated as permanent errors.

2.6 Security Overview

This section provides an overview of the overall ecosystem security features.

2.6.1 Certification of the Entities

Functional compliance and security certification requirements according to SGP.24 [64] SHALL apply for the eUICC, the EUM, the SM-DP+ and the SM-DS.

The Device and the LPA SHALL comply with the security features defined in this specification and SGP.21 [4].

2.6.2 Remote Secure Communication

The RSP ecosystem relies on remote secure communication to achieve function execution requests and data exchanges. Any of the remote secure communication defined for RSP and not listed in the exceptions at the end of this section SHALL follow the rules below.

Mutual authentication:

- The Server (the entity providing the function, e.g., SM-DP+) SHALL be authenticated first by the Client (the entity requesting the function). Authentication SHALL include the verification of a Server Certificate chain ending at an eSIM CA RootCA Certificate (section 4.5.2).
- The Client SHALL be authenticated by the Server in a second step. In case the Client is the eUICC, authentication SHALL include the verification of a eUICC Certificate chain ending at an eSIM CA RootCA Certificate (section 4.5.2). Client authentication does not apply to the LPA.

Data privacy:

- The eUICC, as a Client, SHALL NOT reveal any private data to an unauthenticated Server.
- The eUICC, as a Client, SHALL NOT generate any signed material before having authenticated the Server.

Communication protection:

- Once authenticated, the remote entities SHALL negotiate a common cryptographic suite for further communication.
- The communication SHALL be origin authenticated, as well as integrity and confidentiality protected.
- Session keys SHALL be generated using Perfect Forward Secrecy.

Authorisation:

- On the basis of authentication, the Server SHALL always check that the requesting Client is authorised before delivering the requested function execution.

The following exception from these security rules exists:

- Other Notifications are origin authenticated, integrity and replay protected, in addition to the protection provided by TLS as defined in section 2.6.6.
- For the communication between the LPA and the HRI Server, only the protection by TLS as defined in section 2.6.6 SHALL be used.

2.6.3 Public Key Infrastructure

General security of the RSP ecosystem is based on a Public Key Infrastructure (PKI).

Any Certificate defined in this specification has a validation chain ending at an eSIM CA RootCA Certificate (section 4.5.2), except TLS Certificates that MAY instead chain to a Public CA.

Certificates MAY be revoked; Revocation status are managed and made available by their issuer (see section 2.7).

Certificates are used for authentication of their owners via signatures created with the associated private keys.

2.6.4 BPP Security Protocol

This section defines a security protocol for Profile Protection and eUICC Binding, named BPP Security Protocol (BSP).

NOTE: The protocol was initially derived from GlobalPlatform's SCP11a as specified by the GlobalPlatform Card Specification Amendment F [13].

2.6.4.1 Key agreement

An Elliptic Curve Key Agreement Algorithm (ECKA) is used for the establishment of a shared secret value. It shall follow the definition for the Anonymous Diffie-Hellman Key Agreement in BSI TR-03111 [41]. The algorithm is executed

- by the SM-DP+ using otPK.EUICC.KA and otSK.DP.KA, and
- by the eUICC using otPK.DP.KA and otSK.EUICC.KA

to calculate the shared secret value.

NOTE 1: Ephemeral keys are renamed to one-time keys in this specification, as they MAY live longer and are stored in non-volatile memory. With respect to Perfect Forward Secrecy, they serve the same purpose.

NOTE 2: The BSP itself does not provide authentication. Authentication is provided by signatures outside of the BSP.

2.6.4.2 Key derivation

Session keys and an initial MAC chaining value are derived from the shared secret value as follows:

- Concatenate the following values as SharedInfo as input for the Key Derivation process (this data is the one given as input data in the function "ES8+.InitialiseSecureChannel"):
  - Key type (1 byte)
  - Key length (1 byte)
  - HostID-LV and EID-LV. HostID-LV comprises the length and the value field of the HostID given in the input data; EID-LV comprises the length and value field of the EID.
- Initial MAC Chaining value, S-ENC and S-MAC are taken from KeyData derived from the shared secret value and the SharedInfo as defined in BSI TR-03111 [41] for the "X9.63 Key Derivation Function". SHA-256 SHALL be used for the key derivation to calculate KeyData of sufficient length. Data is assigned as defined in the following table:

KeyData	Key
1 to L	Initial MAC chaining value
L+1 to 2L	S-ENC
2L+1 to 3L	S-MAC

Table 4c: Key Data

The initial MAC chaining value is used for the computation of the MAC of the first data block as defined below.

2.6.4.3 Command TLV Protection Overview

The next two section describe how data is protected by the BSP.

NOTE: An earlier version of the protocol, named SCP03t, was derived from GlobalPlatform's SCP03 [11] and can be found in SGP.02 [2]. As the protocol in this document was further modified and not much is left of SCP03, a new name is used.

BSP defines two options for protecting Command TLVs:

- MAC and encryption
- MAC only

Command TLVs do not create response TLVs.

2.6.4.4 Command TLV MACing and Encryption

BSP C-MAC and C-ENCRYPTION are done as follows:

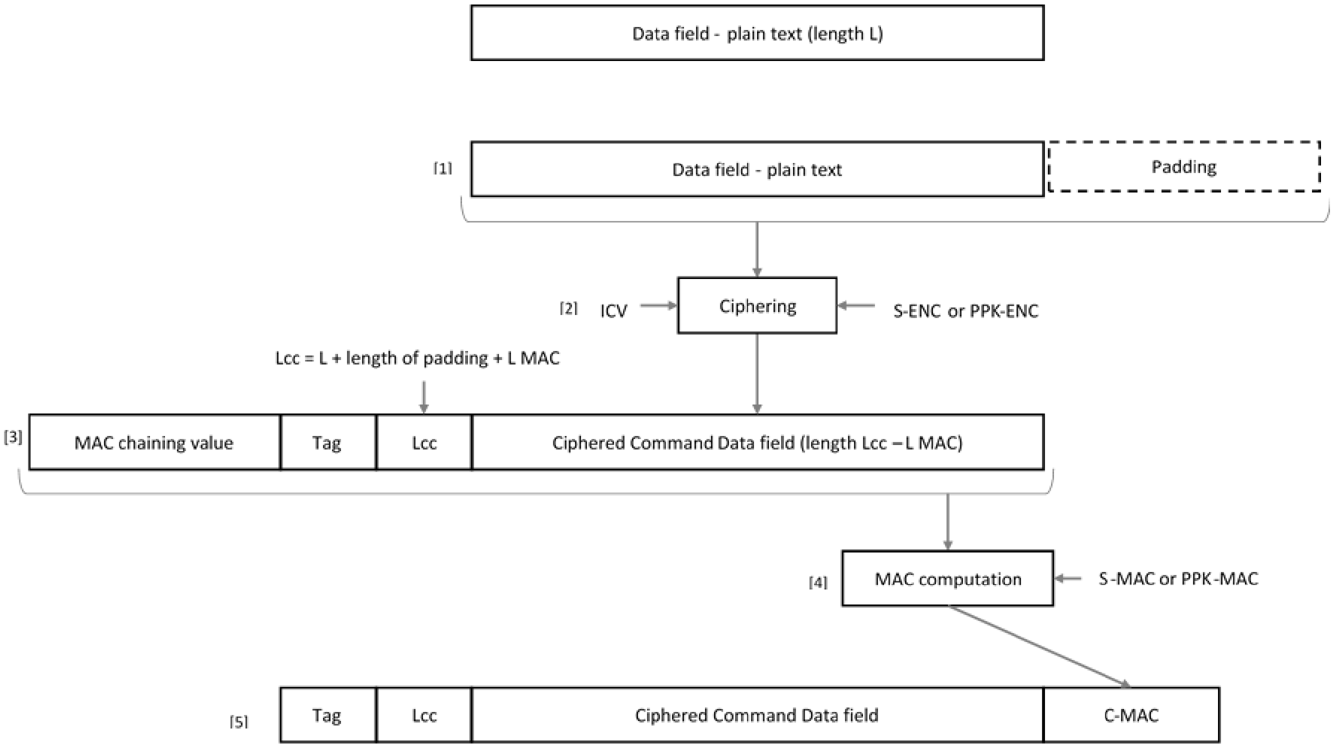

Data field - plain text (length L)
[1]
Data field - plain text
Padding
[2]
ICV
Ciphering
S-ENC or PPK-ENC
Lcc = L + length of padding + L MAC
[3]
MAC chaining value
Tag
Lcc
Ciphered Command Data field (length Lcc – L MAC)
[4]
MAC computation
S-MAC or PPK-MAC
[5]
Tag
Lcc
Ciphered Command Data field
C-MAC

Figure 4a: TLV Command Data Field Encryption and C-MAC computation

Procedure:

1. The data field is padded according to the encryption algorithm and mode used, as defined in table 4c.

   If this algorithm is AES-CBC-128 or SM4-CBC, padding SHALL be done as follows:

   - Append a byte with value '80' to the right of the data block;

- Append 0 to 15 bytes with value '00' so that the length of the padded data block is a multiple of 16 bytes.

2. The result of step 1 is encrypted according to the encryption algorithm and mode used, as defined in table 4c.

   If this algorithm is AES-CBC-128 or SM4-CBC, the following applies:
   - The data blocks SHALL be numbered starting from 1
   - The binary value of this number SHALL be left padded with zeroes to form a full block.
   - This block SHALL be encrypted with S-ENC to produce the ICV for command encryption.
3. The input data used for C-MAC computation comprises the MAC Chaining value, the tag, the final length and the result of step 2.
   - The initial MAC Chaining value is set as defined in 2.6.4.2 or 2.6.4.6.
   - Subsequent MAC chaining values are the full result of step 4 of the previous data block (which may also be a data block with C-MAC only).
4. The full MAC value is computed using the MACing algorithm as defined in table 4c.
5. The output data is computed by concatenating the following data: the tag, the final length, the result of step 2 and the C-MAC value.

   If the algorithm is AES-CBC-128 or SM4-CBC, the C-MAC value is the 8 most significant bytes of the result of step 4.

2.6.4.5 Command TLV MACing

BSP C-MAC only is done as follows:

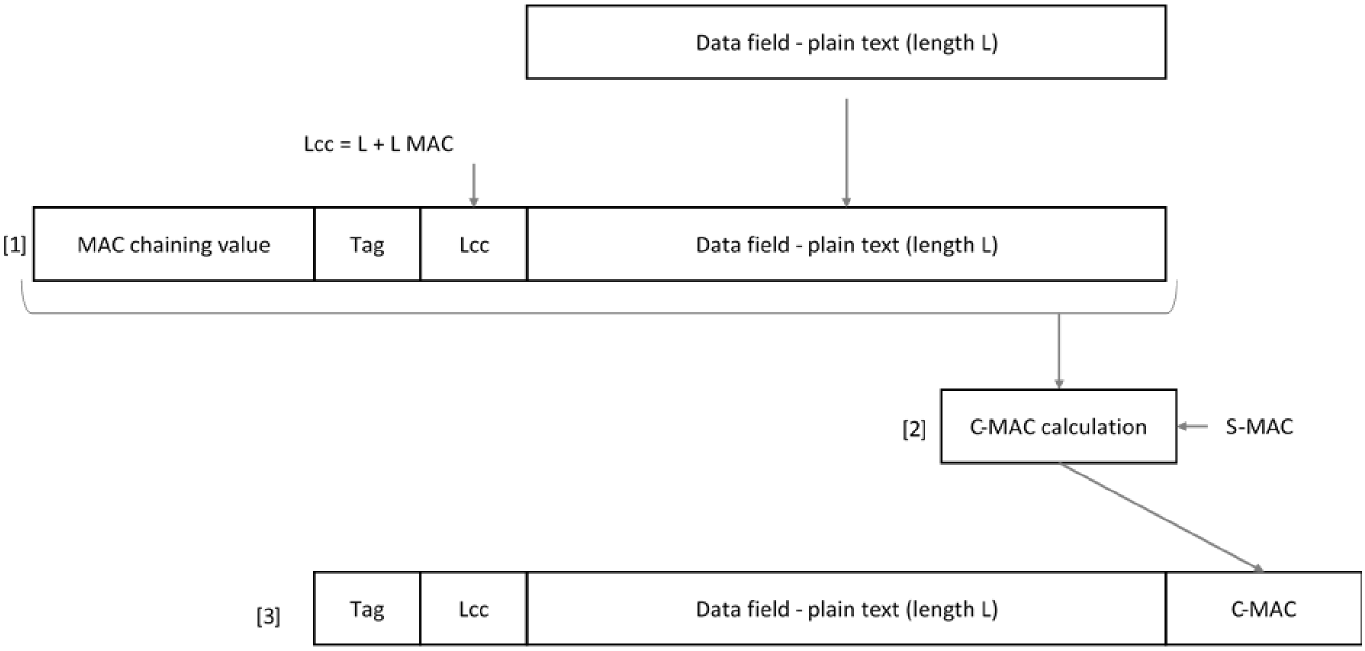

Data field - plain text (length L)
Lcc = L + L MAC
[1]
MAC chaining value
Tag
Lcc
Data field - plain text (length L)
[2]
C-MAC calculation
S-MAC
[3]
Tag
Lcc
Data field - plain text (length L)
C-MAC

Figure 4b: TLV Command C-MAC computation

Procedure:

1. Data used for C-MAC computation comprised MAC Chaining value, applicable tag (e.g., tag ‘88’ for Metadata data segment), length and data.

   In case algorithm is AES-CMAC-128 or SM4-CBC, the following applies:
   - MAC chaining values are the full result of the previous data block (which may also be a data block with C-ENCRYPTION and C-MAC).
2. C-MAC is computed according to symmetric algorithm used, as defined in table 4c.
3. The output data is computed by concatenating the following data: the tag, the final length, the data and the C-MAC value.

   If the algorithm is AES-CBC-128 or SM4-CBC, the C-MAC value is the 8 most significant bytes of the result of step 2.

The data block counter for ICV calculation is incremented also for each segment with C-MAC only.

2.6.4.6 Key replacement

The session keys can be replaced by providing new keys and a new initial MAC chaining value within a protected Command TLV, see section 5.5.4.

The data block counter used to calculate the ICV for encryption SHALL also be reset to its initial value after a key replacement.

2.6.5 Cryptographic Negotiation, Algorithms and Key Length

This section describes the mechanism for cryptographic negotiation at application layer between the eUICC and the RSP Server used on top of TLS. TLS provides its own cryptographic negotiation (see section 2.6.6.1). The cryptographic negotiation at the application layer is independent of TLS.

The application layer cryptographic negotiation proceeds between the eUICC and the RSP Server at the beginning of each RSP Session. All the cryptography is determined based on the RSP Server Certificate selected for the Common Mutual Authentication procedure. See section 3.0.1 for the detailed description of the procedure. Application layer cryptography is constrained to a limited number of algorithm sets listed in Tables 4c and 4d. Additional algorithm sets may be defined in the future.

The cryptographic negotiation starts with the eUICC providing the list of eSIM CA RootCA Public Key Identifiers that it supports for signature verification and signature generation within the ES9+.InitiateAuthentication function (see section 5.6.1).

Then the SM-DP+ selects a Certificate (CERT.XXauth.SIG) whose chain ends at one of the eSIM CA RootCA Certificate(s) that the eUICC supports for signature verification (the selection process is further detailed in section 5.6.1).

Each Public Key Identifier refers to an eSIM CA RootCA Certificate, which implicitly defines other cryptographic algorithms that the RSP Server and the eUICC SHALL use during the RSP session. This specification also mandates a simplification that all the Certificates in a chain SHALL have the same signature algorithm (i.e., 'AlgorithmIdentifier.algorithm' and 'AlgorithmIdentifier.parameters' fields in the 'subjectPublicKeyInfo' have the same values in all Certificates of the chain). This simplification ensures that a receiving party is able to verify both the selected Certificate chain and the signature.

Based on the signature algorithm mentioned in the CERT.XXauth.SIG, cryptographic algorithms to be used during the RSP Session SHALL be determined according to the table below:

Signature algorithm and parameters from CERT.XXauth.SIG	Key agreement algorithm and parameters	Symmetric algorithms for PPP	Hashing for SM-XX digital signatures
ECDSA[(1)], NIST P-256	ECKA[(2)], NIST P-256, SHA-256	AES-CBC-128 AES-CMAC-128	As indicated in the CERT.XXauth.SIG[(3)]
ECDSA[(1)], BrainpoolP256r1	ECKA[(2)], BrainpoolP256r1, SHA-256	AES-CBC-128 AES-CMAC-128	As indicated in the CERT.XXauth.SIG[(3)]
ECDSA[(1)], FRP256V1	ECKA[(2)], FRP256V1, SHA-256	AES-CBC-128 AES-CMAC-128	As indicated in the CERT.XXauth.SIG[(3)]
SM2 Signature	ECKA[(2)], SM2 Curve, SM3	SM4-CBC SM4-CMAC	SM3
Note 1: ECDSA SHALL follow GlobalPlatform Card Specification Amendment E [12]. Note 2: The ECKA algorithm is defined in section 2.6.4. Note 3: Currently restricted to SHA-256.			

Table 4c: Cryptographic algorithm sets based on CERT.XXauth.SIG

NOTE: As per NIST SP 800-57 Part 1 [86], ECC256 (128-bit security strength) is sufficient for implementations beyond year 2031.

Algorithm references:

- NIST P-256 is defined in Digital Signature Standard [29] (recommended by NIST).
- BrainpoolP256r1 is defined in RFC 5639 [18] (recommended by BSI).
- FRP256V1 is defined in ANSSI ECC [20] (recommended by ANSSI).
- AES-CBC-128, AES in CBC mode with key length of 128 bits, is defined in NIST SP 800-38A [83].
- AES-CMAC-128, CMAC with AES in CBC mode with key length of 128 bits, is defined in NIST SP 800-38B [84].
- SM2 Curve is defined in RFC8998 [96]
- SM2 is defined in SM2 algorithm [93]
- SM3 is defined in SM3 algorithm [94]
- SM4 is defined in SM4 algorithm [95]
- SM4-CBC, SM4 in CBC mode, with CBC mode as defined in NIST SP 800-38A [83].
- SM4-CMAC, SM4 in CMAC mode, with CMAC mode as defined in NIST SP 800-38B [84].

An eUICC SHALL have at least two sets of elliptic curve parameters preloaded by the EUM during eUICC manufacturing, subject to support by the corresponding eSIM CA.

For each eSIM CA trust chain to which it belongs, an RSP Server SHALL support all sets of elliptic curve parameters that the corresponding eSIM CA supports.

The Key Agreement algorithm is used in RSP for the establishment of the session keys between the eUICC and the SM-DP+.

The symmetric algorithms are used in RSP for the protection of the Profile Package (Protected Profile Package), see section 2.5.

This specification allows the eUICC to use different signature algorithms than the RSP server to generate its signatures, subject to their support by both parties. This is selected by the RSP Server and indicated in ES9+.InitiateAuthentication response (field `euiccCiPKIdToBeUsed`, see section 5.6.1). The eUICC SHALL use the signature algorithm selected by the RSP Server for all signatures generated by the eUICC during the RSP Session, including for the Profile Installation Result and the Load RPM Package Result. For all other signed Notifications related to a Profile, the eUICC SHALL use the digital signature algorithm selected during the Profile Download and Installation procedure. The selection SHALL be done according to the table below:

Digital signature algorithm and parameters from CERT.EUICC.SIG	Hashing for eUICC digital signatures
ECDSA(1), NIST P-256(2)	As indicated in the CERT.EUICC.SIG(3)
ECDSA(1), BrainpoolP256r1(2)	As indicated in the CERT.EUICC.SIG(3)
ECDSA(1), FRP256V1(2)	As indicated in the CERT.EUICC.SIG(3)
SM2 Signature	SM3
Note 1: ECDSA SHALL follow GlobalPlatform Card Specification Amendment E [12]. Note 2: The ECC domain parameters are defined above. Note 3: Currently restricted to SHA-256	

Table 4d: Cryptographic algorithm sets for the eUICC Signature

2.6.6 TLS Requirements

RSP mandates use of TLS v1.2 as defined in RFC 5246 [16] as the minimal version for TLS connection.

TLS with mutual authentication SHALL be used over ES12 and ES15.

TLS with server authentication SHALL be used over ES9+, ES11, ES20 and EShri.

TLS with mutual authentication MAY be used over ES2+. If TLS with mutual authentication is not used over ES2+, ES2+ SHALL be protected with level of security equivalent to TLS.

On ES12 and ES15, the Root Certificate of a certificate chain used for mutual authentication SHALL be an eSIM CA or any CA that both parties trust.

On ES9+, ES11 and EShri, the Root Certificate of a certificate chain used for server authentication SHALL be an eSIM CA or a Public CA.

On ES20, the Root Certificate of a certificate chain used for server authentication SHALL be a Public CA.

On ES2+, if TLS is used, the Root Certificate of a certificate chain used for mutual authentication can be any CA that both parties trust.

The certificates used for TLS SHALL fulfil the requirements described in the next section.

2.6.6.1 Cipher suites and algorithms

In this specification, a number of cryptographic algorithms and security properties applicable to TLS are specified to ensure a minimum security level and also to facilitate interoperability.

NOTE: for TLS certificates, RSP Servers are allowed to use Public CAs in addition to eSIM CAs.

2.6.6.1.1 Elliptic curves

For each eSIM CA that it supports, an RSP Server SHALL support the following elliptic curve(s) for signing and key-agreement:

- NIST P-256 with the namedCurve secp256r1 (23) as defined in RFC 8422 [65].

In addition, for each eSIM CA that is supports, an RSP Server implementing TLS 1.3 [42] MAY support one or more additional elliptic curves for signing and key-agreement, including but not limited to:

- brainpoolP256r1(26) as defined in RFC 7027 [66],
- NIST P-384 with the namedCurve secp384r1 (24) as defined in RFC 8422 [65],
- Ed25519 as defined in RFC 8410 [90], and x25519 with the namedCurve x25519 (29) as defined in RFC 8422 [65]

For each eSIM CA that it supports, an RSP Client SHALL support the following elliptic curve(s) for signing and key-agreement:

- NIST P-256 with the namedCurve secp256r1 (23) as defined in RFC 8422 [65].

In addition, for each eSIM CA that it supports, an RSP Client MAY support one or more additional elliptic curves for signing and key-agreement, including but not limited to:

- brainpoolP256r1(26) as defined in RFC 7027 [66],
- NIST P-384 with the namedCurve secp384r1 (24) as defined in RFC 8422 [65],
- Ed25519 as defined in RFC 8410 [90], and x25519 with the namedCurve x25519 (29) as defined in RFC 8422 [65].

2.6.6.1.2 Digital signature algorithm for signing

The eSIM CA SHALL use the following digital signature algorithm for signing RSP Server TLS certificates:

- ECDSA with NIST P-256 with SHA-256

NOTE: other digital signature algorithm for signing RSP Server TLS certificates might be used.

2.6.6.1.3 Cipher suites

To fulfil the security requirements of the previous section, the client SHALL offer sha256/ecdsa in the "supported_signature_algorithms" of TLS 1.2 [16] and the server SHALL select this hash/signature pair.

RSP Servers implementing TLS 1.2 [16] (e.g., SM-DP+) SHALL support at least the following cipher suites which provide(s) perfect forward secrecy:

- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289 [46]
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289 [46]

RSP Servers implementing TLS 1.3 [42] SHALL support at least the following cipher suite(s) which provide(s) perfect forward secrecy:

- TLS_AES_128_GCM_SHA256 as defined in TLS 1.3 [42]
- TLS_AES_256_GCM_SHA384 as defined in TLS 1.3 [42]

RSP Servers implementing TLS 1.2 [16] MAY support one or more additional cipher suites which SHALL provide perfect forward secrecy, including but not limited to:

- TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256 as defined in RFC 7905 [91]
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289 [46]
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289 [46]

RSP Servers that need to support RSP V2 Clients as defined in SGP.22 V2 SHOULD support TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289 [46]. However, the use of this cipher suite SHALL be limited to the case when it is the only cipher suite commonly supported by the RSP Server and the RSP V2 Client.

RSP Servers implementing TLS 1.3 [42] MAY support one or more additional cipher suites which SHALL provide perfect forward secrecy, including but not limited to:

- TLS_CHACHA20_POLY1305_SHA256 as defined in TLS 1.3 [42]

Clients implementing TLS 1.2 [16] SHALL support at least the following cipher suite(s) which provide(s) perfect forward secrecy:

- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289 [46]

Clients implementing TLS 1.2 [16] SHALL NOT support the following cipher suite(s).

- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289 [46]

NOTE: CBC encryption in MAC-then-Encrypt mode has repeatedly shown weaknesses in the past, for which only imperfect fixes exist. For this reason, such cipher suites should no longer be used.

Clients implementing TLS 1.3 [42] SHALL support at least the following cipher suite(s) which provide(s) perfect forward secrecy:

- TLS_AES_128_GCM_SHA256 as defined in TLS 1.3 [42]

Clients implementing TLS 1.2 [16] MAY support one or more additional cipher suites that provide perfect forward secrecy, including but not limited to:

- TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256 as defined in RFC 7905 [91]
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289 [46]
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289 [46]

Clients implementing TLS 1.3 [142] MAY support one or more additional cipher suites that provide perfect forward secrecy, including but not limited to:

- TLS_CHACHA20_POLY1305_SHA256 as defined in TLS 1.3 [42]
- TLS_AES_256_GCM_SHA384 as defined in TLS 1.3 [42]

2.6.6.2 Use of Server Name indication extension for ES9+ and ES11

Starting from version 3.0, the specification allows an RSP Server to use, over ES9+ and ES11, a TLS certificate which chains to a Public CA, whereas versions 2.x required it to chain to an eSIM CA. This flexibility, even if desirable, adds an additional burden on the RSP Server to select a TLS certificate chain that the LPA is able to verify.

This section provides to the LPA and the RSP Server a mechanism to guide the RSP Server selection of an appropriate TLS certificate chain.

The RSP Server and the LPA SHOULD support this mechanism. The Root SM-DS SHALL support this mechanism. An RSP Server and an LPA that support this mechanism SHALL comply with this section.

General principle:

In addition to its "base" FQDN, the RSP Server SHALL support a specific FQDN for v3 Devices. The v3-specific FQDN SHALL be the concatenation of the string "rsp3-" and the "base" UTF-8 encoded FQDN. Therefore the RSP Server SHALL take care that neither the resulting label nor the domain name in their final encoding violate the length limits of domain names.

A v3 Device SHALL include the "server_name" extension defined in RFC 6066 [60] in the ClientHello with the v3-specific FQDN computed from the "base" FQDN known from the RSP Server.

NOTE: This mechanism increases the probability that the LPA supports the TLS certificate chain selected by the RSP Server, but there is still a risk that it is not the case and the TLS handshake may fail. In that case the LPA MAY retry and the RSP Server MAY select a different certificate chain.

LPA/Device side:

The LPA SHALL compute the v3-specific FQDN computed from the "base" FQDN known from the RSP Server (e.g., an address retrieved from the eUICC, an address read from an Activation code).

The LPA SHALL perform a DNS resolution using the v3-specific FQDN. In case of DNS lookup failure (e.g., this may happen if the RSP Server is a v2 server), the LPA SHALL revert to the "base" FQDN and proceed with the TLS handshake, which SHOULD include sending the "server_name" extension.

If the DNS resolution using the v3-specific FQDN succeeds, the LPA SHALL include the "server_name" extension in the ClientHello message and containing the v3-specific FQDN. If the RSP Server replies with a fatal-level alert, the LPA SHALL retry the TLS handshake without sending the Server Name indication extension.

RSP Server side:

The RSP Server SHALL support the "server_name" extension to determine the TLS certificate to use.

- If the received ClientHello message contains the "server_name" extension with the v3-specific FQDN, the RSP Server SHOULD use a TLS certificate that chains to a Public CA.
- If the received ClientHello message contains the "server_name" extension with an unrecognized FQDN, the RSP Server SHALL take one of two actions: either abort the handshake by sending a fatal-level unrecognized_name(112) alert or continue the handshake with a TLS certificate that chains to a Public CA.
- If the received ClientHello message doesn't contain the "server_name" extension, the RSP Server SHOULD use a TLS certificate that chains to an eSIM CA.

NOTE: An RSP Server cannot rely upon receiving a "server_name" extension for "base" FQDNs (i.e., those without an "rsp3-" prefix). If the server can be reached through multiple "base" FQDNs that resolve to the same IP address, then it may be unable to select a TLS certificate containing the FQDN expected by the LPA.

2.6.7 Void

2.6.8 Random Number Generation

To protect against attacks, a high quality random number generator is required. Recommendations for appropriate random number generators are given by BSI [78] and NIST [79].

2.6.9 Digital Signature Computation

When applied to an ASN.1 data object, the digital signature (using the algorithm determined as described in section 2.6.5) SHALL be computed for the data object after encoding (i.e., in its DER representation).

2.7 Certificate Revocation

See also section 4.5.2 that describes certificates' chains of trust.

The following certificates MAY be revoked at any time:

- eSIM CA RootCA Certificate (CERT.CI.SIG)
- eSIM CA SubCA Certificate (CERT.CISubCA.SIG)
- EUM Certificate (CERT.EUM.SIG) and EUM SubCA Certificate (CERT.EUMSubCA.SIG)
- SM-DP+ SubCA Certificate (CERT.DPSubCA.SIG)
- SM-DP+ Certificates (CERT.DPauth.SIG, CERT.DPpb.SIG)
- SM-DP+ TLS Certificate (CERT.DP.TLS)
- SM-DS SubCA Certificate (CERT.DSSubCA.SIG)
- SM-DS Certificate (CERT.DSauth.SIG)
- SM-DS TLS Certificate (CERT.DS.TLS)

The means by which an eSIM CA RootCA Certificate revocation status is made available is out of scope of this specification. However, as soon as such revocation status is known by an RSP entity, it SHALL no longer accept or perform any RSP operation using a certificate chain certified by this eSIM CA RootCA Certificate. Section 2.4.2 describes a means by which the eUICC can be updated to reflect that an eSIM CA RootCA Certificate has been revoked.

Because of their potential number, eUICC Certificates (CERT.EUICC.SIG) are not revoked individually. Also, it is unlikely that an individual eUICC would be compromised. It is instead more probable that an eUICC model or an entire eUICC production batch would be declared as compromised. This approach is reflected by revoking the EUM Certificate or the EUM SubCA Certificate, if used by the EUM, attached to the production of the particular eUICC model or batch.

As a consequence, it is up to the EUM to consider using distinct certificates (CERT.EUM.SIG or CERT.EUMSubCA.SIG) for distinct eUICC models or production batches. This is out of the scope of this specification.

The following RSP entities SHALL manage the revocation status of the Certificates they issue:

- An eSIM CA.
- An EUM, for CERT.EUMSubCA.SIG, when the additional EUM SubCA is used.
- An SM-DP+ that uses an additional SM-DP+ SubCA, for CERT.DPauth.SIG, CERT.DPpb.SIG, and CERT.DP.TLS, when these Certificates do not have a short validity period.
- An SM-DS that uses an additional SM-DS SubCA, for CERT.DSauth.SIG, and CERT.DS.TLS, when these Certificates do not have a short validity period.

NOTE: The notion of the short validity period is further defined in SGP.14 [45].

A revoked certificate SHALL NOT be automatically renewed. Renewal SHALL be upon the eSIM Certification Authority agreement (see SGP.14 [45] section 5.6).

RSP entities that manage certificate revocation (further called a CRL Issuer) SHALL publish the revocation status by means of a CRL.

An RSP Server SHALL support revocation status verification only by means of CRLs.

2.8 Void

The content of this section has been moved to section 2.4a.

2.9 Profile Policy Management

Profile Policy Management provides mechanisms by which Profile Owners can enforce the conditions of use under which services are provided.

Profile Policy Management comprises three main elements:

- Profile Policy Rules (PPR)
- Rules Authorisation Table (RAT)
- Profile Policy Enabler (PPE)

More details are provided in the next sub sections.

2.9.1 Profile Policy Rules

The Profile Policy Rules (PPRs) are defined by the Profile Owners and set by the SM-DP+ in the Profile Metadata. They are also accessible by the LPA for verification or display to the End User.

A Profile MAY have zero or more Profile Policy Rules.

A Test Profile SHOULD NOT contain any Profile Policy Rules.

When installed on the eUICC, Profile Policy Rules SHALL be contained in the associated Profile.

The following Profile Policy Rules are defined in this version of the specification:

- (PPR1) 'Disabling of this Profile is not allowed'
- (PPR2) 'Deletion of this Profile is not allowed'

The coding of PPRs is given in section 2.4a.1.1.

2.9.2 Rules Authorisation Table (RAT)

The Rules Authorisation Table (RAT) contains the description of the acceptable set of PPRs that can be set in a Profile. The RAT is defined at eUICC platform level and is used by the Profile Policy Enabler (PPE) and the LPA to determine whether or not a Profile that contains PPRs is authorised and can be installed on the eUICC.

The RAT is initialised at eUICC manufacturing time or during the initial Device setup provided that there is no installed Operational Profile. The Device manufacturer or EUM is responsible for setting the content of the RAT.

The RAT SHALL NOT be affected by the ES10c.eUICCMemoryReset function (section 5.7.19).

2.9.2.1 Profile Policy Authorisation Rules (PPAR)

The RAT contains a list of Profile Policy Authorisation Rules (PPAR).

A PPAR is composed of the following information:

Data	Description
Profile Policy Rule Identifier	Identifies the Profile Policy Rules to which this PPAR applies. This field SHALL contain one or several PPR(s) being set as defined in 2.4a.1.1.
Allowed Operators	List of Profile Owners, as defined in section 2.4a.1.2, allowed to use this PPR. Wildcards can be used to indicate that all, or a set of, Profile Owners are allowed. See below.
End User Consent Required	Indicates if the related PPR needs the End User Consent for the Profile to be installed (true/false).

Table 5: PPAR description

The RAT MAY contain zero or more PPAR(s) related to a particular PPR. The order of the PPARs in the RAT is significant (see below).

The RAT of an eUICC supporting MEP SHALL NOT contain any PPAR for PPR1.

'Allowed Operators' field

The 'Allowed Operators' field contains the identifier(s) of the Profile Owner(s) (explicitly listed or matching a wild card) allowed to use the related PPR. It SHALL be compared against the `profileOwner` field of the Metadata of the Profile.

Any of the digits of the `mccMnc` data object can be wildcard-ed by setting the appropriate nibble to 'E'.

If present in the PPAR, a non-empty `gid1` or `gid2` value SHALL exactly match the corresponding value in the `profileOwner` field.

The `gid1` or `gid2` data objects can be wildcard-ed by setting an empty value (length zero).

An omitted `gid1` or `gid2` value in the PPAR SHALL only match a `profileOwner` field where the corresponding `gid1` or `gid2` value is absent.

NOTE: A PPR MAY be allowed for all Profile Owners by setting the 'Allowed Operators' field with a unique OperatorId having the `mccMnc` field value set to 'EEEEEE' and `gid1` and `gid2` data objects set with an empty value (length zero).

A PPR MAY be 'forbidden' for all Profile Owners by not defining any related PPAR.

Case where multiple PPARs are defined for a PPR:

A PPR is allowed for a Profile Owner whose identifier appears in the 'Allowed Operators' field (explicitly listed or matching a wild card) in one of the related PPARs.

'End User Consent required' field

When set to 'true', it indicates that for all Profile Owners allowed by the 'Allowed Operators' field the LPA SHALL get the End User Consent for the related PPR to install the Profile.

When set to 'false', it indicates that this End User Consent is not mandatory.

Case where multiple PPARs are defined for a PPR:

When a Profile Owner is allowed in several PPARs (explicitly listed or matching a wild card), the 'End User Consent required' field value of the first of these PPARs SHALL be used.

Example of RAT configuration (for illustration only and not intended to represent a real case):

PPRid	Allowed Operators	End User Consent Required
PPR1	OP-A	false
PPR2	OP-B	false
PPR1, PPR2	*	true

The '*' in the 'Allowed Operators' field denotes a PPR that is allowed for any Profile Owner; and if there is no PPAR for a particular PPR, then that PPR is forbidden.

With this configuration, Operator OP-A:

- can use PPR1 without the End User consent
- can use PPR2 with the End User consent

With this configuration, Operator OP-B:

- can use PPR1 with the End User consent
- can use PPR2 without the End User consent

With this configuration, any other Profile Owner:

- can use PPR1 and PPR2 with the End User consent

2.9.2.2 Notable RAT configurations

'All PPRs allowed for all Profile Owners, End User Consent required'

PPRid	Allowed Operators	End User Consent Required
PPR1, PPR2	*	true

'All PPRs forbidden for all Profile Owners'

PPRid	Allowed Operators	End User Consent Required
<no entry>		

2.9.2.3 Void

2.9.2.4 LPA verification

During the Profile Download and Installation procedure (see section 3.1.3), the LPA SHALL verify that the PPRs defined in the Profile to install can be set by the Profile Owner, and if an End User Consent is required.

The figure below describes the process to determine if all PPRs of a Profile can be set by the Profile Owner, and if an End User consent is required, according to RAT configuration.

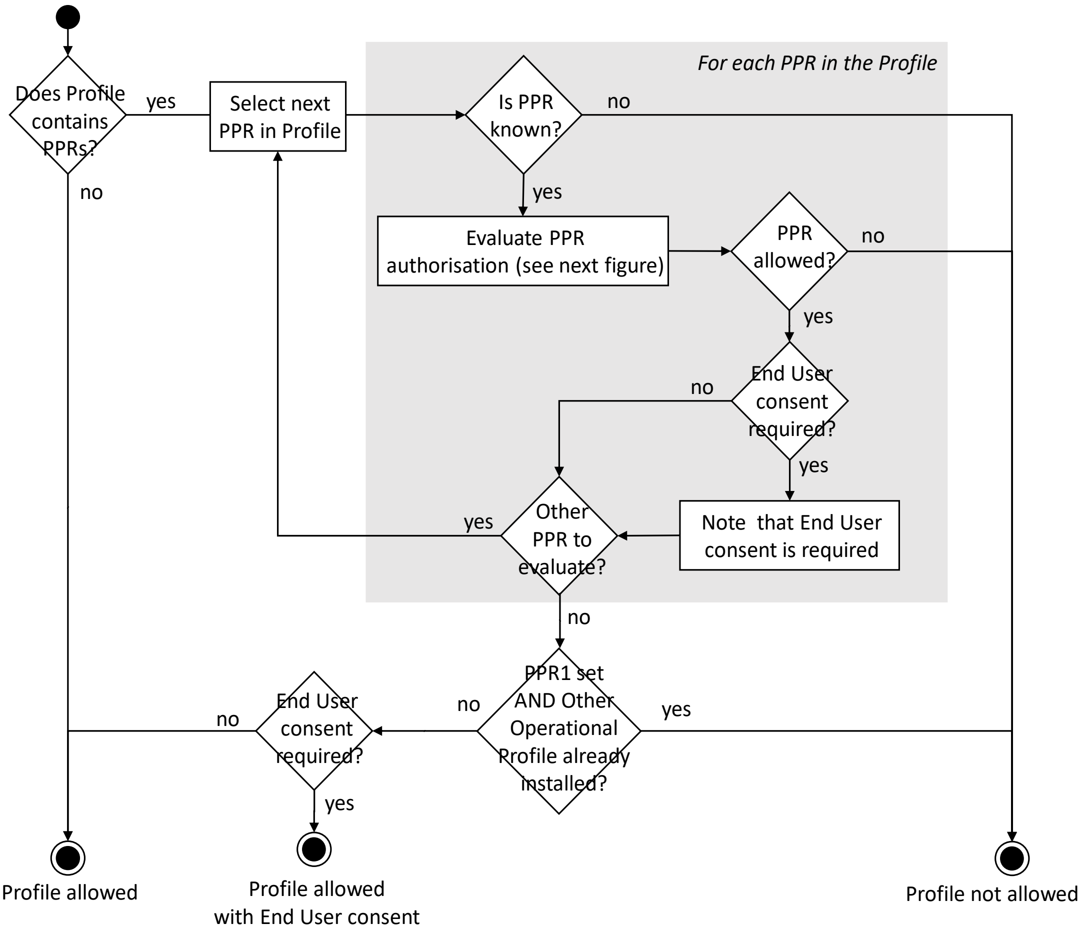

For each PPR in the Profile
Does Profile contains PPRs?
yes
no
Select next PPR in Profile
Is PPR known?
no
yes
Evaluate PPR authorisation (see next figure)
PPR allowed?
no
yes
End User consent required?
no
yes
Note that End User consent is required
Other PPR to evaluate?
yes
no
PPR1 set AND Other Operational Profile already installed?
no
yes
End User consent required?
no
yes
Profile allowed
Profile allowed with End User consent
Profile not allowed

Figure 5: Profile's PPRs verification by LPA

The figure below describes the process to determine if a particular PPR can be set by the Profile Owner, and if an End User consent is required, according to its related PPAR(s) configuration.

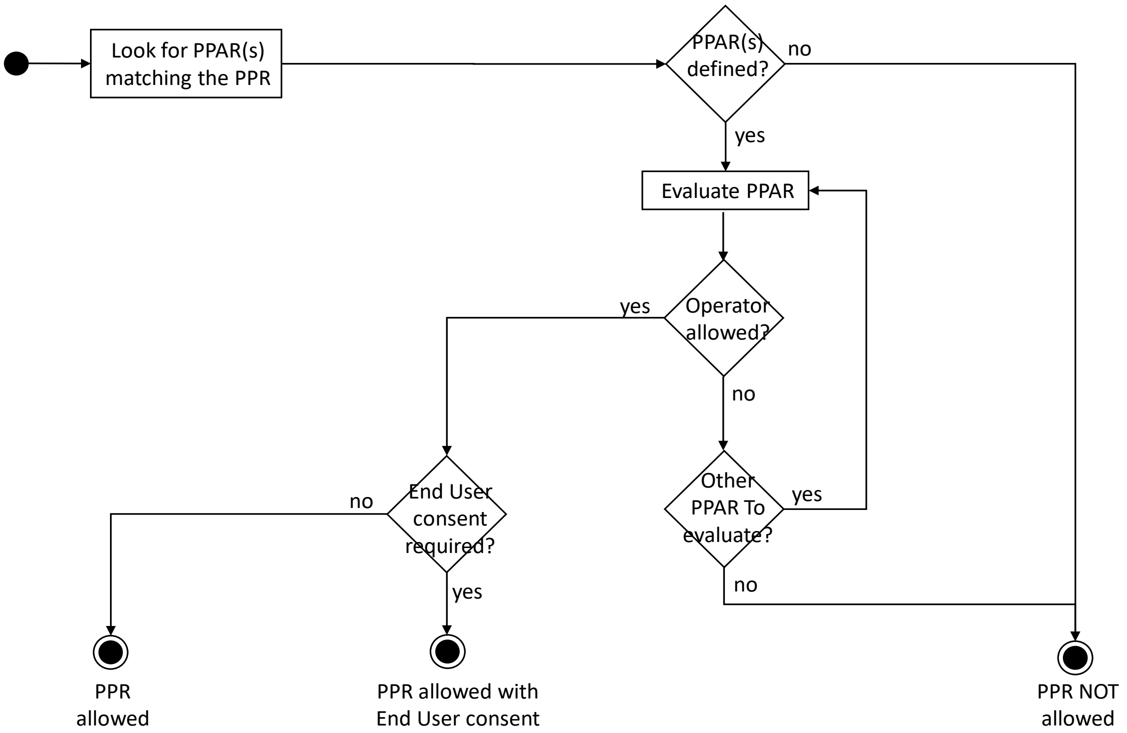


Figure 6: Particular PPR verification by the LPA

2.9.3 Profile Policy Enabler

The Profile Policy Enabler on the eUICC has two functions:

- Verification that a Profile containing PPRs is authorised by the RAT.
- Enforcement of the PPRs of a Profile.

2.9.3.1 PPRs Verification: Profile installation time

At Profile installation time the Profile Policy Enabler SHALL verify each of the PPRs as described below, to determine if it allows the Profile installation to continue. If the verification results in the Profile not being allowed, then the Profile installation SHALL be rejected and a Profile Installation Result SHALL be generated and returned to the LPA.

The PPE cannot enforce that the End User consent, if any is required, is captured by the LPA. As a consequence the 'End User Consent required' field SHALL NOT be considered during the PPRs verification by the PPE.

The figure below describes the process that the PPE SHALL run to determine if a Profile containing PPRs can be installed on the eUICC, according to the RAT configuration.

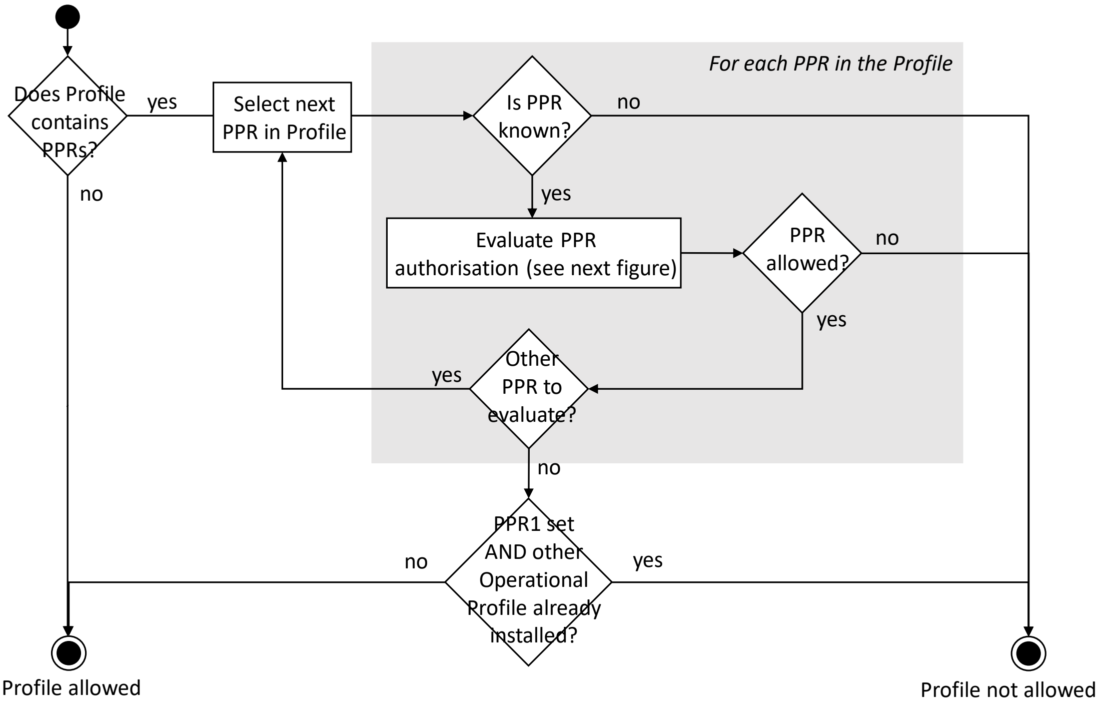

For each PPR in the Profile
Does Profile contains PPRs?
yes
no
Select next PPR in Profile
Is PPR known?
no
yes
Evaluate PPR authorisation (see next figure)
PPR allowed?
no
yes
Other PPR to evaluate?
yes
no
PPR1 set AND other Operational Profile already installed?
no
yes
Profile allowed
Profile not allowed

Figure 7: Profile's PPRs verification by PPE

The figure below describes the process to determine if a PPR is allowed according to its related PPAR(s) configuration.

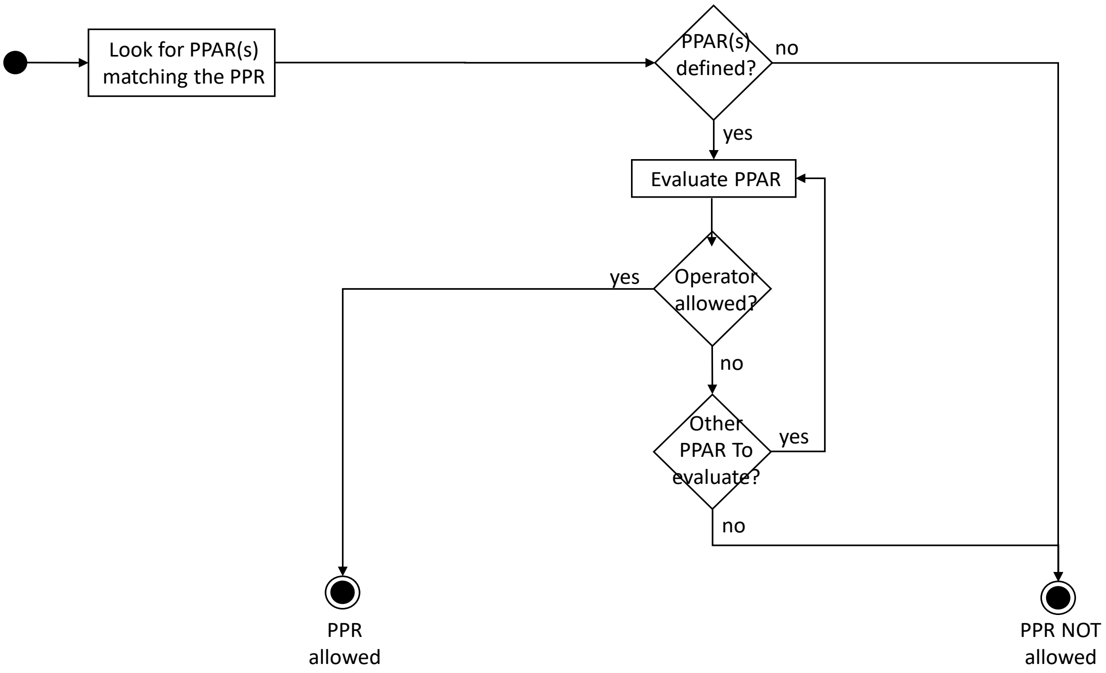


Figure 8: Particular PPR verification by the PPE

2.9.3.2 PPR Verification: PPR update after Profile is installed

A PPR in a Profile installed in the eUICC can be unset (using the "ES6.UpdateMetadata" Function by the Profile Owner or the RPM Command 'Update Metadata'). The setting of a PPR in the eUICC is for further study.

2.9.3.3 PPR Enforcement

The Profile Policy Enabler SHALL enforce the PPRs of a Profile when a Local Profile Management Operation is requested upon this Profile. Each of the defined enforcement cases are described in the concerned procedures (see section 3.2 and 3.3).

2.9.3.3.1 Void

Table 6: Void

2.9.3.3.2 Enforcement involving Test Profile

When a Test Profile is requested to be enabled whereas the currently Enabled Profile has a PPR1 set, PPE SHALL NOT enforce this PPR1 to allow the Test Profile to be enabled.

2.9.3.3.3 Void

2.10 Remote Profile Management

This section describes the structure of RPM Package composed of one or more RPM Commands. This section also describes how an eUICC generates the execution result of an RPM Package.

2.10.1 RPM Package

The SM-DP+ SHALL generate an RPM Package upon the request of Operator. The RPM Package SHALL be encoded in the ASN.1 data object as shown below.

```
-- ASN1START
RpmPackage ::= SEQUENCE OF RpmCommand -- #SupportedForRpmV3.0.0#

RpmCommand ::= SEQUENCE {
   continueOnFailure [0] NULL OPTIONAL,
   rpmCommandDetails CHOICE {
      enable [1] SEQUENCE {iccid [APPLICATION 26] Iccid},
      disable [2] SEQUENCE {iccid [APPLICATION 26] Iccid},
      delete [3] SEQUENCE {iccid [APPLICATION 26] Iccid},
      listProfileInfo [4] ListProfileInfo,
      updateMetadata [5] SEQUENCE {
         iccid [APPLICATION 26] Iccid,
         updateMetadataRequest UpdateMetadataRequest
      },
      contactPcmp [6] SEQUENCE {
         iccid [APPLICATION 26] Iccid,
         dpiRpm UTF8String OPTIONAL
      }
   }
}
-- ASN1STOP
```

The SM-DP+ SHALL limit the size of the value part of one `RpmPackage` to a maximum of 1057 bytes. The eUICC SHALL be able to handle at least this size. If the eUICC receives an `RpmPackage` too large to handle, it SHALL reject it with error code `commandPackageTooLarge`.

NOTE 1: This allows transporting of one icon with maximum size (1024 bytes), together with the icon type and `continueOnFailure` being set.

NOTE 2: If more RPM Commands need to be sent, an SM-DP+ MAY use rpmPending to chain multiple RSP sessions as defined in section 5.6.3.

The rpmCommandDetails identifies the type of an RPM Command. The following limitations exist:

- In order to simplify handling of the REFRESH proactive command after processing of the RPM Package, the SM-DP+ SHALL restrict Enable and Disable commands in one RPM Package as follows:
  - only one Enable command, or
  - only one Disable command, or
  - only one Disable command followed by only one Enable command.

  The eUICC MAY terminate processing of a subsequent Enable or Disable command in the RPM Package with error commandsWithRefreshExceeded.
- RPM Command 'Contact PCMP' SHALL NOT be followed by any other RPM Command in the same RPM Package. Any subsequent command SHALL be rejected with error commandAfterContactPcmp.
- An RPM Command requesting the enabling or disabling of a Test Profile via RPM SHALL be rejected by the eUICC.

On the execution failure of an RPM Command: if continueOnFailure is present, then the eUICC SHALL continue to execute the next RPM Command in the RPM Package. Otherwise, the eUICC SHALL stop processing the RPM.

The iccid indicates the Target Profile.

RPM Command 'List Profile Info'

listProfileInfo SHALL be coded and processed as defined in section 5.7.15.

RPM Command 'Update Metadata'

updateMetadataRequest indicates the Profile Metadata to be updated. It SHALL be coded and processed as defined in section 5.4.1.

2.10.2 Load RPM Package Result

The RPM Commands in the RPM Package are executed sequentially until the end is reached or an error is encountered for a command where continueOnFailure is not present. Each executed command produces an RPM Command Result.

Atomic processing by the eUICC is required only for each RPM Command.

In case of external interruptions (e.g., power loss), the eUICC MAY be unable to process remaining commands. This SHALL be indicated by placing interruption into the RPM Command Result.

If an RPM Command in the sequence is unknown or unsupported, or the command data cannot be interpreted, the eUICC SHALL terminate processing of the command sequence and return unknownOrDamagedCommand as RPM Command Result.

The eUICC SHALL be able to create a finalResult with a value size of at least 1072 Bytes.

NOTE: This allows for the response to include one icon with maximum size (1024 bytes), together with the icon type and the ICCID.

If the processing of an RPM command would result in exceeding the maximum size the eUICC can handle, independent of the presence of `continueOnFailure`, the eUICC SHALL NOT process this and all subsequent commands and place `resultSizeOverflow` into the RPM Command Result.

The Load RPM Package Result SHALL be returned at the end of executing an RPM Package.

A signed Load RPM Package Result SHALL be kept by the eUICC (which can hold one or several signed Load RPM Package Results) until explicitly deleted by the LPA, after successfully delivered to the SM-DP+. Before being deleted the signed Load RPM Package Result(s) MAY be retrieved at any time by the LPA.

When the eUICC needs to store a new signed Load RPM Package Result and/or Profile Installation Result, if there is not enough room the eUICC SHALL delete one or more of the previously stored signed Load RPM Package Results or Profile Installation Results in order of their Sequence Number, beginning with the lowest.

The Load RPM Package Result SHALL be encoded in the ASN.1 data object as shown below.

```
-- ASN1START
LoadRpmPackageResult ::= [68] CHOICE { -- Tag 'BF44' #SupportedForRpmV3.0.0#
   loadRpmPackageResultSigned LoadRpmPackageResultSigned,
   loadRpmPackageResultNotSigned LoadRpmPackageResultNotSigned
}

LoadRpmPackageResultSigned ::= SEQUENCE {
   loadRpmPackageResultDataSigned LoadRpmPackageResultDataSigned,
   euiccSignRPR EuiccSign
}

LoadRpmPackageResultDataSigned ::= SEQUENCE {
   transactionId [0] TransactionId,
   notificationMetadata[47] NotificationMetadata,
   smdpOid OBJECT IDENTIFIER, -- SM-DP+ OID (value from CERT.DPauth.SIG)
   finalResult [2] CHOICE {
      rpmPackageExecutionResult SEQUENCE OF RpmCommandResult,
      loadRpmPackageErrorCodeSigned LoadRpmPackageErrorCodeSigned
   }
}

RpmCommandResult ::= SEQUENCE { -- #SupportedForRpmV3.0.0#
   iccid [APPLICATION 26] Iccid OPTIONAL, -- SHALL be present, except for
listProfileInfoResult and rpmProcessingTerminated
   rpmCommandResultData CHOICE {
      enableResult [49] EnableProfileResponse, -- ES10c.EnableProfile
      disableResult [50] DisableProfileResponse, -- ES10c.DisableProfile
      deleteResult [51] DeleteProfileResponse, -- ES10c.DeleteProfile
      listProfileInfoResult [45] ProfileInfoListResponse, -- ES10c.GetProfilesInfo
      updateMetadataResult [42] UpdateMetadataResponse, -- ES6.UpdateMetadata
      contactPcmpResult [0] ContactPcmpResponse,
      rpmProcessingTerminated INTEGER {
         resultSizeOverflow (1),
         unknownOrDamagedCommand (2),
         interruption (3),
         commandsWithRefreshExceeded (4),
         commandAfterContactPcmp (5),
         commandPackageTooLarge (6)
      }
   }
}
```

```
ContactPcmpResponse ::= CHOICE {
   contactPcmpResponseOk SEQUENCE {
      pcmpAddress UTF8String
   },
   contactPcmpResponseError INTEGER {
      profileNotEnabled(2),
      commandError(7),
      noLprConfiguration(13),
      undefinedError(127)}
}

LoadRpmPackageResultNotSigned ::= SEQUENCE {
   transactionId [0] TransactionId,
   loadRpmPackageErrorCodeNotSigned LoadRpmPackageErrorCodeNotSigned
}

LoadRpmPackageErrorCodeSigned ::= INTEGER { invalidSignature(2),
invalidTransactionId(5), undefinedError(127)}

LoadRpmPackageErrorCodeNotSigned ::= INTEGER { noSession(4), undefinedError(127)}
-- ASN1STOP
```

euiccSignRPR SHALL be created using the SK.EUICC.SIG selected during the RPM Download procedure, and verified using the related PK.EUICC.SIG as described in section 2.6.9. euiccSignRPR SHALL apply on the concatenated data objects loadRpmPackageResultDataSigned and smdpSignature3.

In case of the error invalidTransactionId, the transactionId in the LoadRpmPackageResultDataSigned SHALL be set to the value from the AuthenticateServerRequest.

dpiRpm contains the DPI which if present in the command data is appended by the LPA to the pcmpAddress received in the response in order to trigger the LPR.

NOTE: Error codes to RPM commands do not disclose any information about the existence of a Profile unless the SM-DP+ is authenticated and authorised.

2.11 Overview of version interoperability

Depending on the interfaces, interoperability between parties implementing different versions of this specification is enabled by SVN and/or RSP capability indication by one party and appropriate reaction by the other party.

For (LPA and eUICC) to RSP Server communication related functions (ES8+, ES9+, ES10a, ES10b, ES11), this specification provides a built-in mechanism for capability indication. Each party compliant with version 3 or higher of this specification has to include its RSP capability. When the RSP capability is absent, it indicates a party implementing a version prior to version 3. This mechanism is described in section 3.0.1 Common Mutual Authentication Procedure and related functions.

For RSP Server to RSP Server communication (ES2+, ES12, ES15), the RSP Server that acts as a client indicates its SVN using the HTTP header X-Admin-Protocol (see section 6.2).

For information, the Device and the LPAd can determine the highestSvn of the eUICC by using the ES10b.GetEUICCInfo function as described in section 5.7.8. The LPAd uses the ES10 functions and associated parameters in line with the EuiccRspCapability for further communication with the eUICC. The eUICC doesn't need to know the lpaSvn of the LPAd and operates according to the LPAd requests.

To communicate on ES6 with the eUICC, the Operator SHALL use the protocol and functions corresponding to the capabilities communicated by the eUICC in `EUICCInfo2` during the Profile Download and Installation.

There is no SVN negotiation on EShri. But the HRI Server can version its services using different URLs.

2.12 Multiple Enabled Profiles

This specification defines optional support of Multiple Enabled Profiles (MEP), where several Profiles can be in Enabled state. This enables a Device with more than one baseband to use more than one Profile at the same time for providing connectivity to different networks.

The multiplexing of the APDU streams to those Profiles on a single physical interface is specified in ETSI TS 102 221 [6]. This specification uses the term "eSIM Port" for what ETSI TS 102 221 [6] calls a logical SE interface (LSI).

eSIM Ports are identified by consecutive numbers starting from zero (and will be written as eSIM Port 0, eSIM Port 1, etc.).

Each eSIM Port SHALL be assigned to at most one Enabled Profile at any point in time. Each Profile SHALL be assigned to at most one eSIM Port at any point in time.

Profile Enabling assigns a Profile to an eSIM Port. Profile Disabling releases this assignment. A Disabled Profile is not assigned to any eSIM Port.

Several options for different features are defined for MEP:

- APDU multiplexing can use any of the mechanisms defined in ETSI TS 102 221 [6], e.g.:
  - eSIM Port selection via the APDU MANAGE LSI (select LSI) when the transmission protocol T=0 or T=1 is used.
  - eSIM Port selection via the NAD byte when the transmission protocol T=1 is used.
- Three options are defined for the selection of the ISD-R and the assignments of eSIM Ports:
  - MEP-A1: The ISD-R is selected on eSIM Port 0 only and Profiles are selected on eSIM Ports 1 and higher, with the eSIM Port being assigned by the LPA. I.e., Command Port and Target Port will always be different.
  - MEP-A2: The ISD-R is selected on eSIM Port 0 only and Profiles are selected on eSIM Ports 1 and higher, with the eSIM Port being assigned by the eUICC. I.e., Command Port and Target Port will always be different.
  - MEP-B: Profiles are selected on eSIM Ports 0 and higher, with the ISD-R being selectable on any of these eSIM Ports. ES10c.EnableProfile and, if CAT is initialised on the Target Port, ES10c.DisableProfile are always sent on the Target Port (i.e., Command Port and Target Port are identical). If CAT is not initialised on the Target Port, ES10c.DisableProfile can be sent on any

eSIM Port. Other ES10 commands can be sent on any eSIM Port where CAT is initialised.

- As already defined in v2 of this specification, two options exist for processing a Profile switch:
  - Profile switching requires a REFRESH proactive command to be issued by the eUICC.
  - Profile switching is performed without such a proactive command.

This specification treats these options independently and an eUICC MAY implement any combination. Limitations for the LPA are given in sections 3.2.1 and 3.2.2.

MEP is only defined for non-removable eUICCs, where the options to be used can be pre-agreed by the OEM and the EUM. Still, an optional setup mechanism is defined in section 3.4.1, which allows Devices and eUICCs to support several modes.

2.13 Overview of Push Service

A Push Service is a service that allows an application server to send push notifications to an application on a Device. This specification allows an SM-DS and LDS to leverage a Push Service so that the LDS is informed in a timely manner when a relevant Event Record is available on the SM-DS.

Figure 8aa shows the entities required for Push Service. A Push Service is implemented by a combination of a push server and a push client on the Device. The interfaces between the push server and the push client, between the push client and the LPAd, and between the SM-DS and the push server, as well as the interactions between them, are out of scope of this specification.

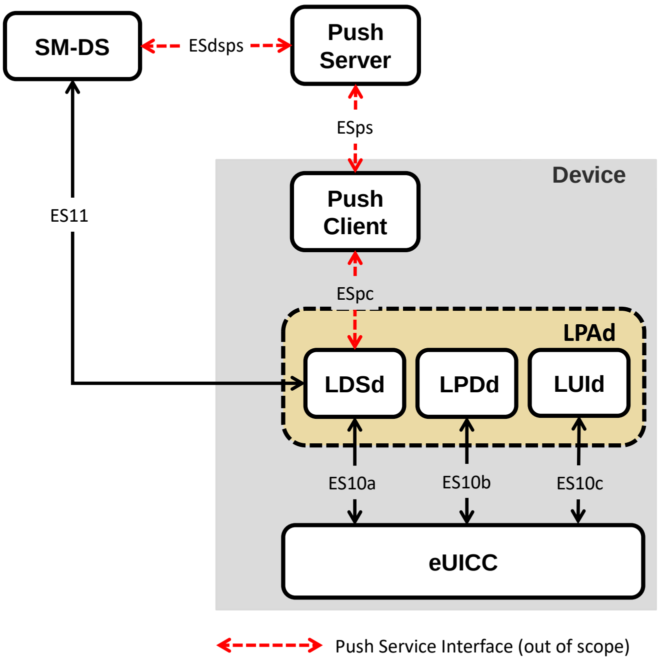


Figure 8aa: Entities for Push Service, LPA in the Device

The SM-DS and a Device MAY support one or more Push Services. The LPAd and SM-DS perform the Push Service registration so that the SM-DS can subsequently send push notifications to the LPAd, following the general sequence that follows:

The SM-DS indicates, during Common Mutual Authentication, one or more Push Service(s) that it supports. If the Device supports at least one of these Push Services, the LPAd requests the corresponding push client on the Device to allow the LPAd to use this Push Service. The push client and its related push server perform an operation to generate a Push Token dedicated to the LPAd. The LPAd then forwards its Push Token together with the EID of its associated eUICC to the SM-DS (see section 3.6.5).

Later, to inform the LPAd that an Event Record is pending for this LPAd on the SM-DS, the SM-DS requests the push server to send a push notification for this Push Token. The push server routes this notification to the push client, which forwards it to the LPAd. The LPAd MAY then perform an Event Retrieval procedure.

A Push Service MAY limit the validity of a Push Token depending on its implementation. Upon expiration of the validity, the LPAd SHOULD re-register a Push Token to the SM-DS to remain able to receive push notifications.

The SM-DS MAY also clean its database of Push Tokens to avoid keeping obsolete Push Tokens. The LPAd and Push Service may not know about when this happens. An SM-DS that enacts such a clean-up policy SHOULD instruct the LPAd of a maximum Push Token retention time. It is the responsibility of the LPAd to register a Push Token, which was previously used or is newly generated, to the SM-DS when this retention time expires.

3 Procedures

This section specifies the procedures associated with Remote SIM Provisioning and Management of the eUICC for consumer Devices.

Some call flows illustrate the case where the LPA is in the Device (LPAd). Such call flows with an LPAe would be identical except that all ES10a, ES10b and ES10c calls become internal to the eUICC and out of scope of this specification.

3.0 Common Procedures

3.0.1 Common Mutual Authentication Procedure

This section describes the common mutual authentication call flow that is used in various others places in this document.

In this section the following notations are used:

- SM-XX denotes either an SM-DP+ or an SM-DS.
- CERT.XXauth.SIG denotes either CERT.DPauth.SIG or CERT.DSauth.SIG.
- SK.XXauth.SIG denotes either SK.DPauth.SIG or SK.DSauth.SIG.
- CERT.XX.TLS denotes either CERT.DP.TLS or CERT.DS.TLS.
- SK.XX.TLS denotes either SK.DP.TLS or SK.DS.TLS.
- ESXX denotes either ES9+ when communicating with an SM-DP+ or an ES11 when communicating with an SM-DS.
- SM-XX SubCA Certificates denote one or more of CERT.CISubCA.SIG, CERT.DPSubCA.SIG, CERT.DSSubCA.SIG, or SubCA Certificates in a trust chain from CERT.XX.TLS to a public RootCA.

This procedure implies the use of CERT.XXauth.SIG. Following this common mutual authentication procedure, if any other Certificates of the SM-XX are used, e.g., the CERT.DPpb.SIG, these Certificates SHALL have a trust chain leading to the same eSIM CA RootCA Certificate as CERT.XXauth.SIG.

```
@startuml
hide footbox
skinparam sequenceMessageAlign center
skinparam sequenceArrowFontSize 11
skinparam noteFontSize 11
skinparam monochrome true
skinparam lifelinestrategy solid

participant "<b>SM-XX" as DP
participant "<b>LPAd" as LPA
participant "<b>eUICC" as E
LPA -> E : [1a] [ES10b.GetEUICCInfo]
E --> LPA : [1b] [euiccInfo1]
rnote over LPA #FFFFFF
[1c] [Restrict the set of public keys in euiccInfo1
to the allowed CI PKId]
end rnote
LPA -> E : [2] ES10b.GetEUICCChallenge
rnote over E #FFFFFF : [3] Generate euiccChallenge
E --> LPA : [4] eUICCChallenge
rnote over DP, LPA #FFFFFF : [5] Establish HTTPS connection
LPA -> DP : [6] ESXX.InitiateAuthentication \n (eUICCChallenge, euiccInfo1, SM-XX
Address, lpaRspCapabiliy)
```

```
rnote over DP #FFFFFF
[7]
- [Verify SM-XX Address]
- Verify euiccInfo1
Endrnote

DP --> LPA : [error]

rnote over DP #FFFFFF
[8]
- Generate TransactionID
- Generate serverChallenge
- Build serverSigned1 = {TransactionID, euiccChallenge,
  serverChallenge, SM-XX Address[, sessionContext]}
- Compute serverSignature1 over serverSigned1
- [Retrieve CRL(s)]
endrnote

DP --> LPA : [9] TransactionID, serverSigned1,
serverSignature1,\neuiccCiPKIdToBeUsed, CERT.XXauth.SIG [, otherCertsInChain][,
crlList]
rnote over LPA #FFFFFF
[10]
- [Verify OID]
- Verify SM-XX Address
- [Verify CI restriction]
- Verify sessionContext
- Generate ctxParams1
endrnote

LPA -> E : [11] ES10b.AuthenticateServer\n(serverSigned1, serverSignature1,\n
euiccCiPKIdToBeUsed, CERT.XXauth.SIG or serverCertChain, ctxParams1
[,otherCertsInChain][, crlList])
rnote over E #FFFFFF
[12]
- Verify server certificate chain
- Verify serverSignature1 over serverSigned1
- Verify serverSigned1
endrnote
E --> LPA : [error]
rnote over E #FFFFFF
[13]
- Generate euiccSigned1 = {TransactionID, serverChallenge,
   euiccInfo2, ctxParams1}
- Compute euiccSignature1 over euiccSigned1
endrnote
E --> LPA : [14] euiccSigned1, euiccSignature1,\n euiccCertificate, nextCertInChain
[, otherCertsInChain]
LPA -> DP : [15] ESXX.AuthenticateClient \n (euiccSigned1, euiccSignature1,\n
euiccCertificate, nextCertInChain[, otherCertsInChain])

rnote over DP #FFFFFF
[16]
- Verify eUICC certificate chain
- Verify euiccSignature1 over euiccSigned1
- Verify euiccSigned1
endrnote
DP --> LPA : [error]
LPA --> E : [ES10b.CancelSession]

rnote over DP, E #FFFFFF : [17] Continue...

@enduml
```

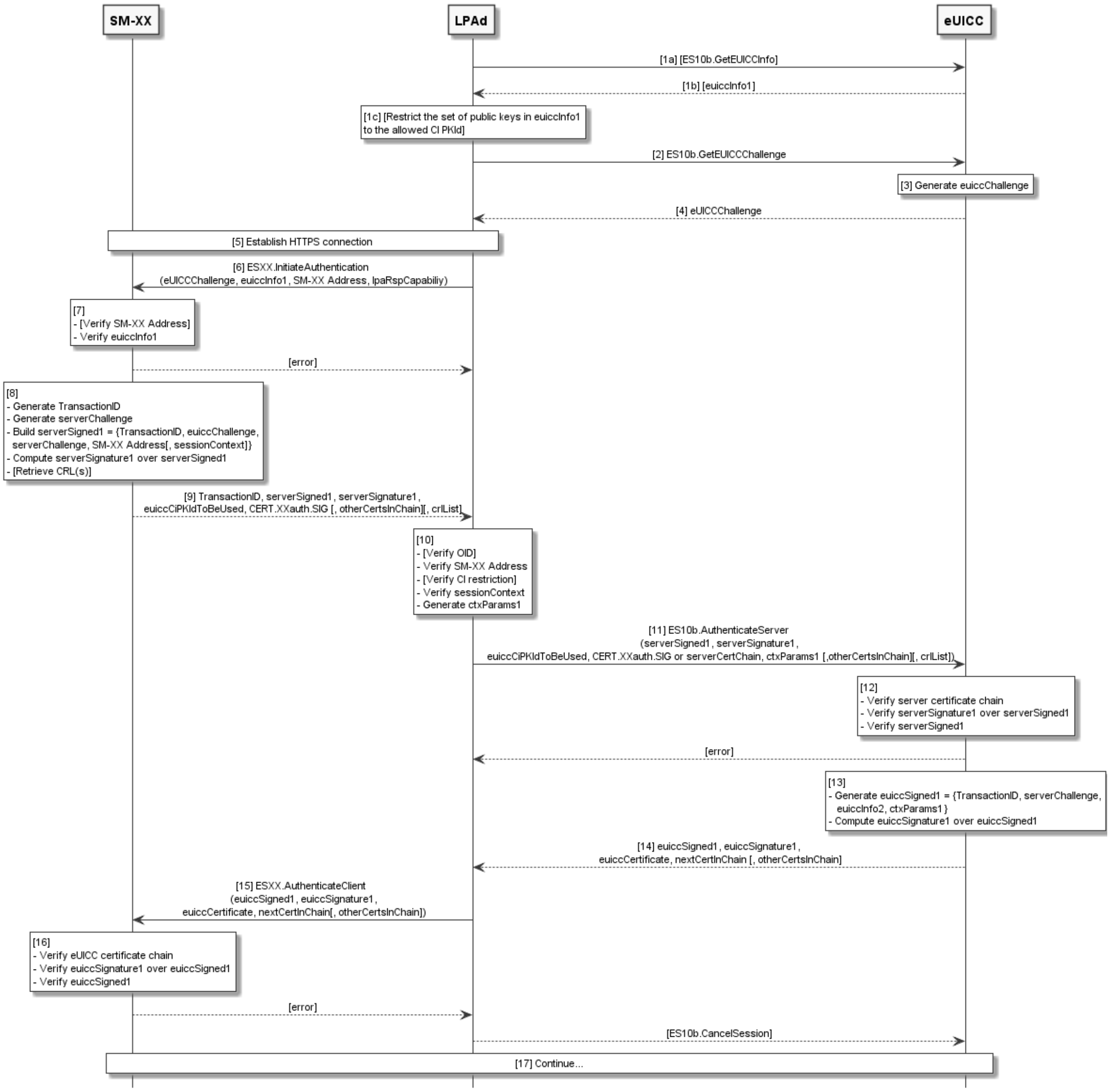

SM-XX
LPAd
eUICC
[1a] [ES10b.GetEUICCInfo]
[1b] [euiccInfo1]
[1c] [Restrict the set of public keys in euiccInfo1 to the allowed CI PKId]
[2] ES10b.GetEUICCChallenge
[3] Generate euiccChallenge
[4] eUICCChallenge
[5] Establish HTTPS connection
[6] ESXX.InitiateAuthentication (eUICCChallenge, euiccInfo1, SM-XX Address, lpaRspCapabiliy)
[7] - [Verify SM-XX Address] - Verify euiccInfo1
[error]
[8] - Generate TransactionID - Generate serverChallenge - Build serverSigned1 = {TransactionID, euiccChallenge, serverChallenge, SM-XX Address[, sessionContext]} - Compute serverSignature1 over serverSigned1 - [Retrieve CRL(s)]
[9] TransactionID, serverSigned1, serverSignature1, euiccCiPKIdToBeUsed, CERT.XXauth.SIG [, otherCertsInChain][, crlList]
[10] - [Verify OID] - Verify SM-XX Address - [Verify CI restriction] - Verify sessionContext - Generate ctxParams1
[11] ES10b.AuthenticateServer (serverSigned1, serverSignature1, euiccCiPKIdToBeUsed, CERT.XXauth.SIG or serverCertChain, ctxParams1 [,otherCertsInChain][, crlList])
[12] - Verify server certificate chain - Verify serverSignature1 over serverSigned1 - Verify serverSigned1
[error]
[13] - Generate euiccSigned1 = {TransactionID, serverChallenge, euiccInfo2, ctxParams1 } - Compute euiccSignature1 over euiccSigned1
[14] euiccSigned1, euiccSignature1, euiccCertificate, nextCertInChain [, otherCertsInChain]
[15] ESXX.AuthenticateClient (euiccSigned1, euiccSignature1, euiccCertificate, nextCertInChain[, otherCertsInChain])
[16] - Verify eUICC certificate chain - Verify euiccSignature1 over euiccSigned1 - Verify euiccSigned1
[error]
[ES10b.CancelSession]
[17] Continue...

Figure 8a: Common Mutual Authentication Procedure

Start conditions:

The SM-XX is provisioned with its Certificate(s) (CERT.XXauth.SIG), its private key(s) (SK.XXauth.SIG), the eSIM CA RootCA Certificate(s) (CERT.CI.SIG), its TLS Certificate(s) (CERT.XX.TLS), its TLS Private Key(s) (SK.XX.TLS), and the SM-XX SubCA Certificates, if any, in the trust chains of its CERT.XXauth.SIG and CERT.XX.TLS Certificates.

The eUICC is provisioned with its Certificate(s) (CERT.EUICC.SIG), its private key(s) (SK.EUICC.SIG), the EUM Certificate(s) (CERT.EUM.SIG), the eSIM CA RootCA SubCA Certificate(s), if any (CERT.CISubCA.SIG), the EUM SubCA Certificate(s), if any (CERT.EUMSubCA.SIG), and the eSIM CA RootCA Public Key(s) (PK.CI.SIG).

The invocation of this procedure conditionally includes a restriction to a single allowed eSIM CA RootCA public key identifier.

Procedure:

1. (a) Optionally, the LPA MAY request eUICC Information euiccInfo1 from eUICC by calling the "ES10b.GetEUICCInfo" function. This is required if the LPAd hasn't already retrieved this information.
1. (b) The eUICC returns the euiccInfo1 to the LPAd.
1. (c) If there is a restriction of the allowed eSIM CA RootCA public key(s), the LPAd SHALL create a new instance of euiccInfo1 for this invocation of the procedure by removing all public key identifiers from euiccCiPKIdListForVerification that do not match the given eSIM CA RootCA public key identifier or indicator. If this process results in an empty list for euiccCiPKIdListForVerification, then the LPAd SHALL inform the End User and the procedure SHALL stop.
2. The LPAd requests an eUICC Challenge from the eUICC by calling the "ES10b. GetEUICCChallenge" function (section 5.7.7).
3. The eUICC SHALL generate an eUICC Challenge which SHALL be signed later by the SM-XX for SM-XX authentication by the eUICC.
4. The eUICC returns the eUICC Challenge to the LPAd.
5. The LPAd establishes a new HTTPS connection with the SM-XX in server authentication mode. The TLS session establishment SHALL perform a new key exchange (it SHALL NOT reuse keys from a previous session). During this step, the LPAd SHALL verify that CERT.XX.TLS is valid as described in section 4.5.2.2. If CERT.XX.TLS is invalid and all retries have been exhausted, the LPAd SHALL stop the procedure. If there is a restriction of the allowed eSIM RootCA public key(s), it SHALL NOT affect the establishment of the HTTPS connection.

NOTE 1: The TLS handshake as defined in RFC 5246 [16] doesn't allow the LPAd to indicate in the "ClientHello" message the list of eSIM CA RootCA public keys it supports for signature verification. Therefore, in a Multiple eSIM CA environment, the SM-XX cannot provide with certainty a CERT.XX.TLS that the LPAd will be able to verify, and the TLS handshake may fail. In that case the LPAd MAY retry the TLS handshake, and the SM-XX MAY select a different CERT.XX.TLS.

NOTE 2: The LPAd MAY use a non-empty session_id in the "ClientHello" as described in RFC 5246 [16].

6. The LPAd SHALL call the "ESXX.InitiateAuthentication" function (sections 5.6.1 and 5.8.1) with its input data comprising the euiccChallenge, euiccInfo1, SM-XX Address and its capability. SM-XX is the Address used by the LPAd to access the SM-XX. The way the SM-XX Address is acquired depends on the procedure where this common call flow is used.

7. The SM-XX SHALL verify that the SM-XX Address sent by the LPAd is valid. If the SM-XX Address is not valid, the SM-XX SHALL return an error.

   The SM-XX SHALL verify the list of eSIM CA RootCA Public Keys that are associated to the eUICC credentials (euiccCiPKIdListForSigning and euiccCiPKIdListForSigningV3 if present, as contained in euiccInfo1). If the SM-XX does not accept any of these eSIM CA RootCA Public Keys it SHALL return an error.

The SM-XX SHALL verify the received eSIM CA RootCA Public Key Identifier list (euiccCiPKIdListForVerification contained in the euiccInfo1). If it cannot provide a CERT.XXauth.SIG which chain can be verified by an eSIM CA RootCA Public Key supported by the eUICC:

- If the LPAd RSP capabilities indicated `euiccCiUpdateSupport`, the SM-XX SHOULD select its preferred CERT.XXauth.SIG.
- In all other cases, it SHALL return an error.

If the LPAd receives an error in this step, then the LPAd SHALL stop the procedure.

8. The SM-XX SHALL perform the following:
   - Generate a TransactionID which is used to uniquely identify the RSP session and to correlate the multiple ESXX request messages that belong to the same RSP session.
   - Generate an SM-XX Challenge (serverChallenge) which SHALL be signed later by the eUICC for the eUICC authentication.
   - Select one eSIM CA RootCA Public Key among those provided within euiccCiPKIdListForSigning or euiccCiPKIdListForSigningV3, that is supported by the RSP Server for signature verification and indicate it in euiccCiPKIdToBeUsed or in euiccCiPKIdToBeUsedV3 respectively.
   - Generate a serverSigned1 data structure.
   - Compute the serverSignature1 over serverSigned1 using the SK.XXauth.SIG corresponding to the CERT.XXauth.SIG determined in step 7.
   - If both eUICC and LPA indicate crlStaplingV3Support, retrieve the latest CRL for each Certificate in the chain that has a `cRLDistributionPoints` extension set (unless it is already available).
9. The SM-XX SHALL return to the LPAd the TransactionID, serverSigned1, serverSignature1, euiccCiPKIdToBeUsed, Certificate(s) and conditionally the CRL(s).
10. The LPAd SHALL:
    - If the SM-XX is an SM-DP+ and if its OID was provided earlier, verify the OID as specified in the procedure where this call flow is used.
    - Verify that the SM-XX Address returned by the SM-XX matches the SM-XX Address that the LPAd has provided in step (6).
    - If there is a restriction to a single allowed eSIM CA RootCA public key identifier, verify that the Subject Key Identifier of the eSIM RootCA corresponding to CERT.XXauth.SIG matches this value.
    - If the LPAd indicated `euiccCiUpdateSupport`, verify that the Subject Key Identifier of the Root Certificate corresponding to CERT.XXauth.SIG is included in `euiccInfo1.euiccCiPKIdListForVerification`. If the verification fails, the LPAd SHALL inform the End User and stop the procedure, after which it MAY perform the eUICC Root Public Key update procedure (section 3.10) indicating that Subject Key Identifier.

- (Optional) Verify that each Certificate in the chain and each CRL in the list (if present) is valid with respect to its time window, i.e., `notBefore` and `thisUpdate` are in the past, and `notAfter` and `nextUpdate` are in the future, with regard to the current time known by the Device.
- If any verification fails, the LPAd SHALL inform the End User and stop the procedure.
- Generate a data structure, `ctxParams1`, to be given to the eUICC to be included in signed data.

11. The LPAd SHALL call "ES10b.AuthenticateServer" function with input data comprising the serverSigned1, serverSignature1, euiccCiPKIdToBeUsed if returned by the SM-XX, CERT.XXauth.SIG, other certificates in the chain if returned by the SM-XX, ctxParams1 and conditionally the CRL(s).

12. The eUICC SHALL:

    - Verify the CERT.XXauth.SIG and other certificates in the chain, if any, starting with CERT.XXauth.SIG, using the relevant PK.CI.SIG.
    - Verify the serverSignature1 performed over serverSigned1.
    - Verify that euiccChallenge contained in serverSigned1 matches the one generated by the eUICC during step (3).
    - Verify that the eSIM CA RootCA Public Key Identifier indicated in either euiccCiPKIdToBeUsed or euiccCiPKIdToBeUsedV3 is supported and related credentials are available for signing.
    - If the sessionContext indicates crlStaplingV3Used, verify the validity of each CRL, and verify that no Certificate in the chain is revoked.

    If any verification fails, the eUICC SHALL return a relevant error status and the procedure SHALL stop.

    If all the verifications succeed, the SM-XX is authenticated by the eUICC.

13. The eUICC SHALL:

    - Generate the euiccSigned1 data structure.
    - Compute the euiccSignature1 over euiccSigned1 using SK.EUICC.SIG. When generating the euiccSignature1, the eUICC SHALL use the credentials identified in the previous step.

14. The eUICC SHALL return the euiccSigned1, euiccSignature1 and the eUICC certificate chain related to the credentials used in the previous step.

15. The LPAd SHALL call the "ESXX.AuthenticateClient" function with input data comprising euiccSigned1, euiccSignature1 and the eUICC certificate chain.

16. On reception of the "ESXX.AuthenticateClient" function call, the SM-XX SHALL:

    - Correlate it with the "ESXX.InitiateAuthentication" function processed in steps (7) and (8), by verifying the two TransactionIDs match.

- Verify that the Root Certificate of the eUICC certificate chain corresponds to the `euiccCiPKIdToBeUsed` or `euiccCiPKIdToBeUsedV3` that the SM-XX selected when executing the "ESXX.InitiateAuthentication" function.
- Verify that the eUICC Certificate chain is valid as described in section 4.5.2.2.
- Verify the euiccSignature1 performed over euiccSigned1 using the PK.EUICC.SIG contained in the CERT.EUICC.SIG.
- Verify that serverChallenge contained in euiccSigned1 matches the one generated by the SM-XX during step (7).
- Verify that the eUICC and LPA RSP capabilities match those received in ESXX.InitiateAuthentication.

If any verification fails, the SM-XX SHALL return a relevant error status to the LPAd.

If all verification succeeds, the SM-XX SHALL return a response comprising the pending RSP operation to the LPAd depending on the procedure within which this procedure is used.

If all verifications succeed but the SM-XX has no pending RSP operation for the eUICC:

- The SM-DP+ SHALL return a status code as specified for the type of RSP operation that was requested.
- The SM-DS SHALL return the TransactionID and an empty list of Event Records in "ES11.AuthenticateClient" response.

If the LPAd receives an error status, or only the TransactionID from the SM-DP+ in this step, then the LPAd SHALL send "ES10b.CancelSession" to the eUICC with a reason `sessionAborted`.

17. This common call flow SHALL be followed by additional steps depending on the procedure within which it is used.

3.0.2 Common Cancel Session Procedure

This section describes the cancel session call flow that is used in various other places in this document.

This procedure can occur due to an error, End User rejection, or timeout at the following steps of the protocol:

- after the response to "ES9+.AuthenticateClient"
- after the response to "ES9+.GetBoundProfilePackage"

The LPAd MAY provide additional places where the End User would be offered to reject the Profile or RPM download.

```
@startuml
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
hide footbox

participant "<b>Operator" as OP
participant "<b>SM-DP+" as DP
participant "<b>LPAd" as LPA
participant "<b>eUICC" as E

LPA -> E : [1] ES10b.CancelSession(TransactionID, reason)

rnote over E
[2]
- Generate euiccCancelSessionSigned = {
   TransactionID, reason}
- Compute euiccCancelSessionSignature
   over euiccCancelSessionSigned
endrnote

E --> LPA : [3] cancelSessionResponseOk

rnote over LPA
[Stop procedure if the reason is sessionAborted]
endrnote

rnote over DP, LPA
[4] [Establish HTTPS connection]
endrnote

LPA -> DP : [5] ES9+.CancelSession(TransactionID, cancelSessionResponseOk)

rnote over DP
[6]
- Retrieve the on-going RSP session
- Verify euiccCancelSessionSignature
- Verify the SM-DP+ OID
endrnote
DP --> LPA : [ERROR]

alt Reason is one of {postponed, timeout}
DP --> LPA : [7] OK
else Otherwise
rnote over DP
[8]
- [Set the Profile in 'Error' state]
- [Delete Event, Refer to Event Deletion section 3.6.3]
endrnote
opt
DP -> OP : [9] ES2+.HandleNotification
OP --> DP : OK
end
DP --> LPA : [10] OK
end

@enduml
```

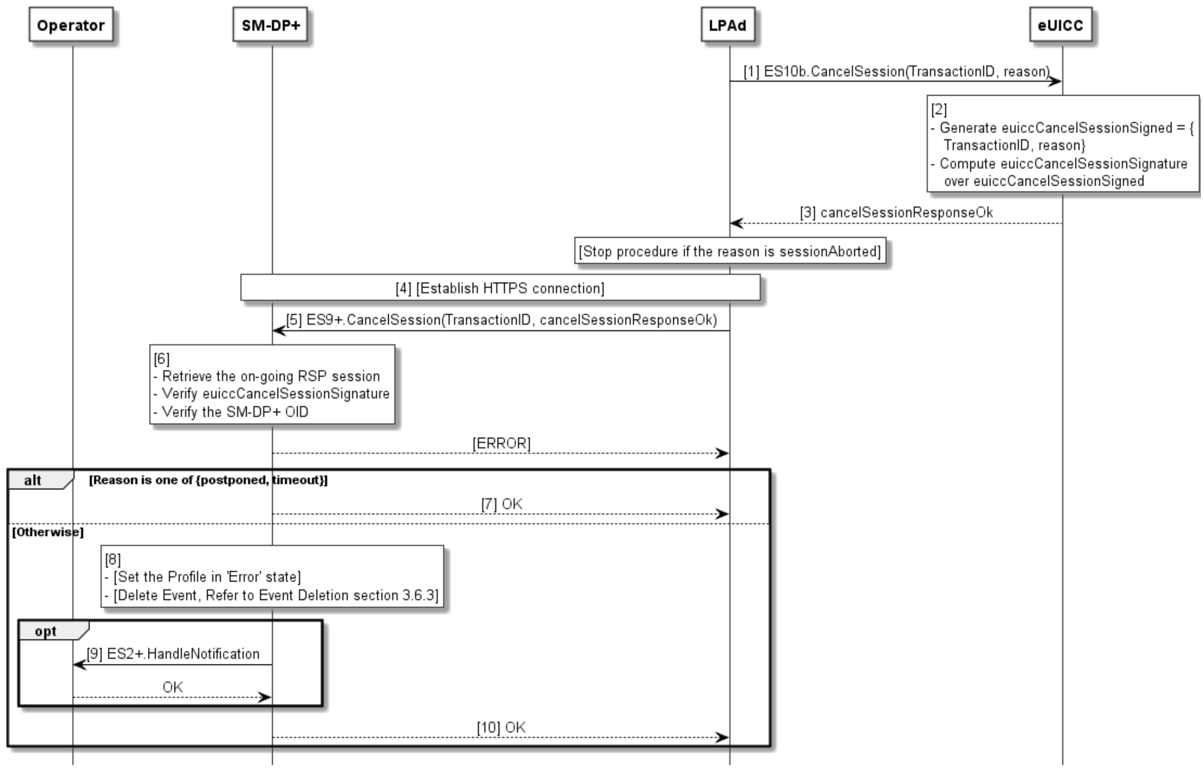

Operator
SM-DP+
LPAd
eUICC
[1] ES10b.CancelSession(TransactionID, reason)
[2]
- Generate euiccCancelSessionSigned = {
TransactionID, reason}
- Compute euiccCancelSessionSignature
over euiccCancelSessionSigned
[3] cancelSessionResponseOk
[Stop procedure if the reason is sessionAborted]
[4] [Establish HTTPS connection]
[5] ES9+.CancelSession(TransactionID, cancelSessionResponseOk)
[6]
- Retrieve the on-going RSP session
- Verify euiccCancelSessionSignature
- Verify the SM-DP+ OID
[ERROR]
alt
[Reason is one of {postponed, timeout}]
[7] OK
[Otherwise]
[8]
- [Set the Profile in 'Error' state]
- [Delete Event, Refer to Event Deletion section 3.6.3]
opt
[9] ES2+.HandleNotification
OK
[10] OK

Figure 8b: Common Cancel Session Procedure

Start Conditions:

This procedure can be used in the following cases.

General reasons in the response to "ES9+.AuthenticateClient":

- The LPAd receives an error in the response to "ES9+.AuthenticateClient". In this case the reason for step (1) SHALL be `sessionAborted`.
- The LPAd receives only the TransactionID in the response to "ES9+.AuthenticateClient". In this case the reason for step (1) SHALL be `sessionAborted`.
- The End User has rejected or postponed to the download of the Profile or the execution of the RPM Command(s) (e.g., by selecting 'No' or 'Not Now'). In these cases the reason for step (1) SHALL be `endUserRejection` or `postponed`, respectively.
- The End User has not responded to the LPAd prompt for User Confirmation within an implementation-dependent timeout interval. In this case the reason for step (1) SHALL be `timeout`.

Cancel reasons after "ES9+.AuthenticateClient" related to Profile download:

- The PPR(s) in the Profile Metadata are not allowed according to the Rules Authorisation Table, or PPR1 is present in the Profile Metadata and an Operational Profile is already installed on the eUICC. In these cases the reason code for step (1) SHALL be `pprNotAllowed`.
- The Profile Metadata in the response to "ES9+.AuthenticateClient" includes an Enterprise Configuration and the eUICC does not support Enterprise Profiles. In this case the reason for step (1) SHALL be `enterpriseProfilesNotSupported`.

- The Profile Metadata in the response to "ES9+.AuthenticateClient" includes Enterprise Rule(s), and the Device is a Non-Enterprise Capable Device, or the End User disallowed the installation of Enterprise Profiles with Enterprise Rules. In this case the reason for step (1) SHALL be `enterpriseRulesNotAllowed`.
- The Profile Metadata in the response to "ES9+.AuthenticateClient" contains an Enterprise Configuration and there is an installed Profile with PPR1 set. In this case the reason for step (1) SHALL be `enterpriseProfileNotAllowed`.
- The Profile Metadata in the response to "ES9+.AuthenticateClient" includes an Enterprise OID that does not match the Enterprise OID of the already installed Enterprise Profile(s). In this case the reason for step (1) SHALL be `enterpriseOidMismatch`.
- The Profile Metadata in the response to "ES9+.AuthenticateClient" includes at least one error in the Enterprise Rule(s). In this case the reason for step (1) SHALL be `enterpriseRulesError`.
- The Reference Enterprise Rule prohibits the installation of non-Enterprise Profiles. In this case the reason for step (1) SHALL be `enterpriseProfilesOnly`.
- The Profile Metadata in the response to "ES9+.AuthenticateClient" includes an LPR Configuration and the Device or the eUICC does not support the LPR. In this case the reason for step (1) SHALL be `lprNotSupported`.
- The Profile Metadata in the response to "ES9+.AuthenticateClient" contains a `serviceProviderName` and/or `profileName` data object with an empty string. If the eUICC and the SM-DP+ both declare `cancelForEmptySpnPnSupport` in respectively the `euiccRspCapability` and the `serverRspCapability`, the reason for step (1) SHALL be `emptyProfileOrSpName`. Otherwise, the reason SHALL be `undefinedReason.`

Cancel reasons after "ES9+.GetBoundProfilePackage":

- The Profile Metadata in the Bound Profile Package does not match the Profile Metadata received previously in the response to "ES9+.AuthenticateClient". In this case the reason for step (1) SHALL be `metadataMismatch`.
- The LPAd has encountered an error while installing a Bound Profile Package. In this case the reason for step (1) SHALL be `loadBppExecutionError`.

Cancel reasons after "ES9+.AuthenticateClient" related to RPM:

- The response to "ES9+.AuthenticateClient" includes an RPM Package and the RPM operation is disabled in the LPA by the End User. In this case the reason for step (1) SHALL be `rpmDisabled`.
- The response to "ES9+.AuthenticateClient" includes an RPM Package and the RPM package violates any of the limitations defined in section 2.10.1. In this case the reason for step (1) SHALL be `invalidRpmPackage`.
- The response to "ES9+.AuthenticateClient" includes an RPM Command 'Contact PCMP' and the LPAd does not support the LPRd. In this case the reason for step (1) SHALL be `lprNotSupported`.
- The response to "ES9+.AuthenticateClient" includes an RPM Command 'Contact PCMP', and the only available data connection is mobile network (cellular) data, and the End User has disallowed use of mobile network data for the LPA Proxy. In this case the reason for step (1) SHALL be `lprNetworkDataNotAllowed`.

- The LPAd has encountered an error while transferring an RPM Package to the eUICC. In this case the reason code for step (1) SHALL be `loadRpmPackageError`.

Procedure:

1. The LPAd SHALL call the "ES10b.CancelSession" function with input data comprising the TransactionID and the reason.

2. The eUICC SHALL:

   - Generate the euiccCancelSessionSigned data object containing the TransactionID and the reason provided by the LPAd.
   - Compute the euiccCancelSessionSignature over euiccCancelSessionSigned using the SK.EUICC.SIG corresponding to the `euiccCiPKIdToBeUsed` as received during the common mutual authentication procedure.

3. The eUICC SHALL return the euiccCancelSessionSigned and euiccCancelSessionSignature. If the reason is `sessionAborted`, the LPAd SHALL ignore the response from the eUICC and stop the procedure.

NOTE: A version 3 or higher LPAd may send the reason `sessionAborted`, which is added in version 3, to a version 2 eUICC.

4. If the HTTPS connection to the SM-DP+ in the RSP session is no longer alive, the LPAd SHALL establish a new HTTPS connection with the SM-DP+ as described in Common Mutual Authentication procedure.

5. The LPAd SHALL call the "ES9+.CancelSession" function with input data comprising TransactionID, euiccCancelSessionSigned and euiccCancelSessionSignature.

6. On reception of the "ES9+.CancelSession" function, the SM-DP+ SHALL:

   - Retrieve the on-going RSP session identified by the TransactionID. If the TransactionID is unknown, the SM-DP+ SHALL return a function execution status 'Failed' with relevant status code and the procedure SHALL stop.
   - Verify the euiccCancelSessionSignature performed over euiccCancelSessionSigned using the PK.EUICC.SIG associated with the ongoing RSP session. If the signature is invalid, the SM-XX SHALL return a function execution status 'Failed' with relevant status code and the procedure SHALL stop.
   - Verify that the received OID is the same value as the one contained in the CERT.DPauth.SIG used during the common mutual authentication procedure. If the value doesn't match, the SM-DP+ SHALL return a function execution status 'Failed' with relevant status code and the procedure SHALL stop.

7. If the reason contained in euiccCancelSessionSigned indicates 'postponed' or 'timeout', the SM-DP+ SHALL simply return a function execution status 'Executed-Success' and keep the Profile download order or RPM download order available for a further retry, and the procedure SHALL stop. If the reason contained in euiccCancelSessionSigned indicates any other condition, the SM-DP+ SHALL perform the following steps.

8. If the on-going RSP session is for Profile download order, the SM-DP+ SHALL set the Profile associated with the on-going RSP session in 'Error' state (section 3.1.6). If an SM-DS was involved in the RSP session identified by the TransactionID, the SM-DP+ SHALL delete the corresponding Event from the SM-DS.
9. Depending on the agreed behaviour with the Operator (out of scope of this specification), the SM-DP+ SHALL call the "ES2+.HandleNotification" function with the relevant notificationEvent set and a notificationEventStatus indicating 'Failed' with status code value depending on the given cancel reason. The cancel session reason mapping to status code is given in section 5.3.5.
10. The SM-DP+ SHALL return a function execution status 'Executed-Success' and the procedure SHALL stop.

3.0.3 RSP Sessions and Error Handling

Several procedures implement RSP Sessions:

- the Profile Download and Installation Procedure (section 3.1.3),
- the RPM Download Procedure together with the RPM Execution Procedure (sections 3.7.2 and 3.7.3),
- the Device Change Procedure (section 3.11.1),
- the Profile Recovery Procedure (section 3.11.2),
- the Event Retrieval Procedure (section 3.6.2), and
- the Push Service Registration Procedure (section 3.6.5).

These procedures comprise a sequence of operations between an RSP Server, the LPA, and the eUICC over a period of time. In addition to errors reported by ES9+, ES11, and ES10 functions, other conditions MAY impact the successful execution of these procedures. The LPA SHOULD indicate such failures to the user; however, the specific presentation of these errors is out of the scope of this document.

An MEP eUICC SHALL only handle one RSP session at any point in time. For MEP-A1 and MEP-A2, RSP sessions SHALL happen on eSIM Port 0. For MEP-B, the LPA MAY select any eSIM Port for an RSP session and the eUICC SHALL accept an RSP session on any eSIM Port. However, the full session SHALL use the same eSIM Port.

The LPA SHOULD NOT initiate a new RSP Session while there is an active RSP Session. However, in the event that this does occur, even on a different eSIM Port in case of MEP-B, when a new RSP session is started with "ES10b.GetEUICCChallenge" the eUICC SHALL discard its session state. Depending upon the specific procedure, this could include the generated eUICC challenge, downloaded Profile Metadata, Profile contents, RPM Package contents, Event Records, a Profile Installation Result, and/or a Load RPM Package Result. However, an unused otPK/otSK.EUICC.KA MAY be stored for future retry.

For MEP-B, in the event that the LPA changes the eSIM Port during an RSP session, the eUICC SHALL return an error and discard its session state.

The eUICC MAY discard its session state if a Profile switch occurs, even on a different eSIM Port, during an RSP Session.

If an eUICC Memory Reset or eUICC Test Memory Reset is successfully processed during a RSP Session, the eUICC SHALL discard its session state.

An RSP Session MAY fail because of a communications failure between the LPA and the RSP Server. The LPA MAY retry for a period of time. The LPA SHALL reset its own session state when all retry attempts have failed.

An RSP Session could fail while the LPA is invoking an ES10 function for reasons other than an error status reported by the eUICC. Examples of such failures include:

- In the case of a removable eUICC card, the End User could remove the card.
- The End User could switch off the power or remove the battery.
- A software fault could cause a crash of the LPA, host Device, and/or baseband processor.

The LPA SHOULD provide an appropriate error indication to the End User when possible (e.g., when power is restored). The specific presentation of such an error notification is out of scope of this document.

3.1 Remote Provisioning

3.1.1 Profile Download Initiation

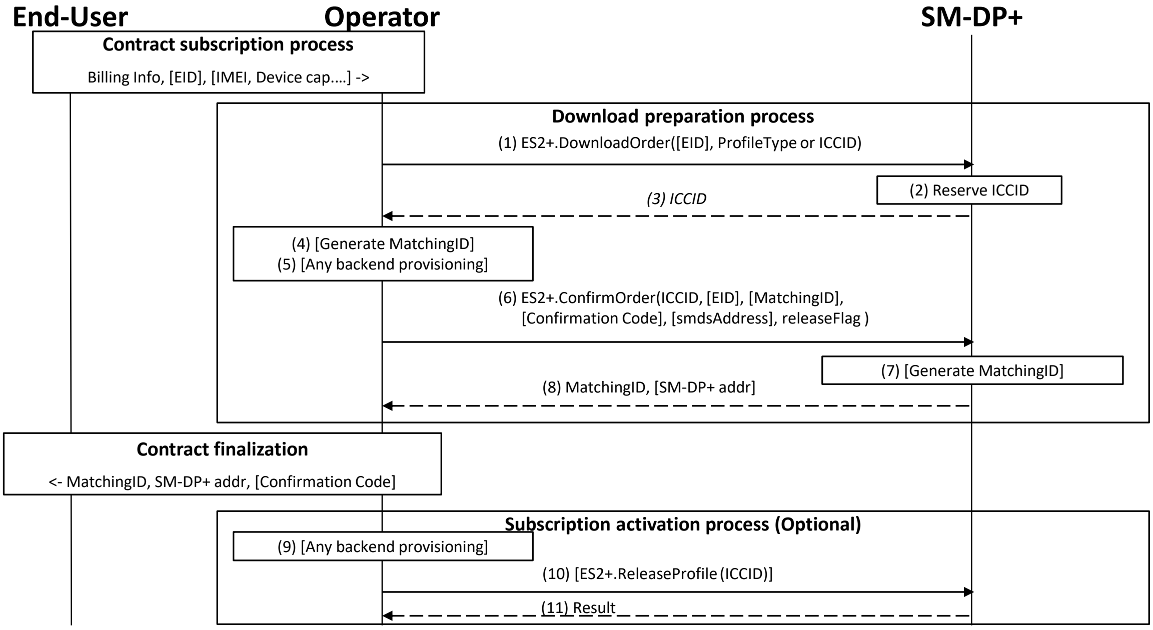

End-User
Operator
SM-DP+
Contract subscription process
Billing Info, [EID], [IMEI, Device cap….] ->
Download preparation process
(1) ES2+.DownloadOrder([EID], ProfileType or ICCID)
(2) Reserve ICCID
(3) ICCID
(4) [Generate MatchingID]
(5) [Any backend provisioning]
(6) ES2+.ConfirmOrder(ICCID, [EID], [MatchingID], [Confirmation Code], [smdsAddress], releaseFlag)
(7) [Generate MatchingID]
(8) MatchingID, [SM-DP+ addr]
Contract finalization
<- MatchingID, SM-DP+ addr, [Confirmation Code]
Subscription activation process (Optional)
(9) [Any backend provisioning]
(10) [ES2+.ReleaseProfile (ICCID)]
(11) Result

Figure 9: Profile Download Initiation

Start Conditions:

The End User has selected the Operator with whom to sign a contract.

The End User MAY initiate the process:

- From any other Device (e.g., PC)

- Through a Customer Agent of the Operator
- Or any other convenient means provided by the Operator

Procedure:

The download initiation procedure consists of the following sub-processes:

A. Contract subscription process
B. Download preparation process
C. Contract finalization process
D. Subscription activation process (Optional)

NOTE: This section describes the case where these sub-process are performed in the described order. In this case, it is most likely that the download and installation procedure will happen right after this procedure. There also are cases where these sub-processes MAY be performed in different order like B -> A -> C [-> D] or B -> C -> A [-> D] (e.g., for prepaid Subscription). In these cases the download order requested from the SM-DP+ MAY remain pending for a significant amount of time.

NOTE: The following table summarizes the input data to be provided in "ES2+.DownloadOrder" and "ES2+.ConfirmOrder" functions.

ES2+ Function	Input Parameters	Profile Download Use Cases		
		Default SM-DP+	Activation Code	SM-DS
DownloadOrder (Section 5.3.1)	eid	O	O	O
	iccid	O	O	O
	profileType	C (1)	C (1)	C (1)
ConfirmOrder (Section 5.3.2)	iccid	M	M	M
	eid	C (2)	O	C (2)
	matchingId	M (3)	O (4)	O (5)
	confirmationCode	O	O	O
	smdsAddress	Not Present	Not Present	C (6)
	rootSmdsAddress	Not Present	Not Present	C (6)
	releaseFlag	M	M	M
NOTE 1: Required if iccid is not present for DownloadOrder NOTE 2: Required if it is not present for DownloadOrder NOTE 3: Value SHALL have zero-length NOTE 4: If present, value SHALL have non-zero-length. If not present, SM-DP+ generates matchingId for ActivationCodeToken NOTE 5: If present, value SHALL have non-zero-length. If not present, SM-DP+ generates matchingId. It is used as EventID NOTE 6: One or both of smdsAddress and rootSmdsAddress SHALL be present				

Table 6a: "ES2+.DownloadOrder" and "ES2+.ConfirmOrder" input data

As the address of a Default SM-DP+ is pre-configured on the eUICC or Device, no MatchingID is required. However, in a typical deployment, an SM-DP+ is expected to handle all kinds of requests, i.e., those triggered via SM-DS or Activation Code, as well as those in the role of a Default SM-DP+. Plus an Operator MAY decide to specify the use of MatchingIDs even if the Default SM-DP+ address has been pre-configured on the Devices. To cover all such cases, this specification defines a tolerant behaviour, where the SM-DP+ will properly handle requests on ES9+ which target a Default SM-DP+ (i.e., no Matching ID is provided), even if a MatchingID has been generated for the eUICC.

3.1.1.1 Contract Subscription Process (Informative)

The contract selection process, while being out of scope of this specification, is given as it SHALL happen prior to the Profile download and installation procedure (section 3.1.3). This process description describes the information exchanged and data that are used as input data for the Profile download and installation procedure.

This process can be performed at an Operator's Point of Sale (POS), using the Operator's web portal from a Device which is not the one onto which the Profile will be downloaded (e.g., a PC) or from a web browser on the Primary Device, or even using a companion application on the Primary Device. Any other mean defined by the Operator can also be possible as far as it provides a convenient End User experience and it provides the expected output data required for the execution of the Profile download and installation procedure.

During the execution of the process of contract Subscription, the Operator acquires the necessary information, such as through acquisition of its Device Information Code. As part of this data, the EID and IMEI of the target Device MAY be provided, and related Device capabilities MAY be acquired (e.g., based on the TAC information comprised in the IMEI). Acquisition and verification of these capabilities are out of scope of this specification. The Root SM-DSs supported by the Device MAY also be acquired. Additional information such as contract details, user details, payment details and similar are also out of scope of this specification.

If the EID and the IMEI are provided, the Operator can verify if the target Device (both eUICC and Device can be relevant for this verification) is supported, and determine the Profile Type for the target Device and the offer given to the End User. If no information about the target Device is provided, this preliminary verification cannot be performed and it will be performed during the execution of the Profile download and installation procedure (section 3.1.3). For additional info see Annex F on Profile eligibility check.

If EID and IMEI are provided and the Operator cannot provide an appropriate Profile, the process fails and stops at this point.

3.1.1.2 Download Preparation Process

1. The Operator calls the "ES2+.DownloadOrder" (section 5.3.1) function of the SM-DP+ with the relevant input data.

   'EID' is optional. If an SM-DS or a Default SM-DP+ is to be used for the Profile download, then the EID SHALL be present. One of the value 'ProfileType' or 'ICCID' SHALL be provided. If ICCID is given, the SM-DP+ SHALL verify that this ICCID is available. If 'ProfileType' is given, the SM-DP+ SHALL pick one of the related ICCID in its inventory.

The SM-DP+ MAY optionally verify additional compatibility between the eUICC (if EID is provided) and the requested Profile Type. This verification is out of scope of this specification.

2. The SM-DP+ reserves the ICCID for this request. At this stage the SM-DP+ MAY simply pick the related Protected Profile Package from its inventory or generate and protect the Profile corresponding to this ICCID.
3. The SM-DP+ returns the acknowledged ICCID (SHALL be the same value as the received one, if any).
4. Optionally, the Operator MAY generate a MatchingID (section 4.1.1). If a Default SM-DP+ is to be used for the Profile download, then the Operator MAY send an empty string in the MatchingID value field.
   At this stage the Operator knows the ICCID selected for this contract Subscription. If an error occurs during this step, the process fails and stops at this point.
5. to 8. The Operator MAY perform any relevant operation on its back-end (e.g., provisioning of HLR). The Operator SHALL confirm the download order by calling the "ES2+.ConfirmOrder" (section 5.3.2) function of the SM-DP+ with the ICCID and its relevant input data.

   - If EID is available, the EID SHALL be included in the input data. If the EID was provided with previous "ES2+.DownloadOrder", the same EID SHALL be provided.
   - If a MatchingID was generated by the Operator in Step 4 or if the Operator provides a zero-length MatchingID, it SHALL be included in the input data and then the SM-DP+ SHALL return the acknowledged value that is the same as the received one. Otherwise, the SM-DP+ SHALL generate a MatchingID and return the generated value to the Operator. The ICCID SHALL be associated to the MatchingID.
   - If it is required for the End User to enter the Confirmation Code to download the Profile, the Confirmation Code SHALL be included in the input data of the "ES2+.ConfirmOrder" (section 5.3.2) function.
   - The Operator MAY send one or two SM-DS addresses to the SM-DP+ as defined in section 3.6.1. If SM-DS address(es) are given, the SM-DP+ SHALL perform Event Registration to the specified SM-DS(s).
   - If all necessary operations on Operator's back-end provisioning has been completed by this point, releaseFlag SHALL be set to 'true' in the input data. Otherwise, releaseFlag SHALL be set to 'false' and additional "ES2+.ReleaseProfile" function SHALL be called later in Subscription activation process.
   - The SM-DP+ SHALL store the functionRequesterIdentifier and functionCallIdentifier values of the "ES2+.ConfirmOrder" function call, which SHALL be used as notificationReceiverIdentifier and notificationIdentifier, respectively, in subsequent "ES2+.HandleNotification" calls related to this order.

   The SM-DP+ MAY return an SM-DP+ address value. In this case the Operator SHALL use this value to generate the Activation Code; otherwise the Default SM-DP+ address SHALL be used.

NOTE: If no EID is given at this stage, the Operator MAY be involved later during the download and installation procedure to determine the right 'ProfileType'/'ICCID' in case the provided 'ProfileType'/'ICCID' is not compatible with the eUICC identified by the EID once it is acquired by SM-DP+ during the download and installation procedure. See Annex F on Profile eligibility check.

3.1.1.3 Contract Finalization (Informative)

The Operator provides the End User with relevant information necessary for the Profile download.

If the Activation Code is to be used for the Profile download, the MatchingID and SM-DP+ address are provided via the Activation Code as described in section 4.1. If the optional Confirmation Code is to be used, it is provided to the End User separately from the Activation Code.

If an SM-DS or Default SM-DP+ is to be used for the Profile download, the Operator informs the End User of the condition that triggers the Profile download procedure, e.g., the very first boot-up and/or IP connection of the device.

3.1.1.4 Subscription Activation Process (Optional)

It is most likely that the Operator backend provisioning can be performed during the download preparation process. But if it cannot be performed, the Subscription activation process can be performed as a separate process to decouple the download preparation processes and Contract finalization process.

9. If all necessary operations on its back-end (e.g., provisioning of HLR) were not performed in Step 5, they SHALL be performed in this stage.
10. The Operator calls the "ES2+.ReleaseProfile" function of the SM-DP+ with ICCID to release the Profile to allow the download and installation procedure to be started by the End User. If the download and installation procedure is initiated by the End User before this function call, the download and installation procedure SHALL NOT be allowed and SHALL return a specific error code.
11. The SM-DP+ SHALL return the result.

3.1.2 Void

Figure 10: Void

3.1.3 Profile Download and Installation

This section describes the Profile download and installation procedure.

```
@startuml
hide footbox
skinparam sequenceMessageAlign center
skinparam sequenceArrowFontSize 11
skinparam noteFontSize 11
skinparam monochrome true
skinparam lifelinestrategy solid

participant "<b>Operator" as OP
participant "<b>SM-DP+" as DP
participant "<b>LPAd" as LPA
participant "<b>eUICC" as E

rnote over LPA #FFFFFF
[1] (a) Get SM-DP+ Address, Parse Activation Code Token, [SM-DP+ OID], [CI PK ind.]
from AC, or
    (b) Get SM-DP+ Address and EventID from SM-DS, or
    (c) Get Default SM-DP+ Address, [CI PKID] from eUICC or Device
End rnote

rnote over DP, E #FFFFFF : [2] [Refer to Common mutual authentication procedure
section 3.0.1]

rnote over DP #FFFFFF
[3]
- Look for Profile download pending order
- Eligibility Check using Device Info, euiccInfo2
endrnote

Group Opt.
DP -> OP : [4] ES2+.HandleNotification(...)
OP --> DP : OK
end
DP --> LPA : [error]

rnote over DP #FFFFFF
[5]
- Check if download retry
- Build smdpSigned2 = {TransactionID,
   Confirmation Code Required Flag, [bppEuiccOtpk], [rpmPending]}
- Compute smdpSignature2 over smdpSigned2 and euiccSignature1
- Build Profile Metadata
endrnote
DP -> LPA : [6] TransactionID, Profile Metadata, smdpSigned2, smdpSignature2,
CERT.DPpb.SIG

rnote over LPA #FFFFFF
[7] Check if the Profile can be installed
endrnote

Opt If required by LPAd
LPA -> E : [ES10b.GetRAT]
E --> LPA :  [RAT]

LPA -> E : [ES10c.GetProfilesInfo]
E --> LPA :  [ProfileInfoListOk]
end

rnote over LPA #FFFFFF
[8] [End User consent(s) with
optional Confirmation Code input]
Endrnote

alt Download rejection
     rnote over OP, E #FFFFFF : [Refer to Common Cancel Session procedure section
3.0.2]

else Download confirmation
     rnote over OP, E #FFFFFF : [Refer to Sub-procedure Profile Download and
installation – Download confirmation]

end

@enduml
```

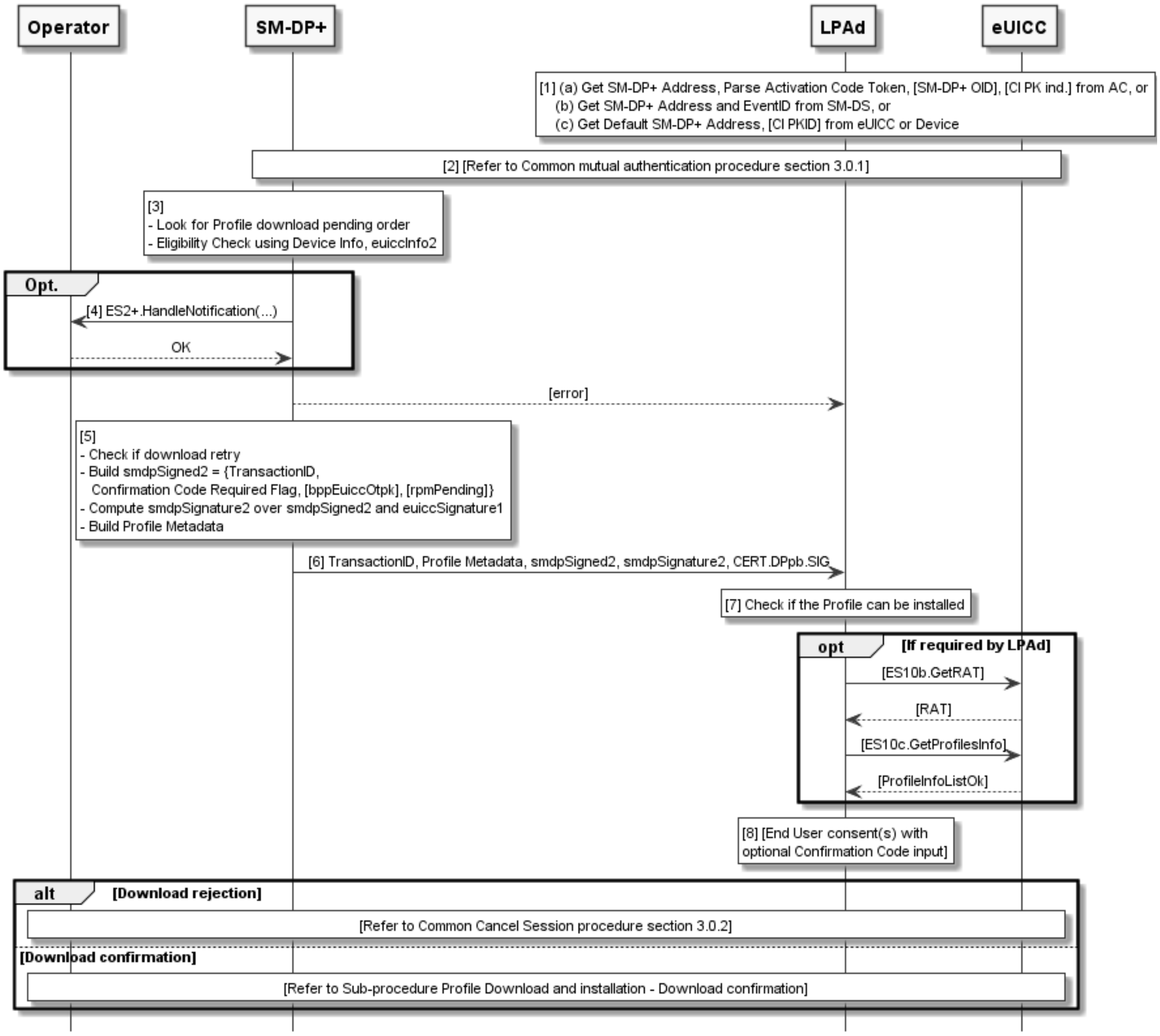

Operator
SM-DP+
LPAd
eUICC
[1] (a) Get SM-DP+ Address, Parse Activation Code Token, [SM-DP+ OID], [CI PK ind.] from AC, or
(b) Get SM-DP+ Address and EventID from SM-DS, or
(c) Get Default SM-DP+ Address, [CI PKID] from eUICC or Device
[2] [Refer to Common mutual authentication procedure section 3.0.1]
[3]
- Look for Profile download pending order
- Eligibility Check using Device Info, euiccInfo2
Opt.
[4] ES2+.HandleNotification(...)
OK
[error]
[5]
- Check if download retry
- Build smdpSigned2 = {TransactionID,
Confirmation Code Required Flag, [bppEuiccOtpk], [rpmPending]}
- Compute smdpSignature2 over smdpSigned2 and euiccSignature1
- Build Profile Metadata
[6] TransactionID, Profile Metadata, smdpSigned2, smdpSignature2, CERT.DPpb.SIG
[7] Check if the Profile can be installed
opt
[If required by LPAd]
[ES10b.GetRAT]
[RAT]
[ES10c.GetProfilesInfo]
[ProfileInfoListOk]
[8] [End User consent(s) with
optional Confirmation Code input]
alt
[Download rejection]
[Refer to Common Cancel Session procedure section 3.0.2]
[Download confirmation]
[Refer to Sub-procedure Profile Download and installation - Download confirmation]

Figure 11: Profile Download and Installation

Start Conditions:

In addition to the start conditions required by the common mutual authentication procedure defined in section 3.0.1, this procedure requires the following start conditions depending on options in step 1:

- If this procedure uses an Activation Code (option a):
    - The End User has an Activation Code that is coded as described in the section 4.1.
    - The End User has entered the Activation Code to the LPAd. Depending on the Device capabilities, the LPAd SHALL support entry of the Activation Code by manual typing and QR code scanning.
    - If the Activation Code specifies an eSIM CA RootCA Public Key indicator, the LPAd SHALL restrict the allowed eSIM CA RootCA public key identifiers to that value.
- If this procedure uses an SM-DS (option b):

  - The LPAd has retrieved an SM-DP+ Address and EventID from the SM-DS.
  - If there was a restriction of the eSIM CA RootCA public key identifier for the SM-DS procedure, the LPAd SHALL apply the same restriction for the Profile download and installation procedure.

- If this procedure uses a Default SM-DP+ (option c):

  - The LPAd has retrieved the Default SM-DP+ Address and optionally an allowed eSIM CA RootCA public key identifier from the eUICC by calling the "ES10a.GetEuiccConfiguredData" function or from the Device in an implementation-dependent manner. If the retrieved data includes an allowed eSIM CA RootCA public key identifier, then the LPAd SHALL restrict the allowed eSIM CA RootCA public key identifiers to that value.

Further, for each Profile in Released state the SM-DP+ SHALL maintain a count of the number of attempts to download that Profile and a count of the number of attempts to enter the Confirmation Code during download of that Profile. The SM-DP+ SHALL limit the number of download attempts and the number of Confirmation Code attempts, respectively.

A Provisioning Profile MAY be enabled by the LPAd upon End User request for RSP operations as defined in SGP.21 [4], which SHALL include End User consent if an Operational Profile is to be disabled and if establishment of the connectivity using the currently Enabled Profile is not successful.

Finally, if there is already an enabled Profile with PPR1 set, the following has occurred: The End User has been advised of this condition and has given consent for download. The LPA MAY alternatively request this consent at any later point during the download procedure.

Procedure:

1. (Optionally for option (a)) The LPAd parses the Activation Code and finds the SM-DP+ address, Activation Code Token, and optional SM-DP+ OID, and optional eSIM CA RootCA Public Key identifier. If the format of the Activation Code is invalid, the procedure SHALL stop with an error message provided by the LPAd to the End User.
2. The common mutual authentication procedure defined in section 3.0.1 SHALL be executed, conditionally restricting the allowed eSIM CA RootCA public key identifiers as described in the Start Conditions above. In this procedure, SM-XX is SM-DP+. CERT.XXauth.SIG, PK.XXauth.SIG and SK.XXauth.SIG are CERT.DPauth.SIG, PK.DPauth.SIG and SK.DPauth.SIG respectively. ESXX is ES9+.

   During the common mutual authentication procedure at step (1), if an Activation Code is used and it includes an eSIM CA RootCA Public Key indicator, the LPAd verifies that it matches one in the list of supported eSIM CA RootCA Public Key Identifiers in eUICCInfo1, and if the verification fails, the LPAd stops the Profile download procedure. After stopping the download procedure, if LPAd and eUICC both support updating the set of eSIM CA RootCA Public Keys on the eUICC, then the LPAd MAY perform the eUICC Root Public Key update procedure (section 3.10) indicating the eSIM CA RootCA Public Key indicator in the Activation Code.

During the common mutual authentication procedure at step (10), the LPAd SHALL verify that the SM-DP+ OID contained in the CERT.DPauth.SIG is identical to the SM-DP+ OID if the LPAd has acquired it from the Activation Code at step (1). If the verification fails, the LPAd SHALL inform the End User, send "ES10b.CancelSession" to the eUICC with a reason `sessionAborted`, and the procedure SHALL stop.

During the common mutual authentication procedure at step (10), the LPAd SHALL build the ctxParams1 data object to provide the MatchingID, MatchingID Source, Device Info and optionally operationType (when the eUICC is version 3 or higher), to the eUICC for signature. operationType, if present, SHALL include 'profileDownload'. Only when the eUICC indicates `EuiccRspCapability.serviceProviderMessageSupport`, the Device Info MAY include preferred languages for End User-readable messages from the SM-XX. The value of the MatchingID and MatchingID Source SHALL be set as follows:

- If an Activation Code is used, the MatchingID value SHALL be set to Activation Code Token and the MatchingID Source value SHALL be set to 'activationCode'.
- If an SM-DS is used, the MatchingID value SHALL be set to EventID and the MatchingID Source value SHALL be set to the OID of the SM-DS that provided the EventID.
- If a Default SM-DP+ is used, the MatchingID SHALL be missing and the MatchingID Source value SHALL be set to 'none'.

3. The SM-DP+ SHALL:

   - Verify that there is a related pending Profile download order for the provided MatchingID.
   - If this Profile download order is already linked to an EID, verify that it matches the EID of the authenticated eUICC.
   - Verify that the Profile corresponding to the pending Profile download order is in 'Released' state, or, in case of a retry due to a previous installation failure, in 'Downloaded' state (section 3.1.6).

   If any of these verifications fail, the SM-DP+ SHALL return a relevant error status and the procedure SHALL stop. Otherwise, the SM-DP+ SHALL:

   - Increment the count of download attempts for the identified Profile. If the maximum number of attempts has been exceeded, the SM-DP+ SHALL terminate the corresponding Profile download order and notify the Operator by calling the "ES2+.HandleNotification" function with the `notificationEventStatus` indicating 'Failed' with the relevant error status, and the procedure SHALL stop.
   - Perform appropriate eligibility checks.

4. (Optional step) Depending on the agreed behaviour with the Operator (out of scope of this specification), the SM-DP+ SHALL notify the Operator with the outcome of the eligibility check using the function "ES2+.HandleNotification" with the `notificationEvent` indicating 'Eligibility an attempt limit check'.

NOTE: This notification step MAY be done asynchronously.

5. If the eligibility check fails, the SM-DP+ SHALL:

- Set the Profile corresponding with the pending Profile download order in 'Error' state (section 3.1.6).

- Return an error status to the LPAd and the procedure SHALL stop.

Otherwise, the SM-DP+ SHALL:

- Determine if a Confirmation Code is required for this pending order.
- Determine whether the Profile is already bound to the EID from a previous unsuccessful download attempt. If so, the SM-DP+ MAY indicate in its response the otPK.EUICC.KA it wants to use.
- Determine if an RPM Package for the `eid` is also pending.
- Generate an `smdpSigned2` data structure containing associated data elements.
- Compute the `smdpSignature2`.
- Generate the Profile Metadata of the Profile.

6. The SM-DP+ returns "ES9+.AuthenticateClient" response to the LPAd.

7. On reception of the SM-DP+ response, the LPAd SHALL check if the Profile can be installed as described hereunder. For this check, the LPAd MAY use previously fetched Rules Authorisation Table and/or list of installed Profiles. If the LPAd has not already fetched the required information, the LPAd SHALL request those from the eUICC by calling the "ES10b.GetRAT" and/or "ES10c.GetProfilesInfo" functions.

- If the Profile Metadata contains PPR(s), the LPAd SHALL check if the PPR(s) are allowed based on the Rules Authorisation Table. If one or more PPR(s) are not allowed, the LPAd SHALL perform the Common Cancel Session procedure with reason `pprNotAllowed` (See section 2.9.2.1 for PPRs allowed for an eUICC supporting MEP).
- If the Profile Metadata contains PPR1, and an Operational Profile is installed, the LPAd SHALL perform the Common Cancel Session procedure with reason `pprNotAllowed`.
- If the Profile Metadata contains an Enterprise Configuration and there is a Profile with PPR1 set, the LPAd SHOULD perform the Common Cancel Session procedure with reason code 'enterpriseProfileNotAllowed'.
- If the Profile Metadata contains any Enterprise Rule and the Device is a Non-Enterprise Capable Device, the LPAd SHOULD perform the Common Cancel Session procedure with reason `enterpriseRulesNotAllowed`.
- If the Profile Metadata contains any Enterprise Rule and the End User disallowed the installation of Enterprise Profile with Enterprise Rules, the LPAd SHALL perform the Common Cancel Session procedure with reason `enterpriseRulesNotAllowed`.
- If the Profile Metadata contains an Enterprise Configuration and there is a Profile with PPR1 set, the LPAd SHOULD perform the Common Cancel Session procedure with reason `enterpriseProfileNotAllowed`.
- If the Profile Metadata contains an Enterprise OID which does not match the Enterprise OID of the already installed Enterprise Profile(s) (if any), the LPAd

SHOULD perform the Common Cancel Session procedure with reason `enterpriseOidMismatch`.

- If the Profile Metadata contains Enterprise Rules with the referenceEnterpriseRule bit set, the LPAd SHOULD perform the Common Cancel Session procedure with reason `enterpriseRulesError`.

NOTE: The referenceEnterpriseRule bit may be assigned to an Enterprise Profile via RPM after its installation. It may be required to first enable the Enterprise Profile in order for the Enterprise Rule update to succeed.

- For an Enterprise Capable Device, if the Reference Enterprise Rule prohibits the installation of non-Enterprise Profile and the ProfileMetadata does not contain Enterprise Configuration, the LPAd SHOULD perform the Common Cancel Session procedure with reason `enterpriseProfilesOnly`.
- If the Profile Metadata contains an LPR Configuration and the Device or the eUICC does not support the LPR, the LPAd SHOULD perform the Common Cancel Session procedure with reason `lprNotSupported`.
- If the Profile Metadata contains an empty string `profileName` and/or `serviceProviderName`, the LPAd MAY perform the Common Cancel Session procedure with reason `emptyProfileOrSpName` if `cancelForEmptySpnPnSupport` is supported by both the SM-DP+ and the eUICC or with reason `undefinedReason` otherwise.

8. If the Profile Metadata contains Profile Policy Rule(s) and/or Enterprise Rule(s) subject to End User consent, the LPAd SHOULD ask for Strong Confirmation by showing relevant information. This information (the "LPA-generated message") MAY include the consequences of the Profile Policy Rule(s) and/or the Enterprise Rule(s) to the End User, if the Profile includes PPR(s) subject to additional End User consent according to the RAT and/or the installation of the Enterprise Profile requires its immediate enabling. The LPA-generated message SHALL be formulated in a descriptive and non-discriminatory manner (e.g., for Enterprise Profile with "Non-Delete" Profile Policy Rule: "The profile that you are about to install can be deleted only under the terms you have agreed with your service provider, and it will be automatically enabled after installation. Enter your PIN to approve installation?"). If the Profile Metadata includes a Service Provider message, the LPAd MAY also display that message. The LPAd SHOULD display this information to the End User such that it can be read in full under the control of the End User. If the Profile Metadata contains neither Profile Policy Rule(s) nor Enterprise Rule(s), the LPAd SHALL ask for Simple Confirmation (e.g., simple 'Yes' or 'No' or 'Not Now') on the Profile download.

   If the Confirmation Code Required Flag is set in either the Activation Code Token or in the `smdpSigned2`, then the LPAd SHALL ask for the End User to enter the Confirmation Code which was provided by the Operator that MAY be considered as a Strong Confirmation. If the Confirmation Code is not required, the LPAd SHALL ask for Simple Confirmation (e.g., simple 'Yes' or 'No' or 'Not Now') on the Profile download.

The LPAd SHOULD ask for the Simple Confirmation on enabling the Profile to be downloaded after its successful installation (e.g., "Do you want to automatically enable the profile after installation YES/NO?"). This request MAY be prompted at any point in time of the Profile Download and Installation procedure, or immediately afterwards.

The Confirmation Requests described above MAY:

- display ProfileName or any relevant information contained in the Profile Metadata to the End User.
- be combined, if prompted, into a single prompt with the highest Confirmation Level therefore requiring a single confirmation by the End User.
- be performed either at this step or after the BPP has been downloaded by the LPAd (see section 3.1.3.2 step (12)), since the same Profile Metadata will also be available then. However, this exception does not apply to Confirmation Code input which SHALL be obtained no later than this step.

If the End User does not agree to the download of the Profile (e.g., by selecting 'No' or 'Not Now') on the Profile download, the LPAd SHALL continue with the Common Cancel Session procedure with reason `endUserRejection` or `postponed`. If a notification is sent to the Operator, the `notificationEvent` SHALL be set to 'Confirmation Failure'.

If the End User does not respond to the LPAd prompt within an implementation-dependent timeout interval, the LPAd SHALL cancel the Profile download by performing the Common Cancel Session procedure with the reason `timeout`.

If required, the LPAd SHALL calculate the hash of the UTF-8-encoded representation of the Confirmation Code as follows:

Hashed Confirmation Code = SHA256 (SHA256 (Confirmation Code) | TransactionID), where '|' means concatenation of data.

If Profile download has not been rejected in the steps above, the procedure SHALL continue with the Sub-procedure "Profile Download and installation – Download confirmation"

3.1.3.1 Void

Figure 12: Void

3.1.3.2 Sub-procedure Profile Download and Installation – Download Confirmation

```
@startuml
hide footbox
skinparam sequenceMessageAlign center
skinparam sequenceArrowFontSize 11
skinparam noteFontSize 11
skinparam monochrome true
skinparam lifelinestrategy solid

participant "<b>Operator" as OP
participant "<b>SM-DP+" as DP
participant "<b>LPAd" as LPA
participant "<b>eUICC" as E
```

```
LPA -> E : [1] ES10b.PrepareDownload \n (smdpSigned2, smdpSignature2,
CERT.DPpb.SIG, [Hashed Confirmation Code])
rnote over E #FFFFFF
[2]
- Verify CERT.DPpb.SIG
- Verify CERT.DPauth.SIG and CERT.DPpb.SIG have same owner
- Verify smdpSignature2 over smdpSigned2
- Verify smdpSigned2
Endrnote
E --> LPA: [error]
rnote over E #FFFFFF
[3]
- Generate one-time KA key pair
   (otPK.EUICC.KA, otSK.EUICC.KA)
unless a valid otPK.EUICC.KA was provided
- Generate euiccSigned2=
   {TransactionID, otPK.EUICC.KA, [Hashed Confirmation Code]}
- Compute euiccSignature2 over euiccSigned2 and smdpSignature2
Endrnote
E -> LPA: [4] euiccSigned2, euiccSignature2
LPA -> DP : [5] ES9+.GetBoundProfilePackage \n (euiccSigned2, euiccSignature2)

rnote over DP #FFFFFF
[6]
- Verify euiccSignature2 over euiccSigned2
- Determine if Confirmation Code required
Endrnote

Group Cond. Confirmation Code handling
rnote over DP #FFFFFF
[7] [Verify Hashed Confirmation Code]
Endrnote
Group Cond. On CC retry exceeded
DP -> OP : ES2+.HandleNotification
OP --> DP : OK
end
end
DP --> LPA : [8] [error]

rnote over DP #FFFFFF
[9]
- [Generate one-time KA key pair (otPK.DP.KA, otSK.DP.KA)]
- [Generate Session Keys]
- Generate Bound Profile Package
Endrnote

Group Opt.
DP -> OP : [10] ES2+.HandleNotification(...)
OP --> DP : OK
end

DP --> LPA : [11] TransactionID, Bound Profile Package

rnote over LPA #FFFFFF
[12]
- Verify Metadata
- [Prompt/Display Profile Metadata to End User]
- [End User consent]
Endnote

alt Verification failed or no End User consent
     rnote over OP, E #FFFFFF : [13] [Refer to Common Cancel Session procedure
section 3.0.2]
else Otherwise
     rnote over OP, E #FFFFFF : [14] [Refer to Sub-procedure Profile Installation]
end
@enduml
```

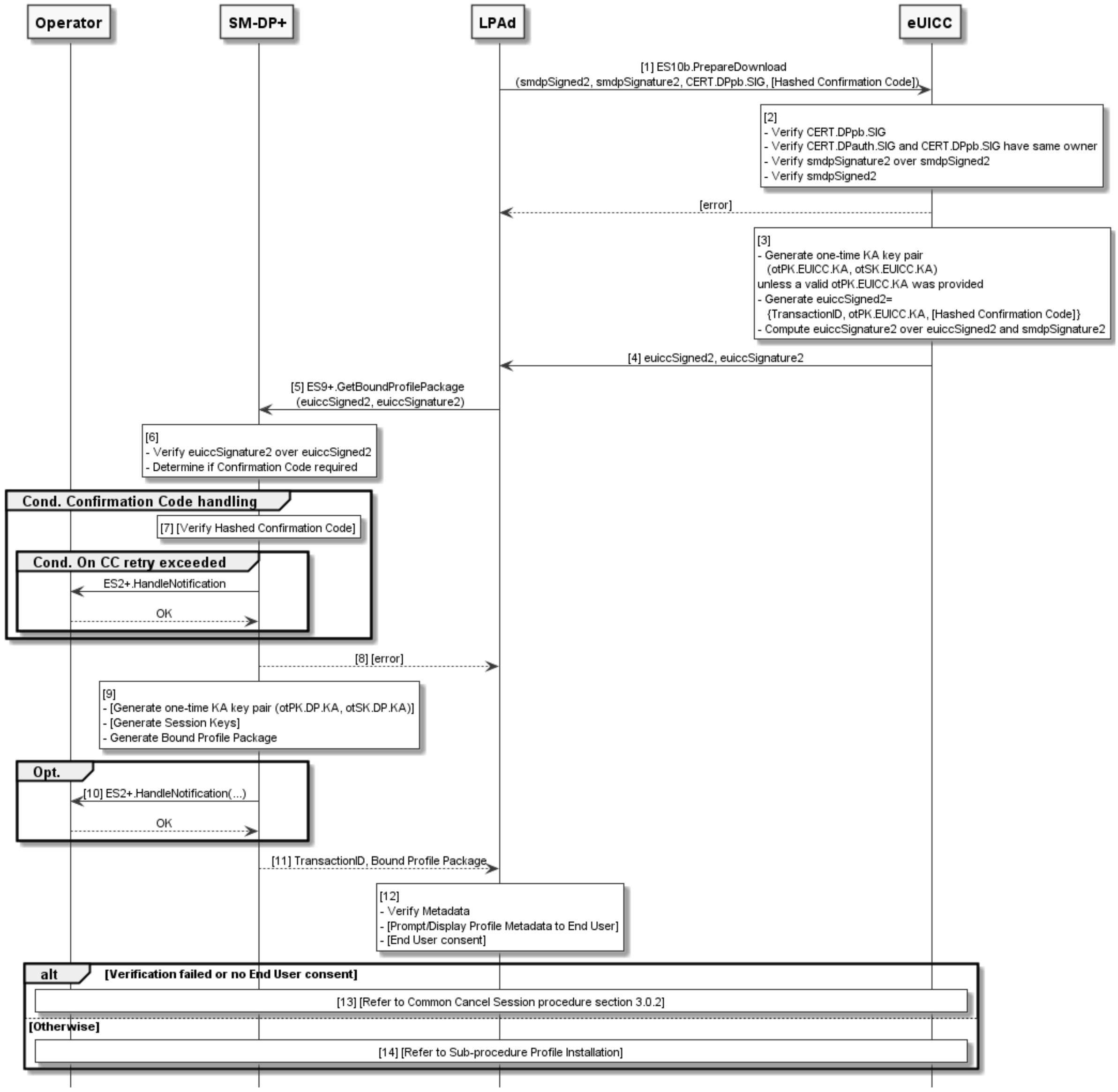

Operator
SM-DP+
LPAd
eUICC
[1] ES10b.PrepareDownload
(smdpSigned2, smdpSignature2, CERT.DPpb.SIG, [Hashed Confirmation Code])
[2]
- Verify CERT.DPpb.SIG
- Verify CERT.DPauth.SIG and CERT.DPpb.SIG have same owner
- Verify smdpSignature2 over smdpSigned2
- Verify smdpSigned2
[error]
[3]
- Generate one-time KA key pair
(otPK.EUICC.KA, otSK.EUICC.KA)
unless a valid otPK.EUICC.KA was provided
- Generate euiccSigned2=
{TransactionID, otPK.EUICC.KA, [Hashed Confirmation Code]}
- Compute euiccSignature2 over euiccSigned2 and smdpSignature2
[4] euiccSigned2, euiccSignature2
[5] ES9+.GetBoundProfilePackage
(euiccSigned2, euiccSignature2)
[6]
- Verify euiccSignature2 over euiccSigned2
- Determine if Confirmation Code required
Cond. Confirmation Code handling
[7] [Verify Hashed Confirmation Code]
Cond. On CC retry exceeded
ES2+.HandleNotification
OK
[8] [error]
[9]
- [Generate one-time KA key pair (otPK.DP.KA, otSK.DP.KA)]
- [Generate Session Keys]
- Generate Bound Profile Package
Opt.
[10] ES2+.HandleNotification(...)
OK
[11] TransactionID, Bound Profile Package
[12]
- Verify Metadata
- [Prompt/Display Profile Metadata to End User]
- [End User consent]
alt [Verification failed or no End User consent]
[13] [Refer to Common Cancel Session procedure section 3.0.2]
[Otherwise]
[14] [Refer to Sub-procedure Profile Installation]

Figure 13: Sub-procedure Profile Download and Installation – Download Confirmation

Start Conditions:

The End User has agreed to the download of the Profile (e.g., by selecting 'Yes').

Procedure:

1. The LPAd SHALL call the "ES10b.PrepareDownload" function optionally including the Hashed Confirmation Code.
2. The eUICC SHALL:

   - Verify that CERT.DPpb.SIG is valid.
   - Verify that CERT.DPauth.SIG and CERT.DPpb.SIG belong to the same entity and are certified by the same certificate.
   - Verify `smdpSignature2`.
   - Verify that the `TransactionID` contained in `smdpSigned2` matches the `TransactionID` of the on-going RSP session.

If any of the verifications fail, the eUICC SHALL return a relevant error status and the procedure SHALL stop.

3. The eUICC SHALL:
   - Use the one-time key pair associated with the `bppEuiccOtpk` if it is provided by the SM-DP+ and it is still stored in the eUICC, or generate a new one-time key pair (see section 5.7.5).
   - Generate the `euiccSigned2` data structure.
   - Compute the `euiccSignature2`.
4. The eUICC SHALL return the "ES10b.PrepareDownload" response.
5. The LPAd calls the "ES9+.GetBoundProfilePackage" function.
6. The SM-DP+ SHALL verify the `euiccSignature2`.

   If a Confirmation Code is required the SM-DP+ SHALL continue with step (7). Otherwise, the SM-DP+ SHALL continue with step (9).
7. The SM-DP+ SHALL:
   - Retrieve the hashed Confirmation Code stored for this order by "ES2.+ConfirmOrder" and calculate the expected hash value as:

     expected hash value =
     SHA256(stored hashed Confirmation Code | TransactionID)
   - Verify that the received Hashed Confirmation Code matches the expected hash value.
   - In case the Confirmation Code verification has failed, the SM-DP+ SHALL increment the count of incorrect Confirmation Code attempts for the Profile. If the maximum number of incorrect attempts has been exceeded, the SM-DP+ SHALL:
     - set the Profile corresponding to the Profile download order in 'Error' state (section 3.1.6).
     - notify the Operator by calling "ES2+.HandleNotification" with the `notificationEvent` indicating 'Confirmation Failure'.
8. If any verification in step 6 or 7 failed, the SM-DP+ SHALL return an error status to the LPAd and the procedure SHALL stop. Unless the maximum number of incorrect attempts for Confirmation Code entries has been exceeded, the LPAd MAY retry by restarting the Profile download and installation procedure.
9. The SM-DP+ SHALL perform the following:
   - Dependent on whether a re-usable BPP is present and whether the eUICC can accept it, the SM-DP+ SHALL do one of the following:
     - Reuse the BPP
     - Rebind the BPP
     - Return an error
     - Create a new BPP
10. (Optional step) Depending on the agreed behaviour with the Operator (out of scope of this specification), the SM-DP+ SHALL notify the Operator that the Profile is

downloaded using the function "ES2+.HandleNotification" with `notificationEvent` indicating 'BPP download'.

NOTE: This notification step MAY be done asynchronously.

11. The SM-DP+ returns "ES9+.GetBoundProfilePackage" response and set the Profile corresponding to the Profile download order in 'Downloaded' state (section 3.1.6).
12. The LPAd SHALL perform additional processing using the Profile Metadata contained within the Bound Profile Package:

    - If the LPAd previously used the Profile Metadata returned by "ES9+.AuthenticateClient" (i.e., in section 3.1.3 step (7) and (8)) then

        - The LPAd SHALL verify that the Profile Policy Rules and the Enterprise Configuration have not changed. If this verification fails, the LPAd SHALL execute the Common Cancel Session procedure with reason `pprNotAllowed` or `metadataMismatch`.
        - The LPAd SHOULD verify that all other Profile Metadata elements it used in that earlier step (such as the Profile Name, Icon, etc.) have not changed. If the verification fails, the LPAd MAY inform the End User and offer the End User to postpone or reject the Profile installation. Alternatively, the LPAd MAY stop the download by executing the Common Cancel Session procedure with reason `metadataMismatch`.

    - If the LPAd has not previously captured the End User consent(s) related to the Profile download as defined in section 3.1.3 step (8), it SHALL do so at this point as described in that step.

13. If any of the cancel conditions given above is true, the Common Cancel Session procedure SHALL be executed. If a notification is sent to the Operator within this procedure, the `notificationEvent` SHALL be set to 'BPP download'.
14. Otherwise, sub-procedure Profile installation described in section hereafter SHALL be executed.

3.1.3.3 Sub-procedure Profile Installation

```
@startuml
hide footbox
skinparam sequenceMessageAlign center
skinparam sequenceArrowFontSize 11
skinparam noteFontSize 11
skinparam monochrome true
skinparam lifelinestrategy solid

participant "<b>Operator" as OP
participant "<b>SM-DP+" as DP
participant "<b>LPAd" as LPA
participant "<b>eUICC" as E

LPA -> E : [1] ES10b.LoadBoundProfilePackage x N\n(ES8+.InitialiseSecureChannel)

rnote over E #FFFFFF
[2]
- Verify InitialiseSecureChannel data
- Generate Session Keys
endnote
```

```
E --> LPA : Response APDU x N
LPA -> E : [3] ES10b.LoadBoundProfilePackage x N\n(ES8+.ConfigureISDP)
E --> LPA : Response APDU x N
LPA -> E : [4] ES10b.LoadBoundProfilePackage x N\n(ES8+.StoreMetadata)
E --> LPA : Response APDU x N

rnote over E #FFFFFF
[4a] Verify PPR(s) against RAT
[4b] [Verify Enterprise Configuration]
endrnote

LPA -> E : [5] [ES10b.LoadBoundProfilePackage x N]\n(ES8+.ReplaceSessionKeys)
E --> LPA : [Response APDU x N]
LPA -> E : [6] ES10b.LoadBoundProfilePackage x N\n(ES8+.LoadProfileElements)
E --> LPA : Response APDU x N \n(ProfileInstallationResult)

LPA -> DP : [7] ES9+.HandleNotification(ProfileInstallationResult)
DP --> LPA : OK

rnote over DP #FFFFFF
[8] [Terminate Download order]
endrnote

DP -> OP : [9] [ES2+.HandleNotification]
OP --> DP : OK

rnote over DP #FFFFFF
[10] [Delete Event, Refer to Event Deletion section 3.6.3]
endrnote

LPA -> E : [11] ES10b.RemoveNotificationFromList

rnote over E #FFFFFF
[12] Delete Notification
endrnote

E --> LPA : OK

rnote over DP, E #FFFFFF
[13] [Next RSP session follows]
endrnote

@enduml
```

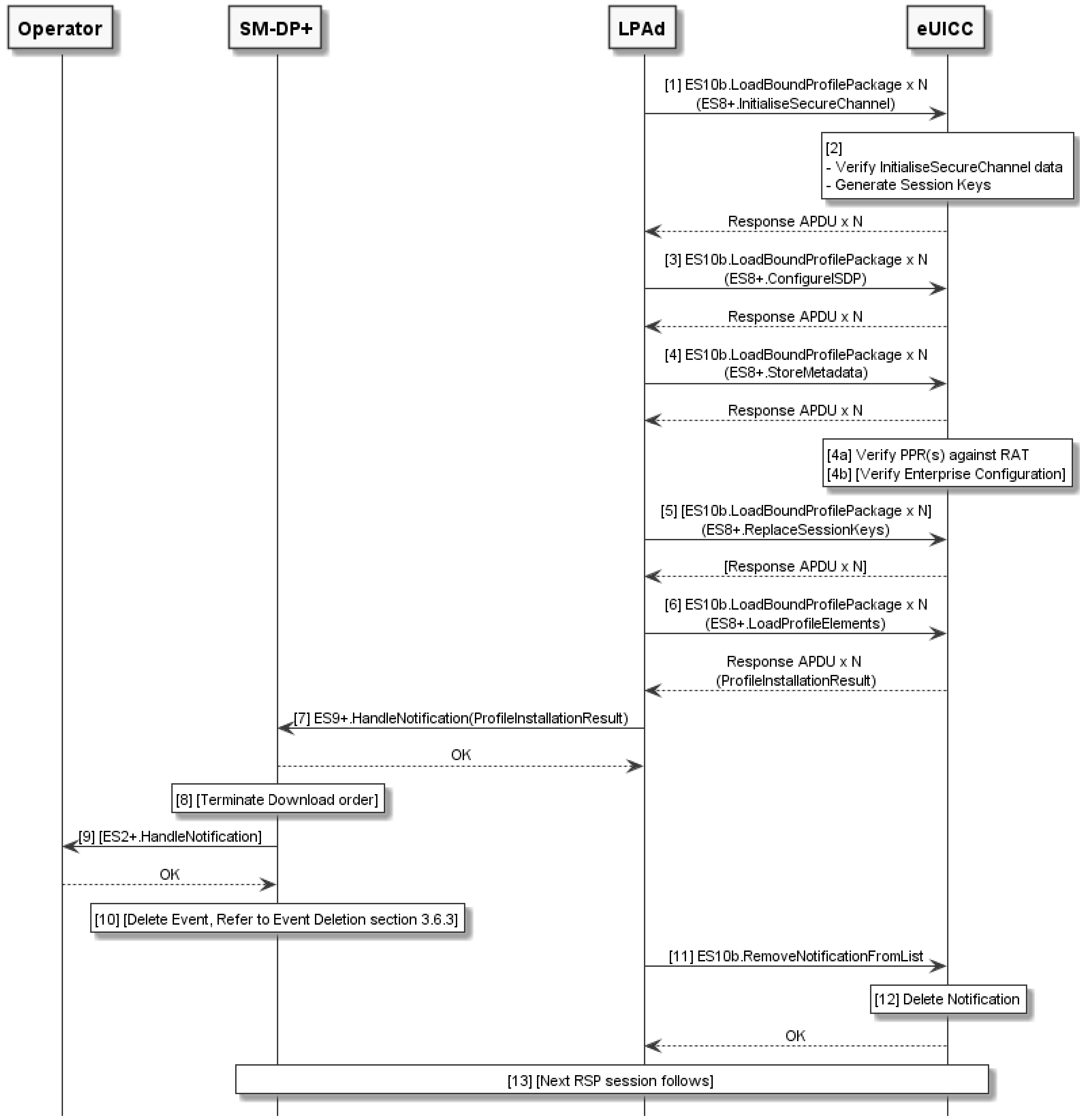

Operator
SM-DP+
LPAd
eUICC
[1] ES10b.LoadBoundProfilePackage x N (ES8+.InitialiseSecureChannel)
[2] - Verify InitialiseSecureChannel data - Generate Session Keys
Response APDU x N
[3] ES10b.LoadBoundProfilePackage x N (ES8+.ConfigureISDP)
Response APDU x N
[4] ES10b.LoadBoundProfilePackage x N (ES8+.StoreMetadata)
Response APDU x N
[4a] Verify PPR(s) against RAT
[4b] [Verify Enterprise Configuration]
[5] [ES10b.LoadBoundProfilePackage x N] (ES8+.ReplaceSessionKeys)
[Response APDU x N]
[6] ES10b.LoadBoundProfilePackage x N (ES8+.LoadProfileElements)
Response APDU x N (ProfileInstallationResult)
[7] ES9+.HandleNotification(ProfileInstallationResult)
OK
[8] [Terminate Download order]
[9] [ES2+.HandleNotification]
OK
[10] [Delete Event, Refer to Event Deletion section 3.6.3]
[11] ES10b.RemoveNotificationFromList
[12] Delete Notification
OK
[13] [Next RSP session follows]

Figure 14: Sub-procedure Profile Installation

Start Conditions:

The BPP has been downloaded to the LPA and all verifications by the LPA were successful.

Procedure:

In this sub-procedure the LPAd generates the Segmented Bound Profile Package according to the description in section 2.5.5 and transfers it to the eUICC using a sequence of "ES10b.LoadBoundProfilePackage" commands. If the LPAd is unable to perform the segmentation (e.g., because of an error in the BPP structure), or if any call of "ES10b.LoadBoundProfilePackage" returns status words other than '90 00' or '91 XX', the LPAd SHALL perform the Common Cancel Session procedure with reason `loadBppExecutionError`. If a subsequent notification is sent to the Operator, the `notificationEvent` SHALL be set to 'BPP installation'.

1. The LPAd SHALL transfer the part of Bound Profile Package containing the "ES8+.InitialiseSecureChannel" function to the eUICC by repeatedly calling the "ES10b.LoadBoundProfilePackage" function.
2. The eUICC SHALL verify the received "ES8+.InitialiseSecureChannel". If the verification succeeds, the eUICC SHALL generate the Session Keys using the input data received in previous step and the key agreement algorithm determined according to section 2.6.5.
3. The LPAd SHALL transfer the part of Bound Profile Package containing the "ES8+.ConfigureISDP" function to the eUICC by repeatedly calling the "ES10b.LoadBoundProfilePackage" function.
4. The LPAd SHALL transfer the part of Bound Profile Package containing the "ES8+.StoreMetadata" function to the eUICC by repeatedly calling the "ES10b.LoadBoundProfilePackage" function.

4a. If the Profile Metadata contains PPR(s), the eUICC SHALL verify each PPR according to PPR verification section 2.9.3.1.

4b. If "ES8+.StoreMetadata" contains an Enterprise Configuration, the eUICC SHALL verify that this configuration is acceptable. If the Reference Enterprise Rule only allows Enterprise Profiles to be installed, the eUICC SHALL verify that the Profile to be installed is an Enterprise Profile.

5. If the Profile Protection Keys (PPK) were included in the Bound Profile Package, the LPAd SHALL transfer the part of Bound Profile Package containing the "ES8+.ReplaceSessionKeys" function to the eUICC by repeatedly calling the "ES10b.LoadBoundProfilePackage" function. The eUICC SHALL decrypt the Profile Protection Keys and replace the current BSP Session Keys with the decrypted Profile Protection Keys.
6. The LPAd SHALL transfer the Profile Elements included in the "ES8+.LoadProfileElements" functions by repeatedly calling the "ES10b.LoadBoundProfilePackage" function.

   If all the Profile Elements are successfully processed and installed, with or without any warning, the last response of the "ES10b.LoadBoundProfilePackage" function SHALL deliver the signed Profile Installation Result as defined in section 2.5.6.
   Installation notifications as configured in the Profile Metadata, if any, SHALL be generated.

   Otherwise, if an error occurs during the transfer and processing of the Profile Elements, or any previous ES8+ function, which does not result in error status words, the eUICC SHALL stop the procedure and SHALL report to the LPAd with a response of the "ES10b.LoadBoundProfilePackage" function including the Profile Installation Result.

   The eUICC SHALL store the Profile Installation Result in its non-volatile memory before delivering it to the LPAd.
   The eUICC SHALL erase the otSK.EUICC.KA attached to this RSP session no later than the successful completion of the BPP installation.
   The LPAd MAY inform the End User of the success or error status indicated by the Profile Installation Result.

7. The LPAd calls the "ES9+.HandleNotification" function in order to deliver the Profile Installation Result to the SM-DP+. The SM-DP+ acknowledges the reception of the Notification to the LPAd.
8. The SM-DP+ SHALL:

   - Retrieve the pending download order identified by the TransactionID. If TransactionID is unknown, the SM-DP+ SHALL terminate its processing.
   - (Conditional) Terminate the pending download order and set the corresponding Profile in state 'Installed' or 'Error' (section 3.1.6) as indicated by the Profile Installation Result.

9. (Conditional) The SM-DP+ SHALL call the "ES2+.HandleNotification" with:

   - `notificationEvent` indicating 'BPP installation';
   - `notificationEventStatus` reflecting the value received in ES9+.HandleNotification;
   - `notificationReceiverIdentifier` reflecting the functionRequesterIdentifier value of the associated ES2+.ConfirmOrder;
   - `notificationIdentifier` reflecting the functionCallIdentifier value of the associated ES2+.ConfirmOrder;

10. (Conditional) If this procedure is executed in the context of option (b), the SM-DP+ SHALL execute the SM-DS event deletion procedure (section 3.6.3).
11. On reception of the acknowledgement message from the SM-DP+ the LPAd SHALL call "ES10b.RemoveNotificationFromList" with the corresponding seqNumber.
12. The eUICC SHALL delete the Profile Installation Result from its non-volatile memory.
13. (Conditional) If the LPAd has received `rpmPending` in the response of "ES9+.AuthenticateClient" function call, the LPAd SHOULD initiate an additional RSP Session with the SM-DP+, setting the `operationType` to indicate `rpm`. If this RSP session was triggered by an Event Record from an SM-DS, the pending RSP session with the SM-DP+ SHOULD be executed before continuing processing any remaining Event Records from that SM-DS.

3.1.4 Limitation for Profile Installation

Several profiles MAY be installed on the eUICC, subject to non-volatile memory limitations.

3.1.5 Error Handling Within the Profile Download and Installation Procedure

During Profile Installation (section 3.1.3.3), when the next segment of an SBPP is to be sent to the eUICC, the eUICC SHALL handle TLVs that are different from what is defined for the SBPP in section 2.5.5 as follows:

- The eUICC SHALL process the commands "ES10b.GetEUICCChallenge" (indicating the start of a new download session) and "ES10b.CancelSession" (indicating the termination of the current download session) as specified.
- The eUICC MAY also process other ES10 commands.

- The eUICC SHALL reject other ES10 commands it does not process and any other TLV with status words '69 85' (Conditions of use not satisfied) or '6A 88' (Reference data not found).

In the latter 2 cases, the eUICC SHALL NOT discard the download session state.

3.1.6 Profile Lifecycle at SM-DP+

The previous sections provide detailed procedures associated with Remote Provisioning. Each Profile has state information on the SM-DP+ associated with it during the provisioning into an eUICC. The Profile lifecycle state can be one of the states listed in the following table.

Additional states and additional or customised ES2+ functions MAY be agreed between the Operator and the SM-DP+.

State Name	Description
Available	The Profile is available in the inventory of the SM-DP+.
Allocated	The Profile is reserved for downloading without being linked to an EID.
Linked	The Profile is reserved for downloading and is linked to an EID.
Confirmed	The Profile is reserved for downloading (linked or not linked to an EID) with Matching ID and Confirmation Code if required.
Released	The Profile is ready for download and installation after Network Configuration by the Operator (e.g.: HLR Registration).
Downloaded	The Bound Profile was delivered to the LPA.
Installed	The Profile was successfully installed on the eUICC.
Error	The Profile has not been installed because of one of the following error cases: - Confirmation Code Retry Limit exceeded - Download Retry Limit exceeded - End User Rejection - Permanent error during download and installation
Unavailable	The Profile cannot be reused anymore by the SM-DP+.

Table 6b: Profile State in the SM-DP+

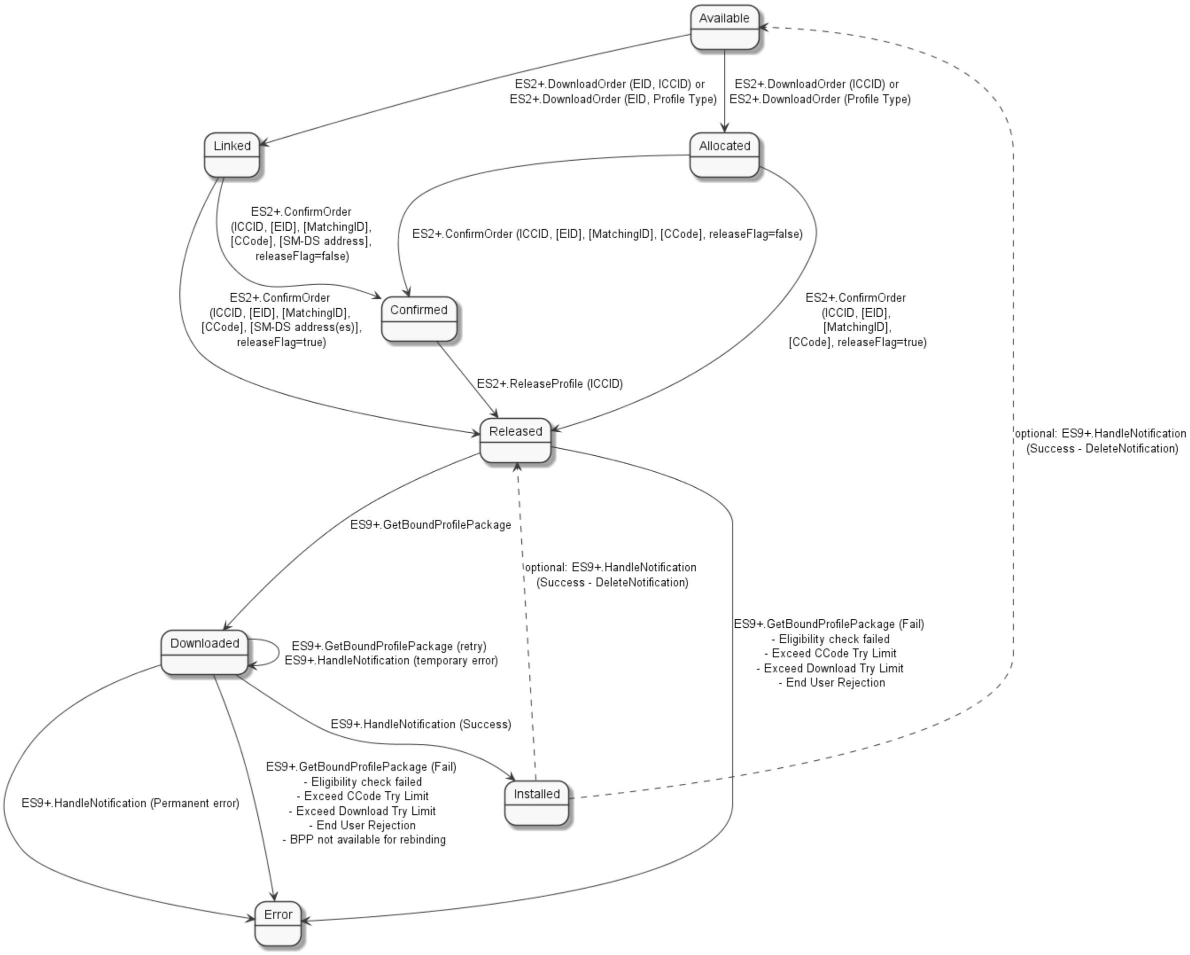
The following additional Profile states MAY be tracked by an SM-DP+ that receives Notifications from an eUICC.

State Name	Description
Enabled	The last Notification received from the eUICC was an enable Notification.
Disabled	The last Notification received from the eUICC was a disable Notification.
Deleted	The last Notification received from the eUICC was a delete Notification.

Table 6c: Notified Profile States in the SM-DP+

The following two state transition diagrams show the Profile lifecycle state on the SM-DP+ and provide the details of the actions previously performed on a Profile together with the possible next action.

```
@startuml
skinparam monochrome true

State Available {
}

state Allocated {
}

state Linked {
}

state Confirmed {
}

state Released {
}

state Downloaded {
}

state Installed {
}

state Error{
}

Available --> Allocated : ES2+.DownloadOrder (ICCID) or \n ES2+.DownloadOrder
(Profile Type)
Available --> Linked : ES2+.DownloadOrder (EID, ICCID) or \n ES2+.DownloadOrder
(EID, Profile Type)

Allocated --> Confirmed : ES2+.ConfirmOrder (ICCID, [EID], [MatchingID], [CCode],
releaseFlag=false)
Allocated --> Released : ES2+.ConfirmOrder \n(ICCID, [EID], \n [MatchingID],
\n[CCode], releaseFlag=true)
Linked --> Confirmed : ES2+.ConfirmOrder \n(ICCID, [EID], [MatchingID], \n[CCode],
[SM-DS address], \nreleaseFlag=false)
Linked --> Released : ES2+.ConfirmOrder \n(ICCID, [EID], [MatchingID], \n [CCode],
[SM-DS address(es)], \nreleaseFlag=true)


Confirmed --> Released : ES2+.ReleaseProfile (ICCID)
Released -down--> Downloaded : ES9+.GetBoundProfilePackage

Downloaded --> Installed : ES9+.HandleNotification (Success)

Installed -[dashed]-> Released : optional: ES9+.HandleNotification \n(Success -
DeleteNotification)
Installed -[dashed]-> Available : optional: ES9+.HandleNotification \n(Success -
DeleteNotification)

Downloaded --> Downloaded : ES9+.GetBoundProfilePackage (retry)
\nES9+.HandleNotification (temporary error)
Downloaded -down--> Error : ES9+.HandleNotification (Permanent error)
Downloaded --> Error : ES9+.GetBoundProfilePackage (Fail) \n - Eligibility check
failed \n - Exceed CCode Try Limit \n - Exceed Download Try Limit \n - End User
Rejection \n - BPP not available for rebinding

Released --> Error : ES9+.GetBoundProfilePackage (Fail) \n - Eligibility check
failed \n - Exceed CCode Try Limit \n - Exceed Download Try Limit \n - End User
Rejection

@enduml
```


Available
ES2+.DownloadOrder (EID, ICCID) or
ES2+.DownloadOrder (EID, Profile Type)
ES2+.DownloadOrder (ICCID) or
ES2+.DownloadOrder (Profile Type)
Linked
Allocated
ES2+.ConfirmOrder
(ICCID, [EID], [MatchingID],
[CCode], [SM-DS address],
releaseFlag=false)
ES2+.ConfirmOrder (ICCID, [EID], [MatchingID], [CCode], releaseFlag=false)
ES2+.ConfirmOrder
(ICCID, [EID], [MatchingID],
[CCode], [SM-DS address(es)],
releaseFlag=true)
Confirmed
ES2+.ConfirmOrder
(ICCID, [EID],
[MatchingID],
[CCode], releaseFlag=true)
ES2+.ReleaseProfile (ICCID)
Released
optional: ES9+.HandleNotification
(Success - DeleteNotification)
ES9+.GetBoundProfilePackage
optional: ES9+.HandleNotification
(Success - DeleteNotification)
ES9+.GetBoundProfilePackage (Fail)
- Eligibility check failed
- Exceed CCode Try Limit
- Exceed Download Try Limit
- End User Rejection
Downloaded
ES9+.GetBoundProfilePackage (retry)
ES9+.HandleNotification (temporary error)
ES9+.HandleNotification (Success)
ES9+.GetBoundProfilePackage (Fail)
- Eligibility check failed
- Exceed CCode Try Limit
- Exceed Download Try Limit
- End User Rejection
- BPP not available for rebinding
Installed
ES9+.HandleNotification (Permanent error)
Error

NOTE: "ES2+.HandleNotification" does not have any impact on the Profile state.

```
@startuml
skinparam monochrome true

state Available {
}

state States {
      state Allocated {
      }
      state Linked {
      }
      state Confirmed {
      }
      state Released {
      }
}

state Error{
}

State Unavailable {
}

States -left--> Available : ES2+.CancelOrder (ICCID, EID, MatchingID) \n with
finalProfileStatusIndicator = Available
States -right--> Unavailable : ES2+.CancelOrder (ICCID, EID, MatchingID) \n with
finalProfileStatusIndicator = Unavailable
Error -up--> Available : ES2+.CancelOrder (ICCID, EID, MatchingID) \n with
finalProfileStatusIndicator = Available \n or Automatic Transition
```

```
Error -up--> Unavailable: ES2+.CancelOrder (ICCID, EID, MatchingID) \n with
finalProfileStatusIndicator = Unavailable \n or Automatic Transition
@enduml
```

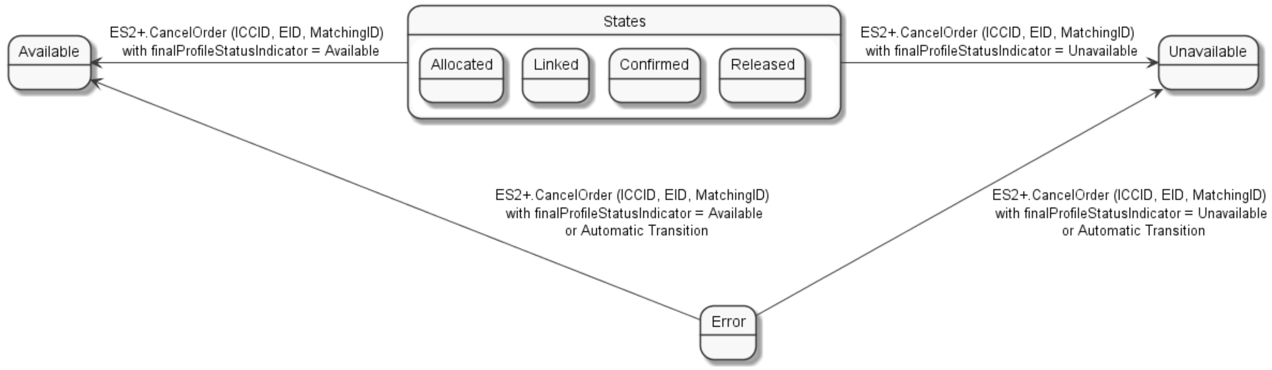

Available
ES2+.CancelOrder (ICCID, EID, MatchingID)
with finalProfileStatusIndicator = Available
States
Allocated
Linked
Confirmed
Released
ES2+.CancelOrder (ICCID, EID, MatchingID)
with finalProfileStatusIndicator = Unavailable
Unavailable
ES2+.CancelOrder (ICCID, EID, MatchingID)
with finalProfileStatusIndicator = Available
or Automatic Transition
ES2+.CancelOrder (ICCID, EID, MatchingID)
with finalProfileStatusIndicator = Unavailable
or Automatic Transition
Error

Figure 15: Profile Instance Lifecycle State Transit Diagram at SM-DP+

3.2 Local Profile Management

The End User initiates Local Profile Management procedures using the LUI. As specified in SGP.21 [4], User Intent is required for all procedures directed to Operational Profiles, except List Profiles. For each procedure the LPA SHALL enforce the Confirmation Level that is equal to or higher than the Confirmation Level defined in this specification. The specific implementation of Confirmation Requests by the LPA is out of scope of this specification.

In all cases, if the End User refuses or does not respond to a Confirmation Request, then the associated operation SHALL stop.

Confirmation Requests MAY be combined for consecutive operations to simplify the user experience and avoid repeated input steps for the End User. For instance, when performing a Profile download with an Activation Code, the Strong Confirmation for download and Simple Confirmation for enabling the Profile MAY be combined. In the case of combined Confirmation Requests, it SHALL be clear to the End User what operations will be performed, and the highest level of confirmation SHALL be obtained.

The LUI implementation of Add Profile (section 3.2.5) and Update all Profiles (in section 3.2.7) MAY be combined. For instance, as a composite Add/Update All operation, the LPAd MAY initiate Profile downloads and RPM downloads from the Default SM-DP+(s), Root SM-DS(s), and all Polling Addresses of the installed Profiles.

Some Local Profile Management procedures provide two options: Using a REFRESH proactive command to notify the Device about a change and performing such a change without such a command. While both options are available for LPAd, the procedures SHALL always use the option with the REFRESH proactive command for the LPAe.

3.2.1 Enable Profile

This procedure is used to enable a Profile already downloaded and installed on an eUICC.

The procedure applies for SEP and MEP. The following applies for Command Port and Target Port:

	Command Port	Target Port
MEP-A1/A2	0	>0
MEP-B	>=0	Identical to Command Port
SEP	Physical interface	Physical interface

For MEP-A2, the eUICC SHALL select an eSIM Port which currently has no Profile enabled, i.e., implicit disabling will never occur.

For MEP-B, the LPA SHALL set the refreshFlag.

For MEP-A2, the LPA SHALL NOT set the refreshFlag.

```
@startuml
skinparam monochrome true
skinparam ArrowColor Black

hide footbox

participant "End-user" as EndUser #FFFFFF
participant "LUId" as LPA #FFFFFF
participant "eUICC \n LPA Services (ISD-R)" as LPAServices #FFFFFF
participant "Device baseband" as DevBB #FFFFFF

rnote over EndUser, LPA #FFFFFF
     [0] End-user interactions
end rnote

Opt refreshFlag not set
rnote over LPA, DevBB #FFFFFF
[1] TP: The Device
  a)  Runs the application session termination procedure
  b)  Closes logical channels
  c)  Terminate an ongoing proactive command session
end rnote
end opt

LPA -> LPAServices: [2] CP: ES10c.Enable(ISD-P AID or ICCID, [TP], refreshFlag)
rnote over LPAServices #FFFFFF
     [3] Verify Profile state
     [4a] Enforce Profile Policy Rules
     [4b] [Verify Reference Enterprise Rule]
     [5] [Perform Test Profile checks]
end rnote
LPAServices --> LPA: [Error]

Alt REFRESH required
rnote over LPAServices #FFFFFF
     [6a] TP: Determine if proactive session is ongoing
end rnote
LPAServices --> LPA: [Error: catBusy]
LPAServices -> LPA: [6b] Ok
LPAServices -> DevBB: [7] TP: REFRESH
DevBB -> LPAServices: Terminal Response or RESET
rnote over LPAServices #FFFFFF
[8] TP: The Target Profile is Enabled
end rnote

Else REFRESH not required
rnote over LPAServices #FFFFFF
[9] TP: The Target Profile is Enabled
end rnote
LPAServices -> LPA: [10] Ok
```

```
LPA -> DevBB: [11] New Enabled Profile

end
rnote over DevBB #FFFFFF
[12]  TP: Baseband executing a
network attach procedure with
the newly Enabled Profile
end rnote

rnote over LPA
[13] [trigger PCM session]
end rnote

@enduml
```

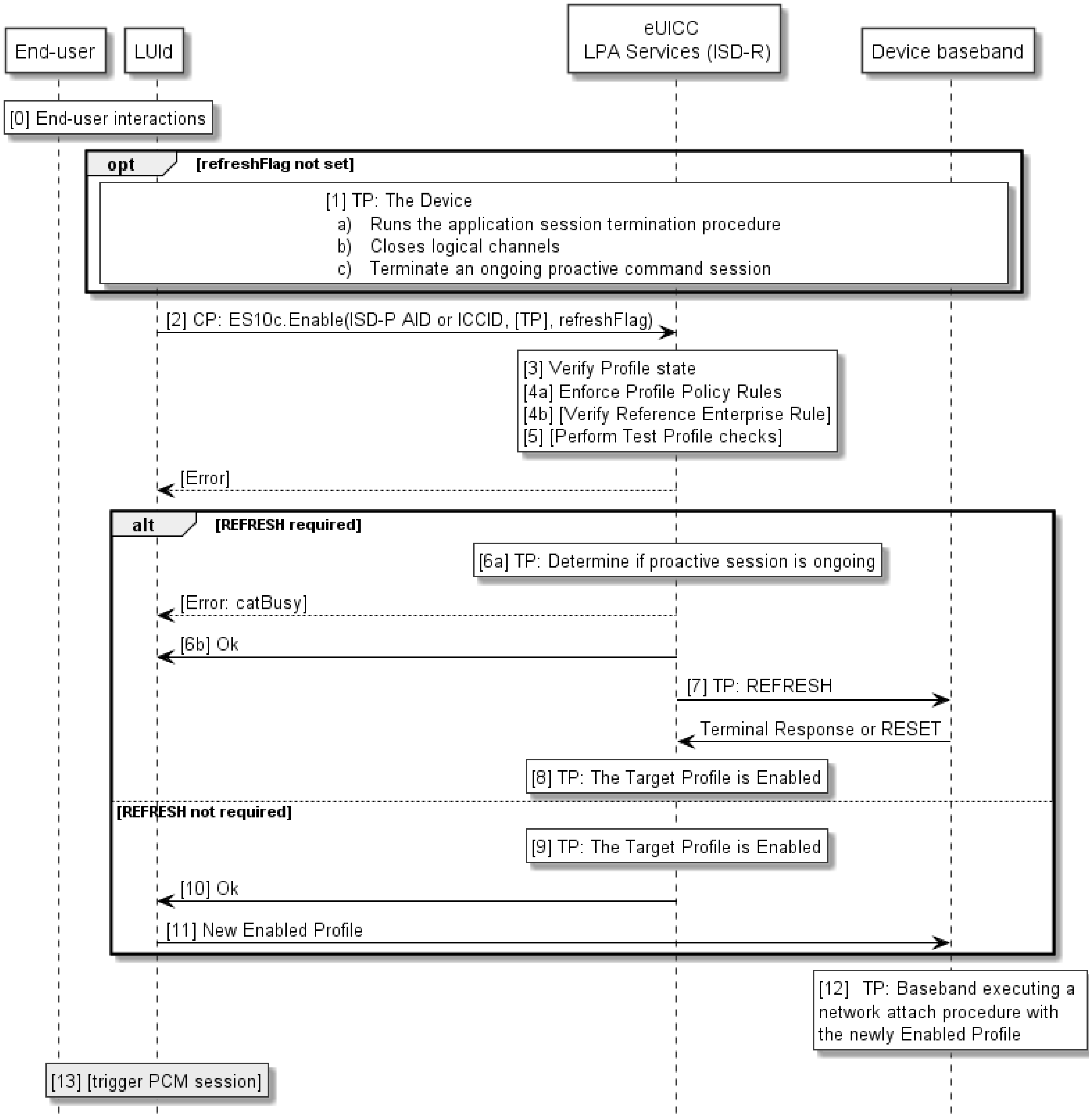

End-user
LUId
eUICC
LPA Services (ISD-R)
Device baseband
[0] End-user interactions
opt
[refreshFlag not set]
[1] TP: The Device
a) Runs the application session termination procedure
b) Closes logical channels
c) Terminate an ongoing proactive command session
[2] CP: ES10c.Enable(ISD-P AID or ICCID, [TP], refreshFlag)
[3] Verify Profile state
[4a] Enforce Profile Policy Rules
[4b] [Verify Reference Enterprise Rule]
[5] [Perform Test Profile checks]
[Error]
alt
[REFRESH required]
[6a] TP: Determine if proactive session is ongoing
[Error: catBusy]
[6b] Ok
[7] TP: REFRESH
Terminal Response or RESET
[8] TP: The Target Profile is Enabled
[REFRESH not required]
[9] TP: The Target Profile is Enabled
[10] Ok
[11] New Enabled Profile
[12] TP: Baseband executing a network attach procedure with the newly Enabled Profile
[13] [trigger PCM session]

Figure 16: Enable Profile

Start Conditions:

When the Profile to be enabled is an Operational Profile:

- User Intent is acquired as defined in SGP.21 [4].

When the Profile to be enabled is a Test Profile:

- The Device is in Device Test Mode.

When the Profile to be enabled is a Provisioning Profile:

- The currently-enabled Operational Profile, if any, is unsuitable to provide the connectivity required for an operation such as Add Profile.

Procedure:

0. The End User is presented with a user interface that displays the list of installed Profiles within the eUICC, with their current states (Enabled or Disabled). The End User selects the Profile to be enabled. The LPAd MAY check the Profile Policy Rules of the Profiles and give appropriate warnings to the End User (e.g., that due to Profile Policy Rules the Profile cannot be enabled). The enabling of a Provisioning Profile can be initiated by the LPAd itself without any End User interaction. The enabling of a Profile combined with its download can omit this step.
1. Before the LPAd calls the EnableProfile function with the refreshFlag not set, the Device has the responsibility to ensure that the relevant conditions for use are met:

   For SEP, MEP-A1 and MEP-B, if a Profile currently is enabled on the Target Port:

   a) The Device SHALL run the application session termination procedure in accordance with ETSI TS 102 221 [6] for every active application of the Profile currently enabled on the Target Port.
   b) The Device SHALL close all logical channels that were used to select these applications.
   c) The Device SHOULD take implementation-dependent action to terminate an ongoing proactive command session.

   For MEP-A2, the Device MAY verify if an eSIM Port is available which has currently no Profile enabled.

   Before the LPAd calls the EnableProfile function with the refreshFlag set, the Device has the responsibility to ensure that CAT is initialised on the Target Port.

2. On the Command Port, the LPAd SHALL call the "ES10c.EnableProfile" (section 5.7.16) function of the ISD-R with its relevant input data, which includes the indication if a REFRESH proactive command is needed. For MEP, the Target Port is determined as follows:
   - For MEP-A1, the Target Port SHALL be indicated in the input data.
   - For MEP-A2, the Target Port SHALL be selected by the eUICC in step 9.
   - For MEP-B, the command SHALL be sent on the Target Port.

3. The ISD-R SHALL verify the state of the Target Profile. If the Target Profile is not in Disabled state, the ISD-R SHALL return a response indicating a failure, and the procedure SHALL stop.

4a. If the Target Profile is not a Test Profile, the ISD-R SHALL check the Profile Policy Rules of the Profile currently enabled on the Target Port (if any). If it has a Profile Policy Rule "Disabling not allowed", the ISD-R SHALL return a response indicating a failure, and the procedure SHALL stop.

4b. If the eUICC contains a Profile with a Reference Enterprise Rule, it SHALL verify the following:

- If the Target Profile is not an Enterprise Profile or a Test Profile: That the maximum number of non-Enterprise Profiles that can be Enabled is not exceeded.
- If the Reference Enterprise Rule indicates "`priorityEnterpriseProfile`" and the Profile with the Reference Enterprise Rule is currently disabled, that itself or a Test Profile is the Target Profile.

If any of these verifications fail, the ISD-R SHALL return a response indicating a failure, and the procedure SHALL stop.

5. If the Profile currently enabled on the Target Port is a Test Profile, the ISD-R SHALL check if the Target Profile is either another Test Profile or the Operational profile that was previously in Enable state. If this is not the case, the ISD-R SHALL return a response indicating a failure, and the procedure SHALL stop.

If the refreshFlag is set, steps 6 to 8 SHALL be executed.

6a. If a proactive session is ongoing on the Target Port, the ISD-R MAY return a `catBusy` error response to the LUId. If this occurs, the LPAd MAY take implementation-dependent actions to terminate the proactive command session, after which the LPAd MAY call again the "ES10c.EnableProfile” function without any further End User interaction.

6b. If the ISD-R does not stop execution due to an ongoing proactive session, then it SHALL return a response indicating result OK back to the LUId.

7. The eUICC SHALL trigger a REFRESH of the Target Port as follows:
   - For SEP and MEP-B, the eUICC SHALL send a REFRESH proactive command on the Target Port.
   - For MEP-A1, the eUICC SHALL send an LSI COMMAND proactive command with the action "Proactive session request" on the Command Port. This results in the Device checking for pending proactive commands on the Target Port, whereupon the eUICC sends a REFRESH proactive command on the Target Port.
8. Upon reception of the Terminal Response or after the RESET of the Target Port, the ISD-R SHALL disable the currently Enabled Profile (if any) and then enable the Target Profile on the Target Port.

If the refreshFlag is not set, steps 9 to 11 SHALL be executed.

9. For MEP-A2, the eUICC SHALL select an eSIM Port which currently has no Profile enabled as Target Port. If no eSIM Port is available, the ISD-R SHALL return a response indicating a failure, and the procedure SHALL stop. The ISD-R SHALL disable the Profile currently enabled on the Target Port (if any). The ISD-R SHALL enable the Target Profile on the Target Port.
10. The ISD-R SHALL return a response indicating result OK back to the LUId. For MEP-A2, the response SHALL include the Target Port.

11. The Device SHALL discard any state, including the PIN state, and any cached file content including EF_{ICCID} and EF_{DIR}, and any proactive command session from the previously enabled Profile, if any, on the Target Port. The LPA signals the baseband connected to the Target Port that a new Profile was enabled. The Device SHALL proceed with the UICC activation procedure including TERMINAL PROFILE, as defined in ETSI TS 102 221 [6] clause 14.5.1.

12. The baseband triggers the execution of a network attach procedure with the newly Enabled Profile on the Target Port.
13. If the Device supports PCM, `lprConfiguration.triggerLprOnEnableProfile` is present in the Profile Metadata of the enabled Profile, and appropriate connectivity is available, then the LPRd SHALL trigger a Profile Content Management session using the optional DPI in the Profile Metadata as described in section 3.9.

If the currently Enabled Profile is not a Provisioning Profile and is not able to provide connectivity, there SHALL NOT be any fallback to the previously Enabled Profile. Further action SHALL remain under the responsibility of the End User.

3.2.2 Disable Profile

This procedure is used to disable a Profile, already downloaded and installed on an eUICC, and Enabled on the Target Port.

The procedure applies for SEP and MEP. The following applies for Command Port and Target Port:

	Command Port	Target Port
MEP-A1/A2	0	>0
MEP-B	>=0	>=0 If CAT has been initialised on the Target Port, then identical to Command Port. Otherwise, MAY be different from Command Port.
SEP	Physical interface	Physical interface

For MEP, the Target Port is the eSIM Port where the Target Profile is currently enabled.

For MEP-B, the LPA SHALL set the refreshFlag.

For MEP-A2, the LPA SHALL NOT set the refreshFlag.

NOTE: There may be situations where a Profile is enabled on an eSIM Port which is no longer accessible by the baseband, e.g., due to switching to a physical SIM. For a Device typically using refresh mode when switching profiles, this procedure allows gracefully handling of such situations: even if the refreshFlag is set, the eUICC will skip the refresh if CAT is not initialised (see step 6), and for MEP-B, the DisableProfile function can be sent on a different eSIM Port.

```
@startuml
skinparam monochrome true
skinparam ArrowColor Black

hide footbox

participant "End-user" as EndUser #FFFFFF
participant "LUId" as LPA #FFFFFF
participant "eUICC \n LPA Services (ISD-R)" as LPAServices #FFFFFF
participant "Device baseband" as DevBB #FFFFFF

rnote over EndUser, LPA #FFFFFF
     [0] End-user interactions
end rnote

Opt refreshFlag not set
rnote over LPA, DevBB #FFFFFF
[1] TP: The Device
  a) Runs the application session termination procedure
  b) Closes logical channels
  c) Terminate an ongoing proactive command session
end rnote
end opt

LPA -> LPAServices: [2] CP: ES10c.Disable(ISD-P AID or ICCID, refreshFlag)
note over LPAServices #FFFFFF
     [3] Verify Profile state
     [4] Enforce Profile Policy Rules
end note

LPAServices --> LPA: [Error]

Alt REFRESH required
rnote over LPAServices #FFFFFF
     [5a] TP: Determine if proactive session is ongoing
end rnote
LPAServices --> LPA: [Error: catBusy]
LPAServices -> LPA: [5b] Ok
LPAServices -> DevBB: [6] TP : REFRESH
DevBB -> LPAServices: Terminal Response or RESET
rnote over LPAServices #FFFFFF
[7] TP: The Target Profile is Disabled
end rnote

Alt If the Target Profile is a Test Profile, an Operational Profile was formerly
Enabled \nand this Operational Profile was not deleted while a Test Profile was
Enabled
rnote over LPAServices #FFFFFF
[8] TP: Enable formerly Enabled Operational Profile
end rnote
rnote over DevBB #FFFFFF
TP: Baseband executing a
network attach procedure with
the newly Enabled Profile
end rnote

end Alt

Else REFRESH not required

rnote over LPAServices #FFFFFF
[9] TP: The Target Profile is Disabled
end rnote

Alt If the Target Profile is a Test Profile, an Operational Profile was formerly
Enabled \nand this Operational Profile was not deleted while a Test Profile was
Enabled
rnote over LPAServices #FFFFFF
```

```
[10] TP: Enable formerly Enabled Operational Profile
end rnote
LPAServices -> LPA: [11] Ok
rnote over LPA #FFFFFF
[12] TP: The Device SHALL discard:
 - any state of the previously Enabled Profile
 - any cached file content
 - any proactive command session
end rnote

LPA -> DevBB: [13] New Enabled Profile
rnote over DevBB #FFFFFF
TP: Baseband executing a
network attach procedure with
the newly Enabled Profile
end rnote

else else
LPAServices -> LPA: [14] Ok
rnote over LPA #FFFFFF
[15] TP: The Device SHALL discard:
 - any state of the previously Enabled Profile
 - any cached file content
 - any proactive command session
end rnote

LPA -> DevBB: [16] No Enabled Profile
end Alt

end

@enduml
```

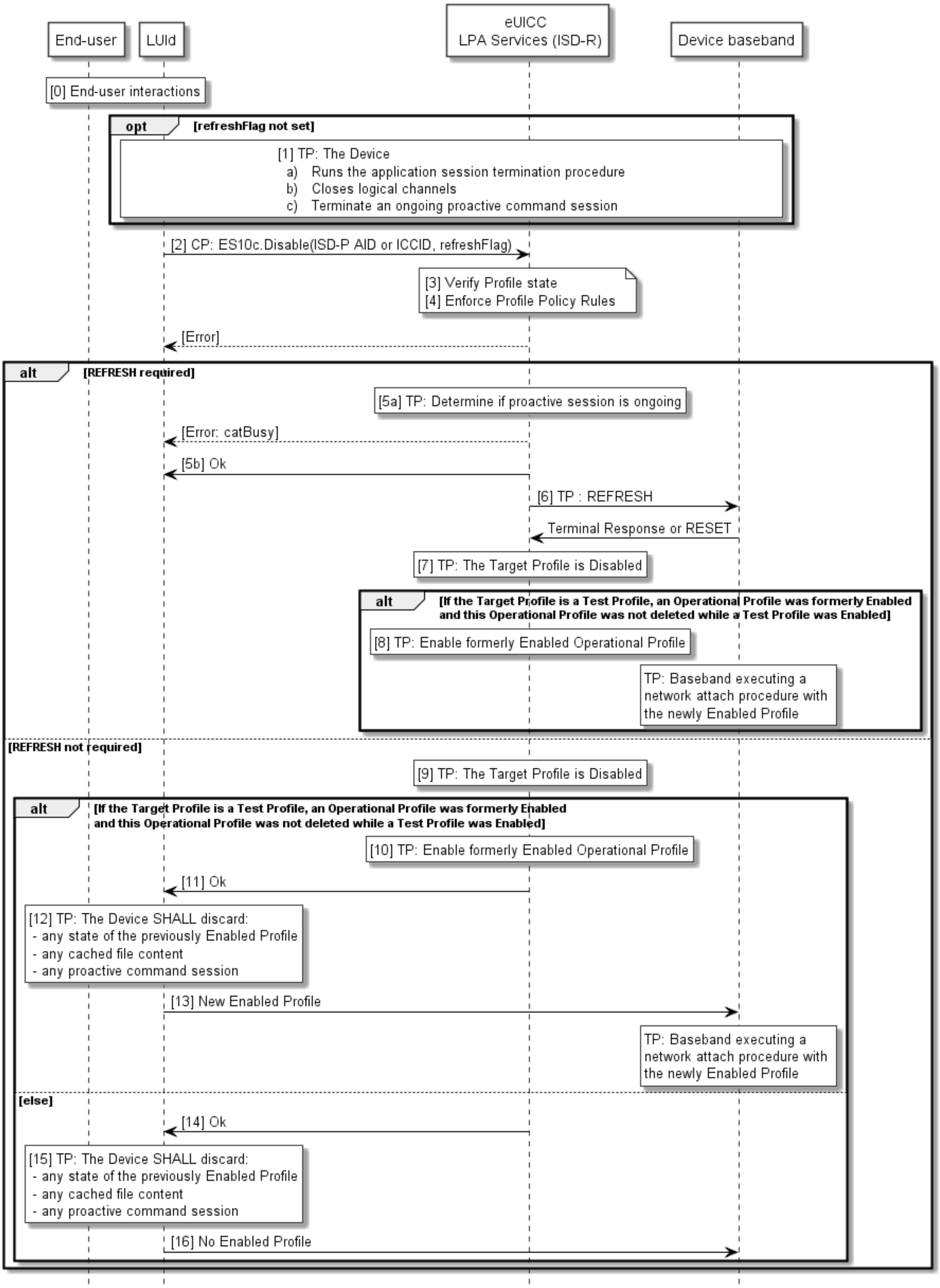

End-user
LUId
eUICC
LPA Services (ISD-R)
Device baseband
[0] End-user interactions
opt
[refreshFlag not set]
[1] TP: The Device
a) Runs the application session termination procedure
b) Closes logical channels
c) Terminate an ongoing proactive command session
[2] CP: ES10c.Disable(ISD-P AID or ICCID, refreshFlag)
[3] Verify Profile state
[4] Enforce Profile Policy Rules
[Error]
alt
[REFRESH required]
[5a] TP: Determine if proactive session is ongoing
[Error: catBusy]
[5b] Ok
[6] TP : REFRESH
Terminal Response or RESET
[7] TP: The Target Profile is Disabled
alt
[If the Target Profile is a Test Profile, an Operational Profile was formerly Enabled and this Operational Profile was not deleted while a Test Profile was Enabled]
[8] TP: Enable formerly Enabled Operational Profile
TP: Baseband executing a network attach procedure with the newly Enabled Profile
[REFRESH not required]
[9] TP: The Target Profile is Disabled
alt
[If the Target Profile is a Test Profile, an Operational Profile was formerly Enabled and this Operational Profile was not deleted while a Test Profile was Enabled]
[10] TP: Enable formerly Enabled Operational Profile
[11] Ok
[12] TP: The Device SHALL discard:
- any state of the previously Enabled Profile
- any cached file content
- any proactive command session
[13] New Enabled Profile
TP: Baseband executing a network attach procedure with the newly Enabled Profile
[else]
[14] Ok
[15] TP: The Device SHALL discard:
- any state of the previously Enabled Profile
- any cached file content
- any proactive command session
[16] No Enabled Profile

Figure 17: Disable Profile

Start Conditions:

When the Profile to be disabled is an Operational Profile:

- User Intent is acquired as defined in SGP.21 [4].

When the Profile to be disabled is a Test Profile:

- The Device is in Device Test Mode.

When the Profile to be disabled is a Provisioning Profile:

- The operation requiring connectivity from the Provisioning Profile, such as a Profile download, has completed.

Procedure:

0. The End User is presented with a user interface that displays the list of installed Profiles within the eUICC, with their current states (Enabled or Disabled), as described in "List Profiles" procedure (section 3.2.4). The End User selects the Profile to be Disabled. The disabling of a Provisioning Profile or a Test Profile can be initiated by the LPAd itself without any End User interaction. The LPAd MAY check the Profile Policy Rules of the Profile and give appropriate warnings to the End User.
1. Before the LPAd calls the DisableProfile function with the refreshFlag not set, the Device has the responsibility to ensure that the relevant conditions for use are met on the Target Port. I.e., the Device:

    a) SHALL run the application session termination procedure in accordance with ETSI TS 102 221 [6] for every active application of the currently enabled Profile on the Target Port.
    b) SHALL close all logical channels that were used to select these applications.
    c) SHOULD take implementation-dependent action to terminate an ongoing proactive command session.

2. On the Command Port, the LPAd SHALL call the "ES10c.DisableProfile" (section 5.7.17) function of the ISD-R with its relevant input data, which includes the indication if a REFRESH proactive command is needed.
3. The ISD-R SHALL verify the state of the Target Profile. If the Target Profile is not in the Enabled state, the ISD-R SHALL return a response indicating a failure, and the procedure SHALL stop.
4. For non MEP-Capable eUICC: The ISD-R SHALL check the Profile Policy Rules of the currently Enabled Profile.
If it has a Profile Policy Rule "Disabling not allowed", the ISD-R SHALL return a response indicating a failure, and the procedure SHALL stop.

If refreshFlag is set, steps 5 to 8 SHALL be executed.

5a. If a proactive session is ongoing on the Target Port, the ISD-R MAY return a `catBusy` error response to the LUId. If this occurs, the LPAd MAY take implementation-dependent actions to terminate the proactive command session, after which the LPAd MAY call again the "ES10c.DisableProfile" function without any further End User interaction.

5b. If the ISD-R does not stop execution due to an ongoing proactive session, then it SHALL return a response indicating result OK back to the LUId.

6. If the Target Profile is enabled on an eSIM Port where CAT has been initialised, the eUICC SHALL trigger a REFRESH of the Target Port as follows:
   - For SEP and MEP-B, the eUICC SHALL send a REFRESH proactive command on the Target Port.
   - For MEP-A1, the eUICC SHALL send an LSI COMMAND proactive command with the action "Proactive session request" on the Command Port. This results in the Device checking for pending proactive commands on the Target Port, whereupon the eUICC sends a REFRESH proactive command on the Target Port.

   If the Target Profile is enabled on an eSIM Port where CAT has not been initialised, the Device SHALL discard any state, including the PIN state, of the Target Profile and any cached file content including EF_{ICCID} and EF_{DIR}.
7. Upon reception of the Terminal Response or after the RESET of the Target Port, or immediately if the Target Profile is enabled on an eSIM Port where CAT has not been initialised, the ISD-R SHALL disable the currently Enabled Profile on the Target Port.
8. If the Target Profile is a Test Profile, an Operational Profile was in Enabled state while the Test Profile was enabled on the Target Port and this Operational Profile was not deleted while a Test Profile was Enabled, this previous Operational Profile SHALL be enabled again on the Target Port.

If refreshFlag is not set, steps 9 to 16 SHALL be executed as specified below.

9. The ISD-R SHALL disable the Target Profile.

   Check if a previous Operational Profile needs to be Enabled: If the Target Profile is a Test Profile, an Operational Profile was in Enabled state on the Target Port while the Test Profile was enabled and such Operational Profile was not deleted while a Test Profile was Enabled, the procedure SHALL end with the execution of steps 10 to 13:

10. This previous Operational Profile SHALL be enabled again on the Target Port.
11. The ISD-R SHALL return a response indicating result OK back to the LUId.
12. The Device SHALL discard any state, including the PIN state, of the previously Enabled Profile, any cached file content including EF_{ICCID} and EF_{DIR}, and any proactive command session.
13. The LPA SHALL signal the baseband that a new Profile was Enabled. The baseband triggers the execution of a network attach procedure with the newly Enabled Profile.

Otherwise: if no previous Operational Profile needs to be Enabled, the procedure SHALL end with the execution of steps 14 to 16.

14. The ISD-R SHALL return a response indicating result OK back to the LUId.
15. The Device SHALL discard any state, including the PIN state, of the previously Enabled Profile, any cached file content including EF_{ICCID} and EF_{DIR}, and any proactive command session.
16. The LPA SHALL signal the baseband that the Profile was disabled.

3.2.3 Delete Profile

This procedure is used to delete a Profile already downloaded and installed on an eUICC.

The conditions under which the LPAd MAY delete a Provisioning Profile are implementation-dependent and out of the scope of this specification. The eUICC implementation MAY not support deletion of a Provisioning Profile or a preloaded Test Profile.

```
@startuml
skinparam monochrome true
skinparam ArrowColor Black
hide footbox

participant "End-user" as EndUser #FFFFFF
participant "LUId" as LPA #FFFFFF
participant "eUICC \n LPA Services (ISD-R)" as LPAServices #FFFFFF
participant "Device Baseband" as Baseband #FFFFFF

note over EndUser, LPAServices #FFFFFF
     [0] End-user interactions
end note

group If Profile to be deleted is enabled
note over LPA, Baseband #FFFFFF
     [1] Steps 1-9 of the Disable Profile Procedure are executed
end note
end
LPA -> LPAServices: [2] ES10c.DeleteProfile(ICCID or ISD-P AID)

alt ISD-R checks the target Profile is in Enabled state or Profile Policy Rules
does not allow deletion
LPAServices -> LPA : [3] ERROR
else The target Profile is in a Disabled state and Profile Policy Rules allows
deletion
note over LPAServices #FFFFFF
[4] Delete Profile
end note
LPAServices -> LPA : [5] OK
end

@enduml
```

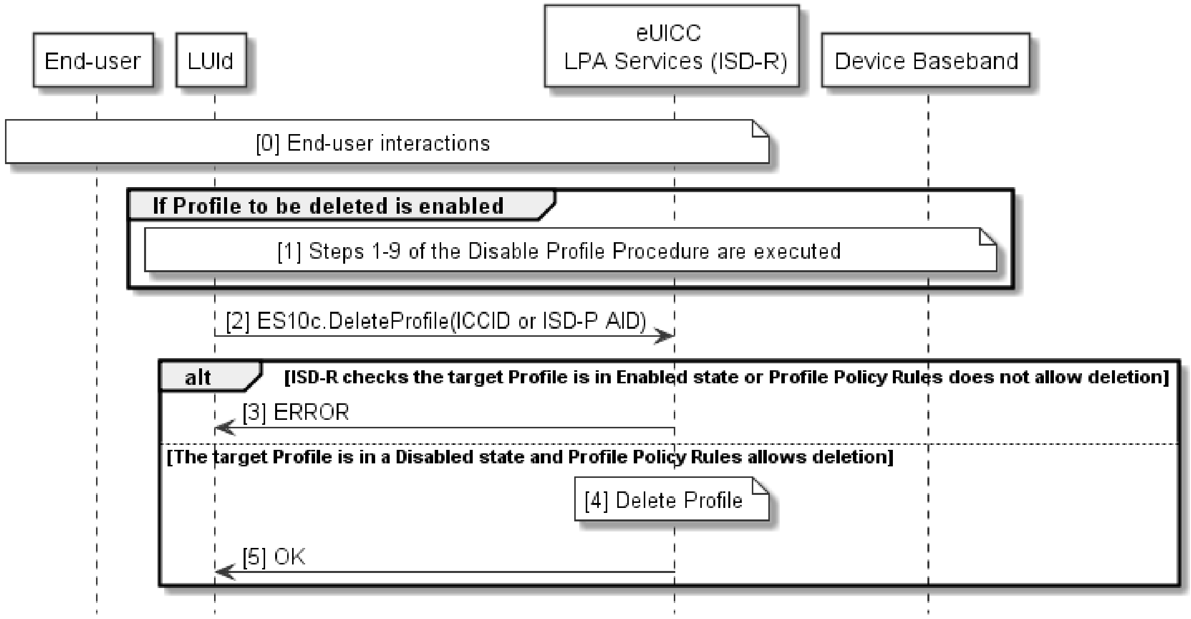

End-user
LUId
eUICC
LPA Services (ISD-R)
Device Baseband
[0] End-user interactions
If Profile to be deleted is enabled
[1] Steps 1-9 of the Disable Profile Procedure are executed
[2] ES10c.DeleteProfile(ICCID or ISD-P AID)
alt
[ISD-R checks the target Profile is in Enabled state or Profile Policy Rules does not allow deletion]
[3] ERROR
[The target Profile is in a Disabled state and Profile Policy Rules allows deletion]
[4] Delete Profile
[5] OK

Figure 18: Delete Profile

Start Conditions:

When the Profile to be deleted is an Operational Profile:

- User Intent is acquired as defined in SGP.21 [4].

When the Profile to be deleted is a Test Profile:

- The Device is in Device Test Mode.
- The Test Profile to be deleted is not a pre-loaded Test Profile, or the eUICC implementation permits deletion of the preloaded Test Profiles.

Procedure:

0. The End User is presented with a user interface that displays the list of installed Profiles within the eUICC, with their current states (Enabled or Disabled), as described in "List Profiles" procedure (section 3.2.4**Error! Reference source not found.**) The End User selects the Profile to be deleted. The LPAd SHALL ask for Strong Confirmation by presenting the consequences. The LPA MAY check the Profile Policy Rules of the Profile and give appropriate warnings to the End User (e.g., that due to Profile Policy Rules the Profile cannot be deleted). The deletion of a Provisioning Profile can be initiated by the LPAd itself without any End User interaction.
1. If the identified Profile to be deleted is Enabled on any eSIM Port then steps 1-9 of the disable profile procedure SHALL be executed as defined in section 3.2.2.
2. The LPAd SHALL call the "ES10c.DeleteProfile" function of the ISD-R with its relevant input data.
3. The ISD-R SHALL verify the state of the target Profile and check its Profile Policy Rules. If the target Profile is in the Enabled state or the Profile Policy Rules do not allow deletion, the ISD-R SHALL return a response indicating a failure, and the procedure SHALL stop.
4. The eUICC SHALL delete the Profile.
5. The ISD-R SHALL return a response indicating result OK back to the LPAd.

3.2.4 List Profiles

This procedure is used by the LPAd to list the Profiles, and their current states, pre-installed or previously downloaded and installed on an eUICC, in human readable format. The procedure is initiated by the LPAd either implicitly (e.g., at first Device boot up) or explicitly (e.g., through LUI user interface options).

```
@startuml
skinparam monochrome true
skinparam ArrowColor Black


hide footbox

participant "End-user" as EndUser #FFFFFF
participant "LUId" as LPA #FFFFFF
participant "eUICC \n LPA Services (ISD-R)" as LPAServices #FFFFFF


note over EndUser, LPA #FFFFFF
     [0] End-user interactions
end note
LPA -> LPAServices: [1] ES10c.GetProfilesInfo()
LPAServices -> LPA: [2] List of Profiles
LPA -> EndUser: [3] Display List
```

```
@enduml
```

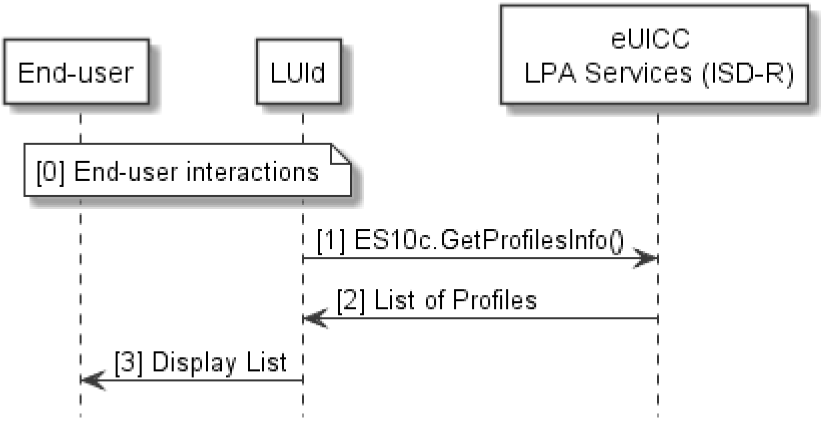


Figure 19: List Profiles

Start Conditions:

- None.

Procedure:

0. The LPAd is started on the Device. The user MAY be presented with the user interface options.
1. Either as part of the LPA launch procedure or through explicit user menu selection, the LPAd SHALL call "ES10c.GetProfilesInfo" to request the list of Profiles from the LPAd Services.
2. The eUICC SHALL return the Profile Metadata and status of the Profile(s) as defined in section 5.7.15.
3. The LUId SHALL display a subset of the set of installed Profiles along with their current states (Enabled or Disabled) to the End User in human readable format. This subset could be empty. The displayed subset SHALL include the Operational Profiles if the Device is not in Device Test Mode. It SHALL include the Test Profiles if the Device is in Device Test Mode. It SHALL NOT include the Provisioning Profiles.

End Conditions:

Any Profile information presented to the user SHALL always be in human readable format.

The specific presentation of Profile information to the End User is out of the scope of this document. However, it SHALL be possible for the End User to obtain the ICCID of each installed Operational Profile.

3.2.5 Add Profile

This procedure will allow the End User to add a single Profile. This procedure can further enable the downloaded Profile upon Confirmation Request, which consequently disables the currently Enabled Profile (if any). Network connectivity is assumed. The download can be initiated by the input of an Activation Code, by retrieval of a pending Profile download Event from an SM-DS, or by retrieval of a pending Profile download from a Default SM-DP+. The LPAd MAY implement a combination of these methods, as applicable, as a composite Add Profile operation.

If a Default SM-DP+ address is configured on the eUICC or Default SM-DP+ addresses are configured on the Device, the LPAd SHALL allow the End User to perform Add Profile operations using a configured Default SM-DP+ and to perform Add Profile operations using the Root SM-DS(s) that it supports. As described above, the LPAd MAY implement this as a composite Add Profile operation.

When the End User initiates the Add Profile procedure and the Profile Metadata indicates that the Profile is not an Operational Profile, the LPAd SHOULD notify the End User and stop the procedure.

```
@startuml
skinparam monochrome true
skinparam ArrowColor Black

hide footbox

participant "End User" as EndUser #FFFFFF
participant "LUId / LPDd" as LPA #FFFFFF
participant "eUICC \n LPA Services (ISD-R)" as LPAServices #FFFFFF
participant "SM-DP+" as SMDP #FFFFFF
participant "SM-DS" as SMDS #FFFFFF

rnote over EndUser, LPA #FFFFFF
      (1) End User interaction
end rnote

alt Activation Code
rnote over EndUser, LPA #FFFFFF
      (2a) Activation Code input
end rnote

else SM-DS
rnote over LPA, SMDS #FFFFFF
      (2b) Event Retrieval Procedure, see 3.6.2
end rnote
else Default SM-DP+
rnote over LPA, LPAServices #FFFFFF
      (2c) Default SM-DP+ address retrieval
end rnote
end

rnote over LPA, SMDP #FFFFFF
      (3) Profile Download and Installation Procedure, see 3.1.3
end rnote
@enduml
```

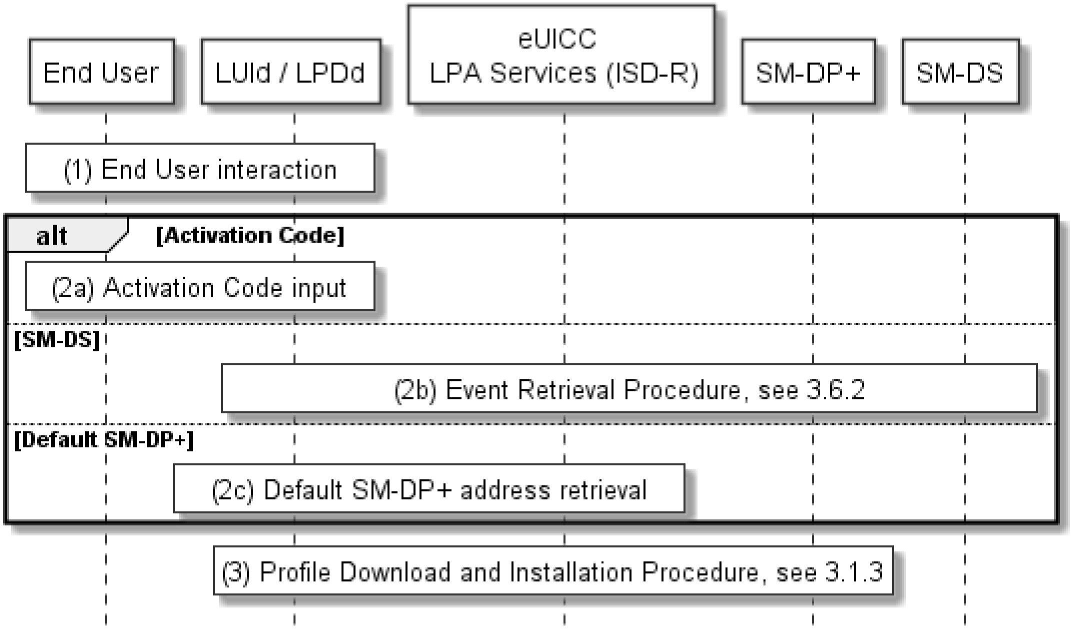

End User
LUId / LPDd
eUICC
LPA Services (ISD-R)
SM-DP+
SM-DS
(1) End User interaction
alt
[Activation Code]
(2a) Activation Code input
[SM-DS]
(2b) Event Retrieval Procedure, see 3.6.2
[Default SM-DP+]
(2c) Default SM-DP+ address retrieval
(3) Profile Download and Installation Procedure, see 3.1.3

Figure 20: Add Profile

Start Conditions:

- User Intent is acquired as defined in SGP.21 [4].

Procedure:

1. The End User initiates the Add Profile operation within the LUId.

   When performing subsequent Common Mutual Authentication procedure(s) with the SM-XX(s), the LPAd SHOULD set `operationType`, which, if present, SHALL include `profileDownload`.

2. The LPAd obtains the parameters for the Profile to be downloaded:

   a. If an Activation Code is used, the LUId SHALL obtain the Activation Code from the End User (e.g., by manual entry or QR code scanning).

   b. If an SM-DS is used, the LPAd SHALL retrieve the SM-DP+ address and EventID from that SM-DS using the Event Retrieval Procedure (section 3.6.2).

   c. If a Default SM-DP+ is used, the LPAd SHALL retrieve the Default SM-DP+ address from the eUICC or from the Device.

3. The Profile is downloaded from the SM-DP+ via the Profile download and installation procedure as defined in section 3.1.3, with the Confirmation Request(s) as described therein.

End Conditions:

1. The Profile and its associated Profile Metadata have been installed on the End User's eUICC.

3.2.6 Set/Edit Nickname

This procedure is used to add or change the Profile Nickname associated to a Profile already downloaded and installed on an eUICC.

This procedure is not applicable to Provisioning Profiles.

```
@startuml
hide footbox
skinparam sequenceMessageAlign center
skinparam sequenceArrowFontSize 12
skinparam noteFontSize 12
skinparam monochrome true

participant "End User" as User
participant "LUId" as LUI
participant "eUICC \n LPA Services (ISD-R)" as LPAsvc

rnote over User,LPAsvc #FFFFFF : [0] End User selects the target Profile
rnote over User,LUI #FFFFFF : [1] End User edits the nickname
LUI -> LPAsvc : [2] ES10c.SetNickname(ICCID, Nickname)
@enduml
```

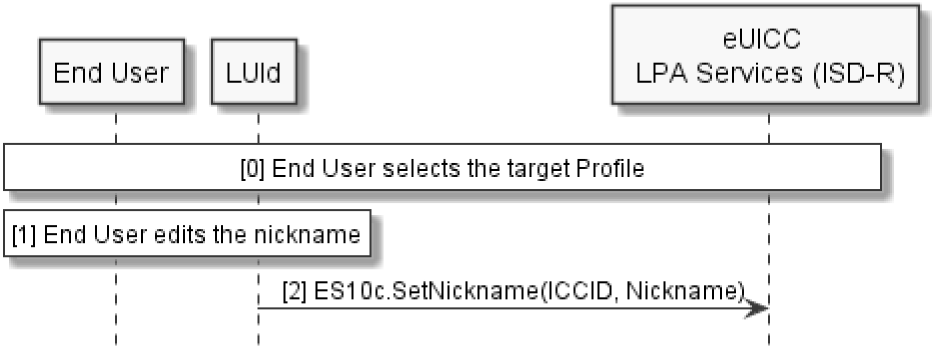

End User
LUId
eUICC
LPA Services (ISD-R)
[0] End User selects the target Profile
[1] End User edits the nickname
[2] ES10c.SetNickname(ICCID, Nickname)

Figure 21: Set/Edit Nickname

Start Conditions:

When the Profile to be renamed is an Operational Profile:

- User Intent is acquired as defined in SGP.21 [4].

When the Profile to be renamed is a Test Profile:

- The Device is in Device Test Mode.

Procedure:

0. The End User selects the Profile to be modified. For example, the End User MAY be presented with a user interface that displays the list of installed profiles within the eUICC, as described in the "List Profiles" procedure (section 3.2.4) from which the relevant Profile can be selected.
1. Through the LUId, the End User edits the Profile Nickname.
2. The LUId calls the function "ES10c.SetNickname" with the relevant ICCID and edited Nickname.

End Conditions:

The new Profile Nickname is stored in the Profile Metadata of the relevant Profile.

3.2.7 Update Profile

This procedure is used to update Profile(s) already downloaded and installed on an eUICC via RPM Command(s). Network connectivity is assumed.

```
@startuml
hide footbox
skinparam sequenceMessageAlign center
skinparam sequenceArrowFontSize 12
skinparam noteFontSize 12
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
hide footbox

participant "End User" as User
participant "LPA" as LPA
participant "eUICC \n LPA Services (ISD-R)" as LPAsvc
participant "SM-DP+" as DP
participant "SM-DS" as DS

rnote over User, LPAsvc : [1] End User interactions

loop As many times as the number of Polling Addresses
rnote over LPA, LPAsvc : [2] Retrieve the Polling Address
rnote over LPA, DS : [3] [Event Retrieval Procedure, see 3.6.2]
rnote over LPA, DP : [4] RPM Download and Execution (section 3.7.2)
end

@enduml
```

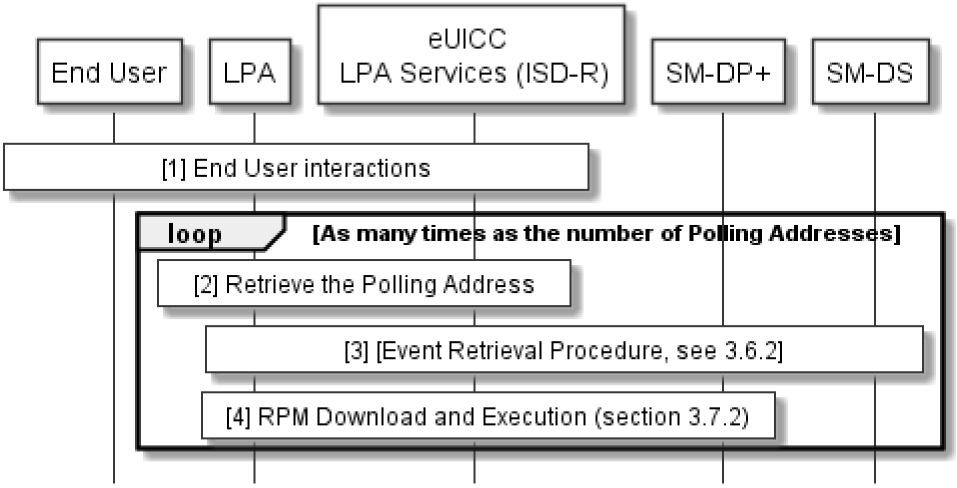

End User
LPA
eUICC
LPA Services (ISD-R)
SM-DP+
SM-DS
[1] End User interactions
loop
[As many times as the number of Polling Addresses]
[2] Retrieve the Polling Address
[3] [Event Retrieval Procedure, see 3.6.2]
[4] RPM Download and Execution (section 3.7.2)

Figure 21a: Update Profile

Start Conditions:

A Polling Address is present in the Profile Metadata of the target Profile.

When the target Profile is an Operational Profile:

- User Intent is acquired as defined in SGP.21 [4].

Procedure:

1. The End User selects target Profile(s) to be updated. The implementation of selecting the Profile is Device manufacturer specific. For instance:
   a) 'Update a Profile': The End User is presented with a user interface that displays the list of installed Profiles within the eUICC as described in "List Profiles" procedure (section 3.2.4). The End User selects a target Profile, and confirms its update. When performing subsequent Common Mutual Authentication procedure(s) with the SM-XX(s), the LPAd SHALL set `operationType` to include `rpm` and SHALL provide the ICCID of the target Profile.
   b) 'Update all Profiles': The End User is presented with a user interface describing that all installed Profiles will be updated. The End User confirms the update of all Profiles. When performing subsequent the Common Mutual Authentication procedure(s) with the SM-XX(s), the LPAd SHALL set `operationType` to include `rpm` and SHOULD provide the ICCID of the target Profile.

The LPAd iterates steps (2) to (4) for all Polling Addresses as required in step (1).

2. The LPAd retrieves the Polling Address of the target Profile.
3. (Optional) If the Polling Address is an SM-DS, the LPAd performs Event Retrieval procedure as described in section 3.6.2.
4. The LPAd retrieves and executes the RPM Command(s) from the SM-DP+ as defined in section 3.7.2, with the Confirmation Request(s) as described therein.

End Conditions:

The target Profile(s) is updated by the RPM Command(s).

3.2.8 Add/Update All

This procedure will allow the End User to add (a) new Profile(s) and update all installed Profiles in a single action. Network connectivity is assumed. The LPAd MAY support this procedure, as applicable, by combining Local Profile Management operations "Add Profile" (see section 3.2.5) and "Update Profile" (see section 3.2.7). If the LPAd supports this procedure, the LPAd SHALL set `operationType` to include both `profileDownload` and `rpm` when performing subsequent Common Mutual Authentication procedure(s) with the SM-XX(s).

3.3 Local eUICC Management

3.3.1 Retrieve EID

The Device SHALL provide means for the End User to access the EID in the numeric text representation described hereunder. The Device SHOULD provide means for the End User to access the EID in the QR code representation described hereunder. If the EID is provided to the End User by the LPAd, it is retrieved by the LPAd over the ES10c interface using the function "ES10c.GetEID" as described in section 5.7.20. The EID MAY also be provided to the End User by any other means in any other representation.

The numeric text representation SHALL comprise 32 digits, where each digit is represented by one character in the set [0123456789]. The QR code representation SHALL also be prefixed with "`EID:`" and SHALL be encoded in alphanumeric mode according to ISO/IEC 18004 [15].

When included in an Octet16 ASN.1 object, the first, third, fifth… digits SHALL be put into the highest four bits of the first, second, third… bytes.

NOTE: Presentation of the EID on the package of a Device should use the same QR code format or, alternatively, a barcode format. The EID should also be printed on the package as a barcode.

3.3.2 eUICC Memory Reset

This procedure is used to delete all the Operational Profiles and their associated Profile Metadata stored on the eUICC regardless of their status. The procedure is initiated by the End User using the LUI of the LPAd.

```
@startuml
skinparam monochrome true
skinparam ArrowColor Black

hide footbox

participant "End User" as EndUser #FFFFFF
participant "LUId" as LPA #FFFFFF
participant "eUICC\n LPA Services (ISD-R)" as LPAServices #FFFFFF
participant "Device Baseband" as Baseband #FFFFFF

note over EndUser, LPA #FFFFFF
     [0] End User interactions
end note
LPA -> LPAServices: [1] ES10c.eUICCMemoryReset(deleteOperationalProfiles)

note over LPAServices #FFFFFF
     [2] Delete all ISD-Ps with Operational Profiles
     and associated Profile Metadata,
     [reset Default SM-DP+ address to initial value]
end note

LPAServices --> LPA : [3] OK

alt SEP
LPAServices --> Baseband : [4] [REFRESH (UICC Reset)]
else MEP
LPAServices -> Baseband : [4] [LSI COMMAND (UICC Platform Reset)]
end
Baseband -> LPAServices: RESET

@enduml
```

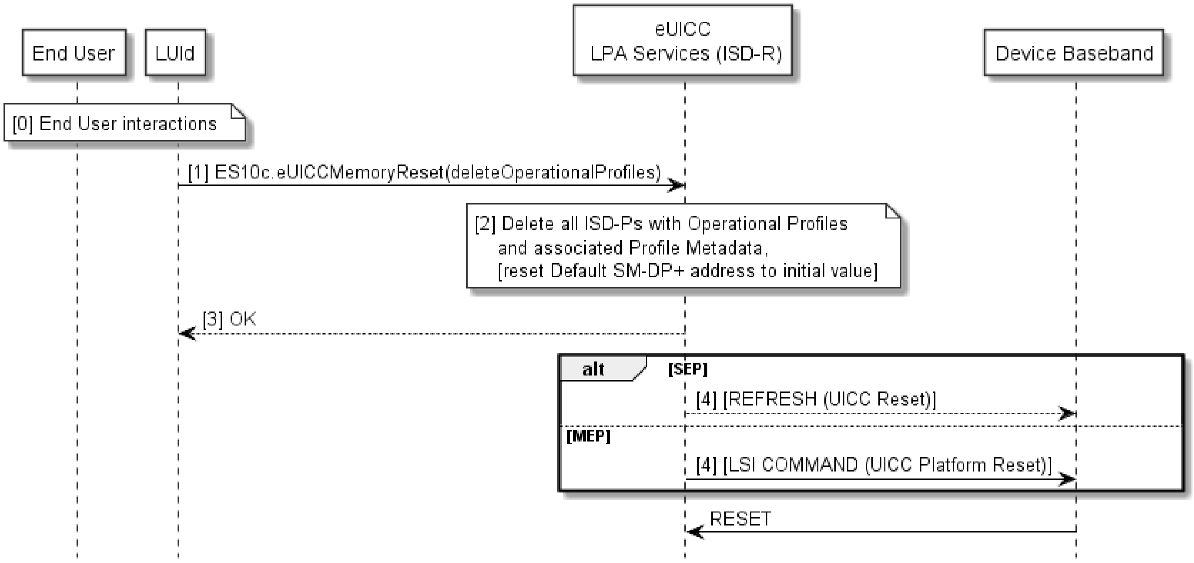

End User
LUId
eUICC
LPA Services (ISD-R)
Device Baseband
[0] End User interactions
[1] ES10c.eUICCMemoryReset(deleteOperationalProfiles)
[2] Delete all ISD-Ps with Operational Profiles
and associated Profile Metadata,
[reset Default SM-DP+ address to initial value]
[3] OK
alt
[SEP]
[4] [REFRESH (UICC Reset)]
[MEP]
[4] [LSI COMMAND (UICC Platform Reset)]
RESET

Figure 22: eUICC Memory Reset

Start Conditions:

- User Intent is acquired as defined in SGP.21 [4].

Procedure:

0. The End User initiates the eUICC Memory Reset. The LPAd SHALL ask for Strong Confirmation by presenting the consequences.
1. The LPAd SHALL call the "ES10c.eUICCMemoryReset" (section 5.7.19) function of the ISD-R, specifying that Operational Profiles are to be erased.
2. If a proactive session is ongoing on the Command Port:
    a) The function MAY return the appropriate error code and stop its execution. If so, the LPAd MAY take implementation-dependent actions to terminate the proactive command session, and MAY call again the "ES10c.eUICCMemoryReset" function without any further End User interaction.

   If the ISD-R does not stop execution due to an ongoing proactive session on the Command Port, it SHALL:

    b) Delete all ISD-Ps with Operational Profiles and their associated data and Profile Metadata.
    c) If required by the command: reset the Default SM-DP+ address to its initial value.
3. The ISD-R SHALL return a response indicating result OK back to the LUId.
4. If there was an Enabled Profile, the ISD-R SHALL send a proactive command to the Device to reset the eUICC. For SEP, the ISD-R SHALL send a REFRESH with mode "UICC Reset". For MEP, the ISD-R SHALL send an LSI COMMAND with "UICC Platform Reset".

NOTE: Instead of fetching the proactive command, the Device MAY reset the eUICC interface.

End Conditions:

The Operational Profiles and their associated Profile Metadata are deleted from the eUICC.

3.3.3 eUICC Test Memory Reset

This procedure is used to delete all the field-loaded (non-preloaded) Test Profiles and their associated Profile Metadata stored on the eUICC regardless of their status. The procedure is initiated by the End User using the LUI while the Device is in Test Mode.

This procedure is only required if the Device supports Test Mode.

```
@startuml
skinparam monochrome true
skinparam ArrowColor Black

hide footbox

participant "End User" as EndUser #FFFFFF
participant "LUId" as LPA #FFFFFF
participant "eUICC\n LPA Services (ISD-R)" as LPAServices #FFFFFF
participant "Device Baseband" as Baseband #FFFFFF

note over EndUser, LPA #FFFFFF
     [0] End User interactions
end note
LPA -> LPAServices: [1] ES10c.eUICCMemoryReset(deleteFieldLoadedTestProfiles)

note over LPAServices #FFFFFF
     [2] Delete all the ISD-Ps with Test Profiles
     and associated Profile Metadata
end note

LPAServices --> LPA : [3] OK

alt SEP
LPAServices --> Baseband : [4] [REFRESH (UICC Reset)]
else MEP
LPAServices -> Baseband : [4] [LSI COMMAND (UICC Platform Reset)]
end
Baseband -> LPAServices: RESET

@enduml
```

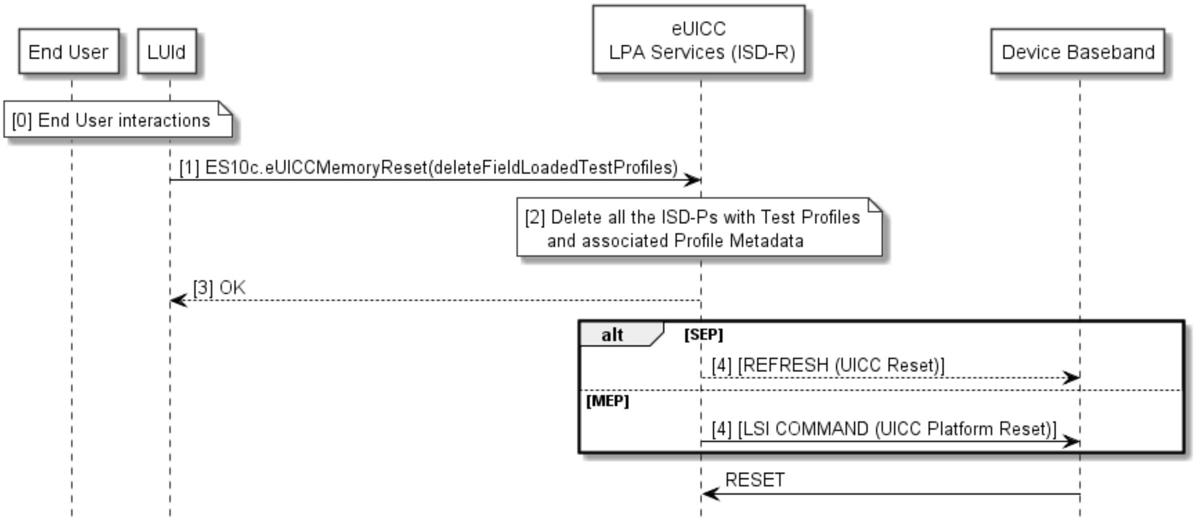

End User
LUId
eUICC
LPA Services (ISD-R)
Device Baseband
[0] End User interactions
[1] ES10c.eUICCMemoryReset(deleteFieldLoadedTestProfiles)
[2] Delete all the ISD-Ps with Test Profiles
and associated Profile Metadata
[3] OK
alt
[SEP]
[4] [REFRESH (UICC Reset)]
[MEP]
[4] [LSI COMMAND (UICC Platform Reset)]
RESET

Figure 23: eUICC Test Memory Reset

Start Conditions:

- The Device is in Test Mode
- User Intent is acquired as defined in SGP.21 [4].

Procedure:

0. The End User initiates the eUICC Test Memory Reset. The LPAd SHALL ask for Simple Confirmation by presenting the consequences.
1. The LPAd SHALL call the "ES10c.eUICCMemoryReset" (section 5.7.19) function of the ISD-R as defined in section 5.7.19, specifying that field-loaded (non-preinstalled) Test Profiles are to be erased.
2. If the eUICC does not support Test Profiles then the ISD-R SHALL return an OK result to the LPA and the procedure SHALL stop. Otherwise:
   a) If a proactive session is ongoing on the Command Port:
      The function MAY return the appropriate error code and stop its execution. If so, the LPAd MAY take implementation-dependent actions to terminate the proactive command session, and MAY call again the "ES10c.eUICCMemoryReset" function without any further End User interaction.
   b) If the ISD-R does not stop execution due to an ongoing proactive session on the Command Port, it SHALL delete all the selected ISD-Ps with their Profiles and their associated data and Profile Metadata.
3. The ISD-R SHALL return a response indicating result OK back to the LUId.
4. If there was an Enabled Profile, the ISD-R SHALL send a proactive command to the Device to reset the eUICC. For SEP, the ISD-R SHALL send a REFRESH with mode "UICC Reset". For MEP, the ISD-R SHALL send an LSI COMMAND with "UICC Platform Reset".

NOTE: Instead of fetching the proactive command, the Device MAY reset the eUICC interface.

End conditions:

The Test Profiles and their associated Profile Metadata are deleted from the eUICC.

3.3.4 Set/Edit Default SM-DP+ Address

This procedure is used to set or update the Default SM-DP+ address stored in an eUICC. The LPAd MAY also support an implementation-specific procedure to set or update a Default SM-DP+ address stored in the Device.

```
@startuml
hide footbox
skinparam sequenceMessageAlign center
skinparam sequenceArrowFontSize 12
skinparam noteFontSize 12
skinparam monochrome true

participant "End User" as User
participant "LUI" as LUI
participant "eUICC\n LPA Services (ISD-R)" as LPAsvc

LUI -> LPAsvc : [1] ES10a.GetEuiccConfiguredData
LPAsvc --> LUI : defaultDpAddress, allowedCiPKId, CI List
rnote over User,LPAsvc #FFFFFF : <b>End User interactions\n[2] End User is shown
the current Default SM-DP+ address\n    [and currently allowed CI]\n[3] End User
enters a new Default SM-DP+ address\n     [and selects CI]
LUI -> LPAsvc : [4] ES10a.SetDefaultDpAddress(defaultDpAddress[, allowedCiPKId])
```

```
@enduml
```

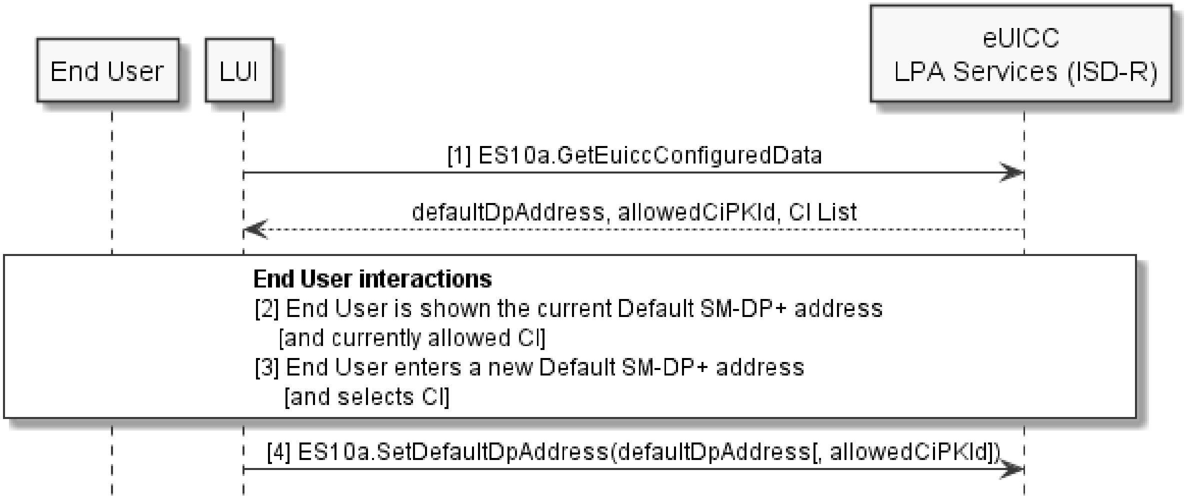

End User
LUI
eUICC
LPA Services (ISD-R)
[1] ES10a.GetEuiccConfiguredData
defaultDpAddress, allowedCiPKId, CI List
End User interactions
[2] End User is shown the current Default SM-DP+ address
[and currently allowed CI]
[3] End User enters a new Default SM-DP+ address
[and selects CI]
[4] ES10a.SetDefaultDpAddress(defaultDpAddress[, allowedCiPKId])

Figure 23a: Set/Edit Default SM-DP+ Address

Start Conditions:

- User Intent is acquired as defined in SGP.21 [4].

Procedure:

1. The LUId calls the function "ES10a.GetEuiccConfiguredData" to retrieve the Default SM-DP+ address and allowed eSIM CA RootCA public key, currently set in the eUICC. Each MAY be an empty value. In addition, the function provides the list of eSIM CA RootCA public key identifiers that the eUICC supports for signature verification together with eSIM CA names.
2. The End User MAY be presented with a user interface that displays the current Default SM-DP+ address. The LUId MAY also display the currently allowed eSIM CA, if any.
3. Through the LUId, the End User enters a new Default SM-DP+ address. The LUId SHALL allow setting an empty value. The LUId MAY display the list of the names of the eSIM CAs supported by the eUICC and allow the End User to select one or an empty value.
4. The LUId calls the function "ES10a.SetDefaultDpAddress" with the new Default SM-DP+ address and identifier of the allowed eSIM CA RootCA public key chosen by the End User, if any.

End Conditions:

The Default SM-DP+ address is updated with the value set by the End User. If the End User selected an eSIM CA, then the allowed eSIM CA RootCA public key is updated with its identifier; otherwise, no allowed eSIM CA RootCA public key is set.

3.3.5 Retrieve DEV-IC

The Device supporting the DEV-IC SHOULD provide a means for the End User to access the DEV-IC in the text and/or the QR code representation defined in section 4.X. If the Device contains multiple eUICCs and the End User has selected one of the eUICCs, the DEV-IC SHALL contain the EID of the selected eUICC in the Path field. The Device MAY also provide a means for the End User to access the DEV-IC in any other means in any other representation.

3.4 Device and eUICC Initialisation

3.4.1 eUICC Initialisation

The eUICC SHALL indicate its support of eUICC functionality in ATR Global Interface byte as defined in ETSI TS 102 221 [6].

An MEP-capable eUICC SHALL indicate its support of LSIs in the ATR Global Interface byte and an MEP-Capable Device SHALL indicate in PPS2 if LSIs are to be used for the card session as defined in ETSI TS 102 221 [6]. eSIM Port 0 SHALL always be implicitly selected by the eUICC at the beginning of a card session.

Unless pre-configured between the MEP-Capable Device and the MEP-Capable eUICC, and before using additional eSIM Ports, an MEP-capable Device SHALL perform the setup of the LSI configuration as defined in ETSI TS 102 221 [6] with the additional TLVs defined in clause 3.4.1.1.

NOTE: It is not required to perform the LSI configuration immediately after PPS.

Upon the indication of support for eUICC in the ATR, the LPAd MAY obtain additional eUICC information, such as SVN, as described in this section below and in section 5.7.1.

If the eUICC contains an Enabled Profile on an eSIM port, the eUICC initialisation procedure SHALL follow the UICC activation procedure as defined in ETSI TS 102 221 [6].

NOTE: For SEP, the term eSIM Port is considered as the physical interface.

If the eUICC does not contain an Enabled Profile on an eSIM Port, but only the default file system as described in section 3.4.3, containing only a limited number of files, the Device SHALL be able to initialise the eUICC and send a Terminal Profile command on this eSIM Port as defined in ETSI TS 102 223 [31] indicating at least that REFRESH (UICC Reset Mode) proactive command is supported.

The Device MAY select the ISD-R to determine if there is an Enabled Profile on the eUICC during eUICC initialisation through FCI template of the ISD-R as defined in section 5.7.1. If there is no Enabled Profile on the eUICC, the Device SHALL maintain the card session between the Device and the eUICC.

If the ISD-R is selected by the Device during eUICC initialisation, the Device MAY send the information contained in the ISDRProprietaryApplicationTemplate data object to the LPA.

The Device SHALL NOT expect the ISD-R to be multi-selectable.

3.4.1.1 Additional TLVs for MANAGE LSI (Configure LSIs)

The command data of the MANAGE LSI (configure LSIs) SHALL contain the following additional TLVs:

Byte(s)	Description	Value	Length
1	Tag for MEP mode(s) of the Device	'90'	1
2	Length of next field	N	1
3 to N+2	MEP mode(s) supported by the Device in the order of priority '01': MEP-A1 '02': MEP-A2 '03': MEP-B		N = 1 to 3
N+3	Tag for maximum number of LSIs for Enabled Profiles of the Device	'91'	1
N+4	Length of next field	1	1
N+5	Maximum number of LSIs supported for Enabled Profiles		1

Table 6d: Configure LSIs additional command data

- The response data of the MANAGE LSI (configure LSIs) SHALL contain the following additional TLVs:

Byte(s)	Description	Value	Length
1	Tag for MEP modes	'90'	1
2	Length of next field	1+M	1
3	Jointly supported MEP mode, see Table 6d for coding Set to '00' in case of no jointly supported MEP mode.		1
4 to M+3	All MEP modes supported by the eUICC (including the mode given in byte 3) in arbitrary order, see Table 6d for coding		M
M+4	Tag for jointly supported maximum number of LSIs for Enabled Profiles	'91'	1
M+5	Length of next field	1	1
M+6	Maximum number of LSIs jointly supported for Enabled Profiles		1

Table 6e: Configure LSIs additional response data

If there is no jointly supported MEP mode, the eUICC SHALL set the value field of the jointly supported MEP mode to '00'. Subsequent actions for the Device (e.g., switching to SEP mode) are out of scope.

NOTE 1: This setup mechanism allows Devices and eUICCs to support several modes.

NOTE 2: As MEP is only defined for non-removable eUICCs, reconfiguration between different MEP modes, between MEP and SEP or when changing the number of eSIM Ports (e.g., assignment of Profiles to the eSIM Ports or implicit disabling of Profiles) does not need to be defined.

NOTE 3: For MEP-A1 and MEP-A2, Device and eUICC use the dedicated eSIM Port 0 for communication with the ISD-R in addition to the eSIM Ports for enabled Profiles. For MEP-B, no additional eSIM Port is used.

Examples
(The value fields of the TLVs are highlighted.)

Example 1:

Command data for a Device supporting a maximum of 3 LSIs, MEP-B (preferred) and MEP-A1, and a maximum of 2 Enabled Profiles:

'80 01 **03** 90 02 **03 01** 91 01 **02**'

Response data for an eUICC supporting MEP-A1 and MEP-A2, with 3 or more LSIs and 2 or more Enabled Profiles:

'80 01 **03** 90 03 **01 02 01** 91 01 **02**'

Jointly supported and used for the card session: 3 LSIs, MEP-A1, and a maximum of 2 Enabled Profiles.

Example 2:

Device as in example 1. Response data for an eUICC supporting MEP-A1, MEP-A2 and MEP-B, with 2 LSIs and 2 Enabled Profiles:

'80 01 **02** 90 04 **03 01 02 03** 91 01 **02**'

Jointly supported and used for the card session: 2 LSIs, MEP-B, and a maximum of 2 Enabled Profiles.

Example 3:

Device as in example 1. Response data for an eUICC supporting MEP-A2 only, with 3 LSIs and 2 Enabled Profiles:

'80 01 **03** 90 02 **00 02** 91 01 **02**'

No jointly supported MEP mode.

Example 4:

Command data for a Device supporting a maximum of 5 LSIs, MEP-A1 (preferred) and MEP-A2, with a maximum of 4 Enabled Profiles:

'80 01 **05** 90 02 **01 02** 91 01 **04**'

Response data for an eUICC supporting MEP-A2 and MEP-B, but only 4 LSIs and 3 Enabled Profiles:

'80 01 **04** 90 03 **02 02 03** 91 01 **03**'

Jointly supported and used for the card session: 4 LSIs, MEP-A2, and a maximum of 3 Enabled Profiles.

3.4.2 RSP Device Capabilities

The eUICC SHALL request the Device to send the Terminal Capability command by setting the related bit in the file control parameters of the MF.

The Device SHALL report its support of LPA functions using the Terminal Capability command data defined in ETSI TS 102 221 [6]. This command SHALL be sent before the SELECT ISD-R command defined in section 5.7.1.

Within the Terminal Capability template (tag 'A9'), the tag '83' is used for indicating the Device's support for eUICC related functions.

The support of LPA and some Device capabilities is indicated in the first byte within the TLV object under tag '83':

b8	b7	b6	b5	b4	b3	b2	b1	Meaning
-	-	-	-	-	-	-	1	Local User Interface in the Device (LUId) supported
-	-	-	-	-	-	-	0	Local User Interface in the Device (LUId) not supported
-	-	-	-	-	-	1	-	Local Profile Download in the Device (LPDd) supported
-	-	-	-	-	-	0	-	Local Profile Download in the Device (LPDd) not supported
-	-	-	-	-	1	-	-	Local Discovery Service in the Device (LDSd) supported
-	-	-	-	-	0	-	-	Local Discovery Service in the Device (LDSd) not supported
-	-	-	-	1	-	-	-	LUIe based on SCWS supported
-	-	-	-	0	-	-	-	LUIe based on SCWS not supported
-	-	-	1	-	-	-	-	Metadata update alerting supported
-	-	-	0	-	-	-	-	Metadata update alerting not supported
-	-	1	-	-	-	-	-	Enterprise Capable Device
-	-	0	-	-	-	-	-	Non-Enterprise Capable Device
-	1	-	-	-	-	-	-	LUIe using E4E supported
-	0	-	-	-	-	-	-	LUIe using E4E not supported
1	-	-	-	-	-	-	-	LPR supported
0	-	-	-	-	-	-	-	LPR not supported

Table 7: eUICC-related Device Capabilities byte 1

For LPAd implementations according to this version of the specification, b1, b2 and b3 SHALL either all be set to 1 or all be set to 0.

The eUICC SHALL only enable the functions of ES10c if the Device indicates support for the LUId.

The eUICC SHALL only enable the functions of ES10b if the Device indicates support for the LPDd.

The eUICC SHALL only enable the functions of ES10a if the Device indicates support for the LDSd.

The conditions for enabling the LPAe are defined in section 5.7.1.

If the Device and eUICC are MEP-Capable, the Terminal Capability sent to the eUICC SHALL be the same on all eSIM Ports.

Support for Metadata update alerting is optional for the Device. However, a Device that caches any Metadata SHOULD support it in order to get informed about Metadata updates that happen via ES6. If Metadata update alerting is supported, the Device SHALL also indicate support for REFRESH with "Application Update" mode in the Terminal Profile according to ETSI TS 102 223 [31] on each eSIM Port where the ISD-R can be selected (i.e., eSIM Port 0 for MEP-A1 and MEP-A2; each eSIM Port where a Profile is enabled for SEP and MEP-B).

For Non-Enterprise Capable Devices, the eUICC SHALL restrict the installation of Profiles with Enterprise Rules as specified in this document. For Enterprise Capable Devices, Enterprise Rules are allowed. For subsequent operation (e.g., enabling or Metadata update), this bit in the Device Capabilities is not (again) evaluated.

A device supporting LUIe using E4E SHALL NOT set bits 1 to 6 of byte 1.

Subsequent bytes are RFU.

3.4.3 eUICC File Structure

If there is no Enabled Profile on the eUICC, the eUICC SHALL ensure a default file system is available to the Device. This file system SHALL contain at least the MF and MAY contain the MF-level EFs shown below.

- $EF_{ENV\text{-}CLASSES}$
- EF_{UMPC}

It SHALL NOT be possible to modify either file via ES6 of a Profile.

$EF_{ENV\text{-}CLASSES}$ SHALL never be present in any Profile Package, however if present, it SHALL be ignored by the eUICC when installing the Profile.

EF_{UMPC} MAY be present in a Profile Package. If present and EF_{UMPC} is also present in the default file system, the second byte of the default file SHALL take the content of the Profile Package into account when this Profile is enabled: it SHALL be set to the higher of both values. The eUICC SHALL ignore the content of all the other bytes of the file present in the Profile Package. If present and EF_{UMPC} is not present in the default file system, the whole file from the Profile SHALL be taken into account.

When a Profile is enabled, the eUICC SHALL present a file system comprising that Profile's file system and the EFs listed above if existing.

3.4.4 Device Setup and Power-on Profile Discovery

As part of Device setup, means SHALL exist for the End User to retrieve pending Profiles via a Default SM-DP+(s) if configured, via Root SM-DS(s), and via the Activation Code procedure. The implementation is Device manufacturer specific. Profile discovery during Device setup MAY be implemented as a special case of Device power-on Profile discovery as described hereunder. Alternatively, it MAY be implemented in some other manner; for example, by informing the End User how to retrieve the pending Profiles after Device setup is completed.

The means by which the LPA detects Device setup is out of the scope of this specification.

When appropriate for the class and usage of the device, the LPA SHALL conditionally perform Profile discovery when the Device is powered on, rebooted, or reset. In addition the LPA MAY support an End User configurable parameter that enables or disables this operation.

When it is supported, the initial value of the configuration parameter SHALL be 'Enabled', and its value SHALL be persistent across Device reset and power cycles.

The specific point at which power-on Profile discovery occurs and the means by which the LPA is launched to perform Profile discovery are Device-specific and out of the scope of this specification. It SHALL be performed if all of the following conditions are satisfied:

- Power-on Profile discovery is appropriate for the class and usage of the Device. (For example, this could be inappropriate for an open-market cellular-enabled notebook computer).
- No Operational Profile is installed on the eUICC.
- The value of the configuration parameter is 'Enabled'.

When all of these conditions are satisfied the LPA SHALL perform the following steps:

1. The LPA SHALL perform the following two sub-steps in any order:
   a. If there is one or more configured Default SM-DP+ address in the eUICC or the Device, then the LPA SHALL initiate the Profile download and installation procedure with each Default SM-DP+ as defined in section 3.1.3, using the Default SM-DP+ address and a missing Matching ID.
   b. If there is one or more configured SM-DS address, then the LPA SHALL initiate the event retrieval procedure with each SM-DS where events could be expected as defined in section 3.6.2, with no EventID, and process any retrieved Profile download(s).

      NOTE: It is out of scope of this specification how the LPA determines from which SM-DS, if any, to expect events. This may be based e.g., on regional or deployment information.
2. If no Operational Profile was downloaded in step 1, the LPA MAY prompt the End User to add a Profile using an Activation Code (such as by manual entry or QR code scanning).

When downloading Profile(s) as part of Device setup, the LPA SHALL perform the Confirmation Request(s) as described in section 3.1.3.

3.5 Notifications

This section describes the procedure to provide a report (in the context of this procedure referred to as "Notification") to a remote entity (Recipient Address) that a Profile Management Operation was successfully performed on the eUICC or about the progress of such an operation. The remote entity receiving the Notification is the SM-DP+ or the Operator.

Several types of Notifications are defined on ES9+: Profile Installation Results, generated as an answer to the Profile installation, see section 2.5.6; Load RPM Package Result generated as an answer to RPM command execution, see section 2.10.2; and Other Notifications, generated due to the 'NotificationConfigurationInformation' data object defined in the "ES8+.StoreMetadata" function. The eUICC SHALL manage the storage of these types of Notifications separately.

A Notification on ES9+ is made up of the following fields:

- Sequence Number (generated by the eUICC)
- Profile Management Operation (i.e., the event whose occurrence SHALL be notified)
- Recipient Address (i.e., SM-DP+ address)
- ICCID, present as specified below
- Additional data in case of a Profile Installation Result, see section 2.5.6
- Additional data in case of a Load RPM Package Result, see section 2.10.2
- eUICC signature
- eUICC certificate chain in case of an Other Notification, see section 5.7.10

Sequence Number is used to protect the recipient against replay attacks. Each time a new Notification is generated by the eUICC, the Sequence Number SHALL be incremented. There SHALL be a single Sequence Number counter per eUICC, which SHALL be used across all Profiles and for all types of Notifications. Neither an eUICC Memory Reset nor a reset of the eUICC SHALL affect this Sequence Number.

The ICCID is used to identify the Profile upon which the operation, leading to the generation of the Notification, has been performed. ICCID SHALL conditionally be present as follows:

- For a Profile Installation Result, ICCID SHALL be present if known by the eUICC.
- For a Load RPM Package Result, ICCID SHALL NOT be present.
- For an Other Notification, ICCID SHALL be present.

Each Profile Metadata MAY contain Notification Configuration Information for Other Notifications, made up of the following:

- Profile Management Operation
- Recipient Address (i.e., SM-DP+ address)

When the Profile Management Operation indicated in the Notification Configuration Information is successfully performed, the eUICC SHALL generate and store a Notification.

When an eUICC Memory Reset is successfully performed, a Local Delete Notification SHALL be generated for all the deleted Profiles for which the local delete Profile operation is indicated in the Notification Configuration List.

When an Enable Profile is successfully performed and the currently enabled Operational Profile is implicitly disabled as a consequence of the enable Profile, the Notifications on both the disable Profile and enable Profile SHALL be generated, provided that each operation is indicated in the Notification Configuration List.

The LPA retrieves the pending Notifications and SHALL send the Notifications one at a time. Each Notification is sent on a best-effort basis, as described below, to the recipient when connectivity is available.

After the LPA receives the acknowledgement from the SM-DP+ that the Notification was received, it SHALL communicate to the eUICC to remove that Notification.

When the eUICC needs to store a new Other Notification and if there is not enough space, the eUICC SHALL delete one or more of the previously stored Other Notifications in order of their Sequence Number, beginning with the lowest.

The LPA SHALL sort the Notifications and group them by SM-DP+. The LPA SHALL send Notifications of a group one by one according to the sequence number, lowest number (oldest Notification) first. The next Notification of a group SHALL NOT be sent until LPA receives a success or failure response from the SM-DP+ for the previous Notification. The LPA MAY send Notifications from different groups sequentially or in parallel (i.e., there is no need to wait for the acknowledgment of a Notification from one group before sending a Notification from another group).

The SM-DP+ SHALL forward notifications as agreed with the Operator owning the Profile.

```
@startuml
hide footbox
skinparam sequenceMessageAlign center
skinparam sequenceArrowFontSize 11
skinparam noteFontSize 11
skinparam monochrome true
skinparam lifelinestrategy solid

participant "Operator " as Operator1 #FFFFFF
participant "SM-DP+" as SMDP1 #FFFFFF
participant "LPAd" as LPA #FFFFFF
Participant "eUICC" as eUICC #FFFFFF

alt Notifications are retrieved from the eUICC LPA -> eUICC : [1]
[ES10b.ListNotification]
eUICC -> LPA : [2] [List of pending Notification Metadata]

LPA -> eUICC : [3] ES10b.Retrieve PendingNotificationsList
eUICC -> LPA : [4] List of pending Notifications
else Notifications are retrieved from another LPAd (e.g., in Device Change)
rnote over LPA #FFFFFF
[4a] LPAd receives Notification(s) from another LPAd
end rnote
end
rnote over LPA #FFFFFF
[4b] LPAd sorts the Notifications according to the SM-DP+s
end rnote
loop for each SM-DP+
rnote over SMDP1, LPA #FFFFFF
[5] TLS session establishment
end rnote
loop for each Notification (sent to this SM-DP+)
LPA -> SMDP1 : [6] ES9+.HandleNotification(notification)
```

```
alt Notification ack failed
SMDP1 --> LPA : \n[7a] Error
else Notification ack successful
SMDP1 -> LPA : \n[7b] Ok
rnote over SMDP1 #FFFFFF
[7c] Verify signature
[7d] [Verify ICCID]
[7e] Handle sequence number
end rnote
SMDP1 -> Operator1 : [8] ES2+.HandleNotification(notification)
Operator1 -> SMDP1 : [9] Ok
LPA -> eUICC : [10] ES10b.RemoveNotificationFromList(SeqNumber)
rnote over eUICC #FFFFFF
[11] eUICC removes
this Notification
from the storage
end rnote
end
end
end

@enduml
```

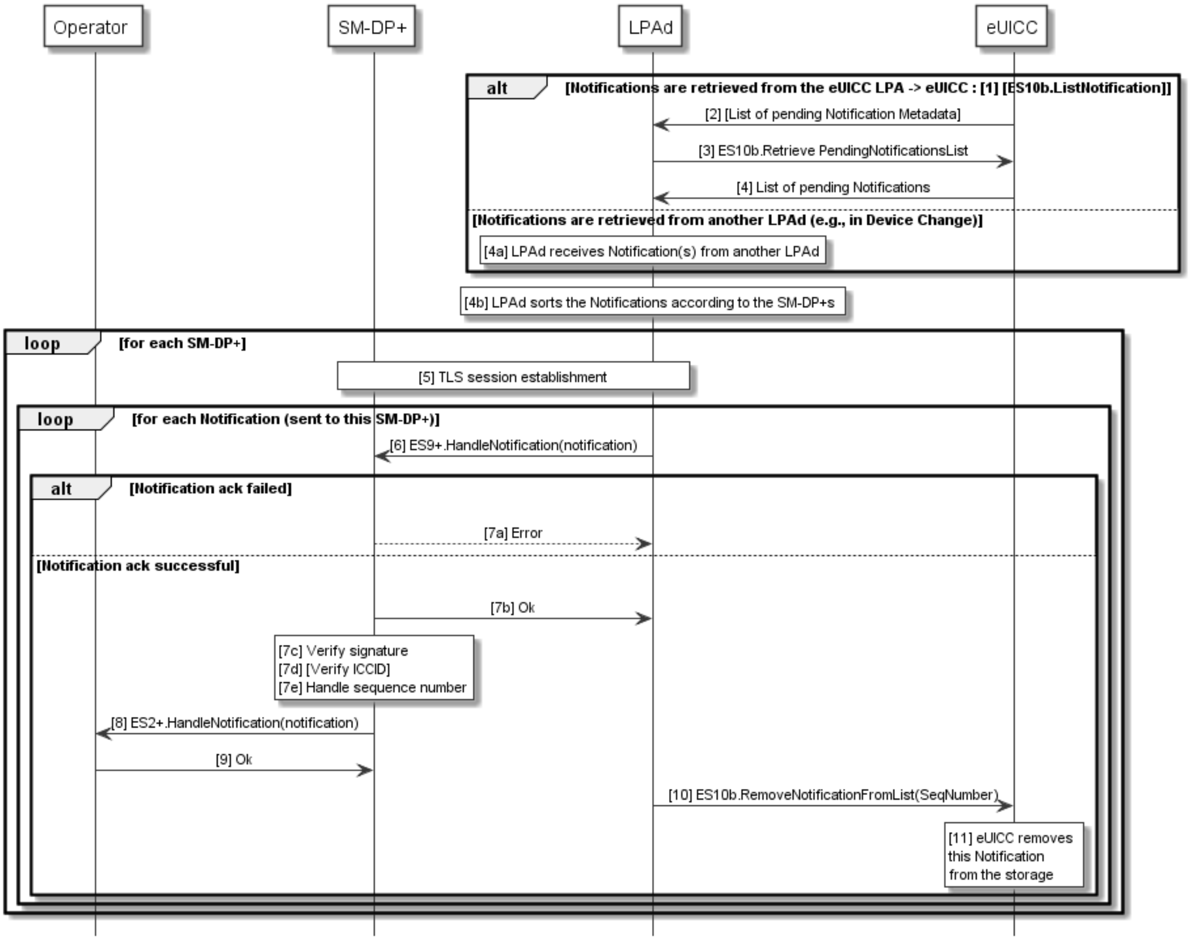

Operator
SM-DP+
LPAd
eUICC
alt
[Notifications are retrieved from the eUICC LPA -> eUICC : [1] [ES10b.ListNotification]]
[2] [List of pending Notification Metadata]
[3] ES10b.Retrieve PendingNotificationsList
[4] List of pending Notifications
[Notifications are retrieved from another LPAd (e.g., in Device Change)]
[4a] LPAd receives Notification(s) from another LPAd
[4b] LPAd sorts the Notifications according to the SM-DP+s
loop
[for each SM-DP+]
[5] TLS session establishment
loop
[for each Notification (sent to this SM-DP+)]
[6] ES9+.HandleNotification(notification)
alt
[Notification ack failed]
[7a] Error
[Notification ack successful]
[7b] Ok
[7c] Verify signature
[7d] [Verify ICCID]
[7e] Handle sequence number
[8] ES2+.HandleNotification(notification)
[9] Ok
[10] ES10b.RemoveNotificationFromList(SeqNumber)
[11] eUICC removes
this Notification
from the storage

Figure 24: Sending of Notifications

The figure above illustrates the sending of Notifications to two distinct SM-DP+s in a sequential manner for ease of representation. However, the LPA MAY send Notifications to SM-DP+ n and SM-DP+ m in parallel. The figure illustrates also the forwarding of Notifications to two distinct Operators.

Start Conditions:

A Profile has been Enabled, Disabled, Installed or Deleted.

Procedure:

The following steps 1 to 4 are performed if the LPAd retrieves Notification(s) from the eUICC:

1. Optionally, the LPAd queries the eUICC for the List of Pending Notifications.
2. The eUICC provides the LPAd with the List of Pending Notification Metadata.
3. The LPAd queries the eUICC for the Pending Notifications List.
4. The eUICC provides the LPAd with the Pending Notifications List.

The following step 4a is performed if the LPAd retrieves Notification(s) from another LPAd (e.g., in case of Device Change):

4a. The LPAd (e.g., the LPAd of new Device in Device Change) receives Notification(s) from another LPAd (e.g., the LPAd of old Device in Device Change).

4b. The LPAd sorts the Notifications according to the addressed SM-DP+s.

The following steps 5 to 11 are performed for each addressed SM-DP+:

5. The LPAd establishes a TLS secure channel with the SM-DP+.

The following steps 6 to 11 are performed for each Notification for this SM-DP+ in order of their sequence numbers:

6. The LPAd sends the Notification to the SM-DP+.

7a. The SM-DP+ SHOULD return one of HTTP status code 4XX or 5XX for an error preventing the PendingNotification from being processed.

7b. The SM-DP+ SHALL return HTTP status code 204 if the PendingNotification has been successfully received, even if the sequence number is lower than or equal to the highest sequence number of all valid Notifications received so far.

7c. The SM-DP+ SHALL verify the signature of the notification. Processing for Notifications failing this verification SHALL stop.

7d. If the SM-DP+ has the information which Profile was loaded onto which eUICC, it SHALL verify that EID and ICCID of the notification are matching.
Actions upon a failure of these verifications are out of scope of this specification.

7e. If the sequence number is lower than or equal to the highest sequence number of all valid Notifications received so far, the SM-DP+ SHALL NOT update the Profile state and SHOULD NOT forward the notification to the Operator.

8. The SM-DP+ forwards the Notification to the Operator.
9. The Operator acknowledges Notification reception.
10. The LPAd:
    - SHALL call the "ES10b.RemoveNotificationFromList" function upon receiving HTTP status code 204.
    - SHALL NOT call the "ES10b.RemoveNotificationFromList" function upon receiving HTTP status code 5XX.
    - MAY call the "ES10b.RemoveNotificationFromList" function upon receiving HTTP status code 4XX.

11. The eUICC removes the Notification from the Pending Notifications List.

Steps 6 – 11 SHALL be repeated per each Notification in the Pending Notifications List.

3.6 SM-DS

3.6.0 Requirements

A Root SM-DS SHALL process Event Registrations and Event Retrieval in a non-discriminatory manner.

An SM-DS SHOULD protect itself to avoid becoming a point of injection for DoS or spam attacks.

An SM-DS MAY limit the lifetime of Event Records based upon service agreements for operational reasons. If this was a cascaded registration to a Root SM-DS, the SM-DS SHALL first delete the cascaded event on that Root SM-DS.

NOTE: the lifetime of an Event Record can expire if, e.g., it is not retrieved or processed by the LPA due to any reason including filtering, or it is not properly deleted by the SM-DP+.

3.6.1 Event Registration

Profile download events and RPM events are registered to a Root SM-DS specified by the Operator, optionally cascaded through an Alternative SM-DS specified by the Operator.

If the Operator specifies an SM-DS using the string `'.unspecified'`, the SM-DP+ SHOULD determine the SM-DS in an implementation-dependent manner. (As one example, the SM-DP+ could default to the GSMA Root SM-DS.) The SM-DP+ MAY also interpret additional strings that begin with the prefix `'.x-'` (e.g., `'.x-example'`) in an implementation-dependent manner.

3.6.1.1 Event Registration without Cascade

This procedure applies when the SM-DP+ registers an event directly to a Root SM-DS.

```
@startuml
hide footbox
skinparam sequenceMessageAlign center
skinparam sequenceArrowFontSize 11
skinparam noteFontSize 11
skinparam monochrome true
skinparam lifelinestrategy solid

participant "<b>Operator" as O
participant "<b>SM-DP+" as DP
participant "<b>SM-DS" as DS

O -> DP : ES2+.ConfirmOrder or\nES2+.RpmOrder

DP -> DS : [1] ES12.RegisterEvent\n(EID, SM-DP+ address, EventID,
ForwardingIndicator=false,\nEventType, [HashedIccid(s)], [Salt],
[ServiceProviderName], [OperatorId])

rnote over DS #FFFFFF
[2] Store the Event Record
   together with SM-DP+ OID
endrnote
DS --> DP : [3] OK
```

```
DP --> O : OK

@enduml
```

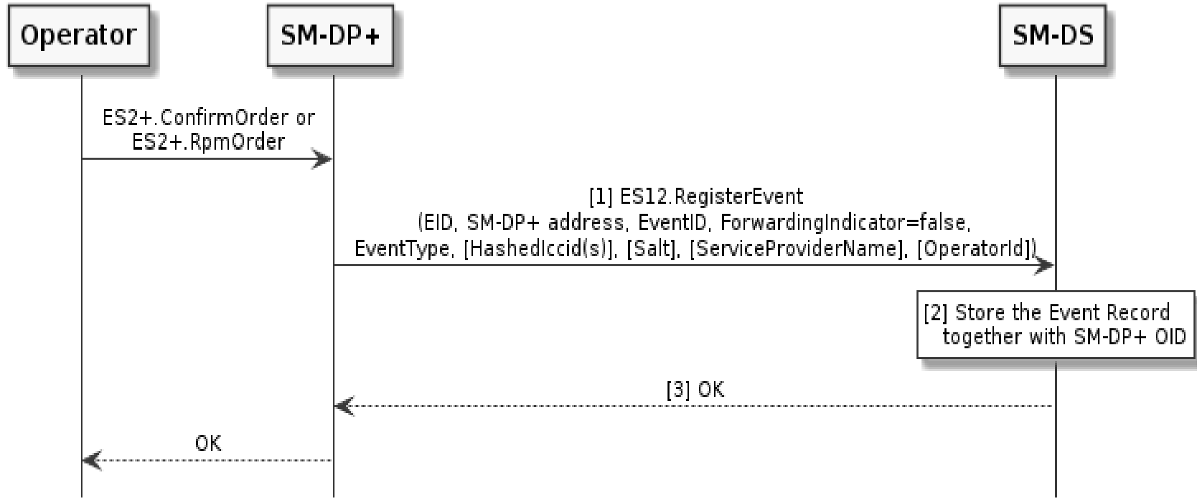

Operator
SM-DP+
SM-DS
ES2+.ConfirmOrder or
ES2+.RpmOrder
[1] ES12.RegisterEvent
(EID, SM-DP+ address, EventID, ForwardingIndicator=false,
EventType, [HashedIccid(s)], [Salt], [ServiceProviderName], [OperatorId])
[2] Store the Event Record
together with SM-DP+ OID
[3] OK
OK

Figure 25: Event Registration Procedure without Cascade

Start Conditions:

The Operator places a Profile download or RPM order to the SM-DP+ with a Root SM-DS Address, as described in section 3.6.1.

The SM-DP+ generates an EventID that is used to uniquely identify within its context either the Profile download order or RPM order.

EventIDs SHALL be unique per SM-DP+ and SHALL NOT be reused.

NOTE: This allows the LPA to keep a trace of already processed events and detect events still pending at an SM-DS that have been already processed.

The SM-DP+ and SM-DS are mutually authenticated. The SM-DP+ OID has been retrieved from the TLS certificate used for mutual authentication.

Procedure:

1. The SM-DP+ calls "ES12.RegisterEvent" function comprising of EID, RSP Server address, EventID, ForwardingIndicator set to 'false', EventType, and optionally HashedIccid(s), Salt, ServiceProviderName, and OperatorId.
2. The SM-DS stores the received Event Record, together with the SM-DP+ OID retrieved from the SM-DP+ Certificate. The value of ForwardingIndicator SHALL be ignored by the Root SM-DS.
3. The SM-DS acknowledges the registration.

3.6.1.2 Event Registration with Cascade

This procedure applies when the SM-DP+ registers an event to an Alternative SM-DS, which in turn registers it to a Root SM-DS.

```
@startuml
hide footbox
```

```
skinparam sequenceMessageAlign center
skinparam sequenceArrowFontSize 11
skinparam noteFontSize 11
skinparam monochrome true
skinparam lifelinestrategy solid

participant "<b>Operator" as O
participant "<b>SM-DP+" as DP
participant "<b>Alt SM-DS" as ADS
participant "<b>Root SM-DS" as RDS

O -> DP : ES2+.ConfirmOrder or\nES2+.RpmOrder

DP -> ADS : [1] ES12.RegisterEvent\n(EID, SM-DP+ address, EventID1,
ForwardingIndicator=true, Root SM-DS address\nEventType, [HashedIccid(s)], [Salt],
[ServiceProviderName], [OperatorId])

rnote over ADS #FFFFFF
[2] Generate EventID2
[3] Store the Event Record
   together with EventID2, SM-DP+ OID, Root SM-DS address
Endrnote

ADS -> RDS : [4] ES15.RegisterEvent\n(EID, Alt SM-DS address, EventID2,
ForwardingIndicator=false,\nEventType, [HashedIccid(s)], [Salt],
[ServiceProviderName], [OperatorId])

rnote over RDS #FFFFFF
[5] Store the Event Record
   together with Alt SM-DS OID
Endrnote

RDS --> ADS : [6] OK
ADS --> DP : [7] OK
DP --> O : OK

@enduml
```

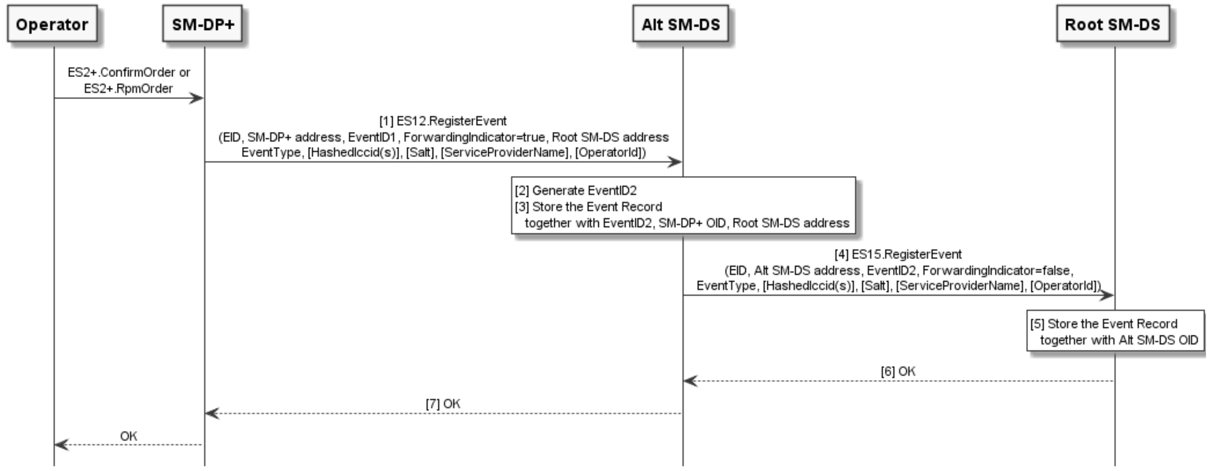

Operator
SM-DP+
Alt SM-DS
Root SM-DS
ES2+.ConfirmOrder or
ES2+.RpmOrder
[1] ES12.RegisterEvent
(EID, SM-DP+ address, EventID1, ForwardingIndicator=true, Root SM-DS address
EventType, [HashedIccid(s)], [Salt], [ServiceProviderName], [OperatorId])
[2] Generate EventID2
[3] Store the Event Record
together with EventID2, SM-DP+ OID, Root SM-DS address
[4] ES15.RegisterEvent
(EID, Alt SM-DS address, EventID2, ForwardingIndicator=false,
EventType, [HashedIccid(s)], [Salt], [ServiceProviderName], [OperatorId])
[5] Store the Event Record
together with Alt SM-DS OID
[6] OK
[7] OK
OK

Figure 26: Event Registration Procedure with Cascade

Start Conditions:

The Operator places a Profile download or RPM order to the SM-DP+ with an Alternative SM-DS address and/or a Root SM-DS address, as described in section 3.6.1.

The requirements for EventIDs in section 3.6.1.1 SHALL also apply to EventID1 and EventID2; the latter SHALL be unique, and not be re-used, per Alternative SM-DS.

The SM-DP+ and Alternative SM-DS are mutually authenticated. The SM-DP+ OID has been retrieved from the TLS certificate used for mutual authentication.

The Alternative SM-DS and Root SM-DS are mutually authenticated. The Alternative SM-DS OID has been retrieved from the TLS certificate used for mutual authentication.

Procedure:

1. The SM-DP+ calls "ES12.RegisterEvent" function comprising of EID, RSP Server address of SM-DP+, EventID1, ForwardingIndicator set to 'true', Root SM-DS address (FQDN), EventType, and optionally HashedIccid(s), Salt, ServiceProviderName, and OperatorId.
2. As the ForwardingIndicator indicates forwarding of the registration, the Alternative SM-DS generates a new EventID2.
3. The Alternative SM-DS stores the received Event Record, together with EventID2, the SM-DP+ OID retrieved from the SM-DP+ Certificate, and the address of the Root SM-DS.
4. The Alternative SM-DS calls "ES15.RegisterEvent" function of the Root SM-DS comprising EID, RSP Server address of the Alternative SM-DS, generated EventID2, ForwardingIndicator set to 'false', and EventType. This function call further comprises HashedIccid(s), Salt, ServiceProviderName, and OperatorId if these are received in step (1).
5. The Root SM-DS stores the received Event Record, together with the SM-DS OID retrieved from the Alternative SM-DS Certificate.
6. The Root SM-DS acknowledges the registration.
7. The Alternative SM-DS acknowledges the registration.

3.6.2 Event Retrieval

```
@startuml
hide footbox
skinparam sequenceMessageAlign center
skinparam sequenceArrowFontSize 11
skinparam noteFontSize 11
skinparam monochrome true
skinparam lifelinestrategy solid

participant "<b>SM-DS" as DS
participant "<b>LPAd" as LPA
participant "<b>eUICC" as E

rnote over DS, E #FFFFFF : [1] [Refer to Common mutual authentication procedure
section 3.0.1]

rnote over DS #FFFFFF
[2]
- Look for pendingEvent Record(s) for the eUICC
Endrnote

group Cond MatchingId not found.
DS --> LPA : error
group Cond. eUICC indicated signedSmdsResponseV3Support
rnote over LPA, E : Refer to Common Cancel Session procedure section 3.0.2
end
end

alt LPA and SM-DS both support signed Event Records
rnote over DS #FFFFFF
```

```
[3]
- [Build smdsSigned2 = {TransactionID, list of Event Records, [ECID],
[pushServiceRefreshTime]}]
- [Compute smdsSignature2 over smdsSigned2 and euiccSignature1]
endrnote

DS --> LPA : [4] TransactionID, smdsSigned2, smdsSignature2

rnote over LPA #FFFFFF : [5] Verify SM-DS response

group Cond. signed Event Records are missing
rnote over LPA, E : Refer to Common Cancel Session procedure section 3.0.2
end
group Cond. eUICC supports signed Event Records

LPA -> E : [6] ES10a.VerifySmdsResponse
rnote over E #FFFFFF : [7] Verify smdsSignature2
E --> LPA : ok/error
end

else Otherwise
DS --> LPA : [8] TransactionID, Event Records
end alt

rnote over LPA #FFFFFF
[9] [Filter the Event Record(s)]
[10] [Process the Event Record(s) sequentially]
endrnote

opt ECID is received
rnote over LPA #FFFFFF : [11] Store and associate the ECID with the SM-DS address
end opt

@enduml
```

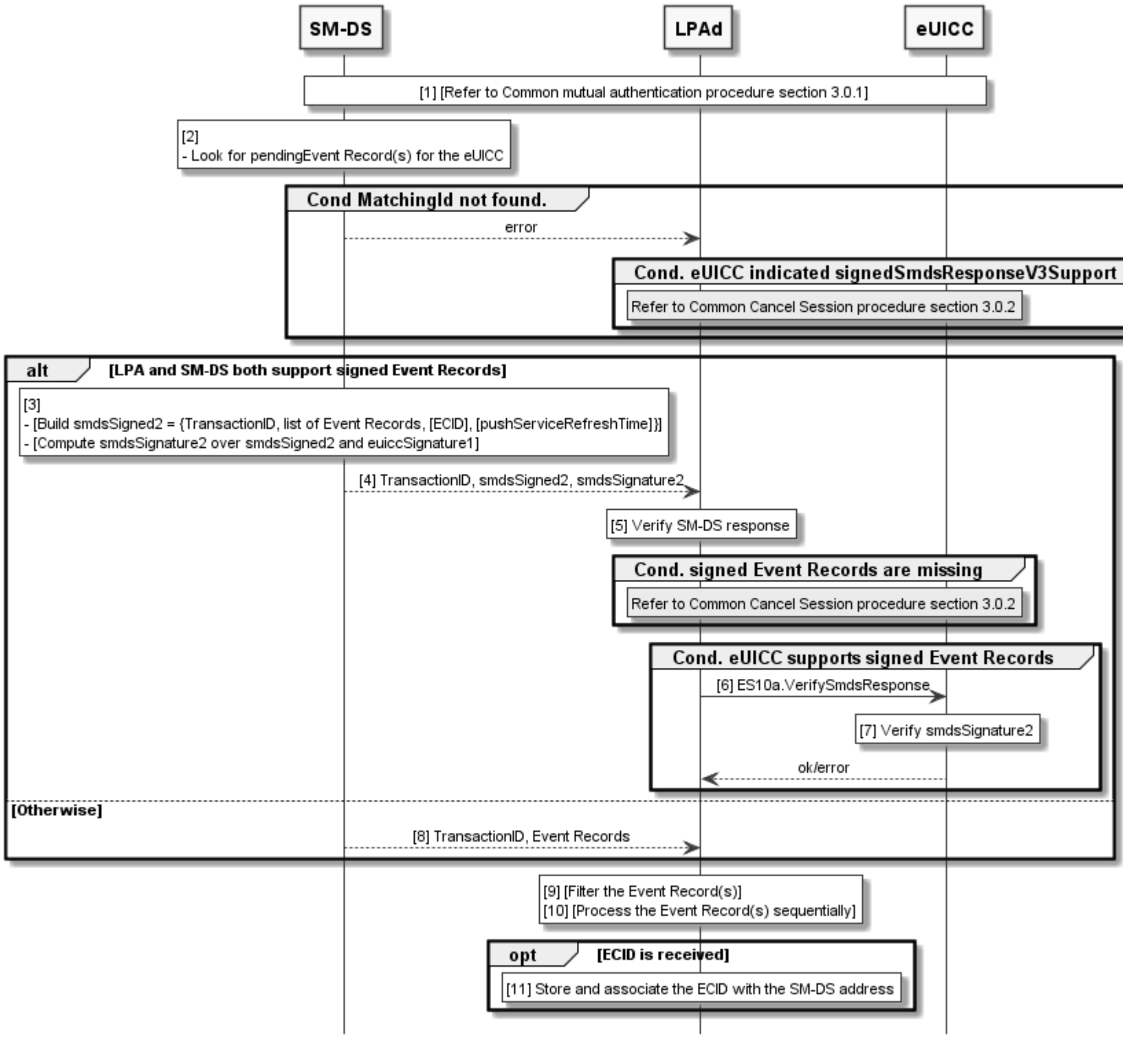

SM-DS
LPAd
eUICC
[1] [Refer to Common mutual authentication procedure section 3.0.1]
[2]
- Look for pendingEvent Record(s) for the eUICC
Cond MatchingId not found.
error
Cond. eUICC indicated signedSmdsResponseV3Support
Refer to Common Cancel Session procedure section 3.0.2
alt [LPA and SM-DS both support signed Event Records]
[3]
- [Build smdsSigned2 = {TransactionID, list of Event Records, [ECID], [pushServiceRefreshTime]}]
- [Compute smdsSignature2 over smdsSigned2 and euiccSignature1]
[4] TransactionID, smdsSigned2, smdsSignature2
[5] Verify SM-DS response
Cond. signed Event Records are missing
Refer to Common Cancel Session procedure section 3.0.2
Cond. eUICC supports signed Event Records
[6] ES10a.VerifySmdsResponse
[7] Verify smdsSignature2
ok/error
[Otherwise]
[8] TransactionID, Event Records
[9] [Filter the Event Record(s)]
[10] [Process the Event Record(s) sequentially]
opt [ECID is received]
[11] Store and associate the ECID with the SM-DS address

Figure 27: Event Retrieval Procedure

Start Conditions:

In addition to the start conditions required by the common mutual authentication procedure defined in section 3.0.1, Event(s) are registered on the SM-DS by one or more SM-DP+(s)/SM-DS(s).

The LPAd has determined the set of configured Root SM-DS addresses. It MAY retrieve one or more Root SM-DS addresses configured on the eUICC (using ES10a.GetEuiccConfiguredData). In addition, it MAY retrieve them from where they are configured on the Device.

As a result of previous Event Retrieval(s), if the LPAd received ECID(s) for the corresponding SM-DS(s) and stored them into the Device, the LPAd SHOULD initiate the Event Checking procedure, as described in section 3.6.4, to the SM-DS prior to the Event Retrieval procedure. The LPAd SHOULD use the Root SM-DS address(es) configured on a removable eUICC.

The event retrieval procedure is used in the following cases:

a) To retrieve Events from an SM-DS (a Root SM-DS or a Profile Polling Address) when there is no EventID. This includes, but is not limited to, the following trigger conditions:

- The End User MAY manually query for pending Event Records from the configured SM-DS(s). The LUId MAY implement this query in combination with other related operations, for example, as a composite 'Add Profile' operation.
- The LPAd MAY query the configured SM-DS(s) as part of Device power-on Profile discovery as described in section 3.4.4.
- The LPAd MAY query the SM-DS(s) if the received `isPendingEvent` was set to `true` as a response of the ES11.CheckEvent from the configured SM-DS(s).
- In the case of the Profile Polling Address, if an allowed eSIM CA RootCA public key identifier is stored in the Profile, the LPAd SHALL restrict the eSIM CA RootCA public key identifiers to that value.
- The LPAd MAY query the SM-DS(s) if the LPAd received a push notification of the SM-DS(s) from the push client hosted on the Device.
- In the case of the Root SM-DS(s) the LPAd SHALL NOT restrict the eSIM CA RootCA public key identifiers that are used for this procedure.

b) To retrieve an Event from an SM-DS with a specific EventID. This corresponds to the retrieval of a cascaded Event from the alternative SM-DS. If there was a restriction to a single eSIM CA RootCA public key identifier to obtain the EventID, then the LPAd SHALL apply the same restriction in this procedure.

Procedure:

1. The common mutual authentication procedure defined in section 3.0.1 SHALL be executed. When this procedure is used for SM-DS, SM-XX, CERT.XXauth.SIG, PK.XXauth.SIG, SK.XXauth.SIG, and ESXX are SM-DS, CERT.DSauth.SIG, PK.DSauth.SIG, SK.DSauth.SIG, and ES11, respectively.

   In addition, the LPAd SHALL build the ctxParams1 data object to provide the MatchingID, the MatchingID Source, the operationType, and Device Info to the eUICC for signature. The value of the MatchingID and MatchingID Source SHALL be set as follows:

   - For case a), the MatchingID SHALL be missing, and the MatchingID Source value SHALL be set to 'none'.
   - For case b), the MatchingID value SHALL be set to the EventID found in the Event Record that LPAd is processing, and the MatchingID Source value SHALL be set to the OID of the Root SM-DS that provided the EventID.

2. After having successfully authenticated the eUICC at the end of the step (1), the SM-DS SHALL ignore the matchingIdSource, and iccid (if any) parameters contained in the "ES9+.AuthenticateClient" function call, and check if there are pending Event Records matching the following criteria, where the absence of operationType SHALL be regarded as a Profile Download request:

   - If MatchingID is missing: the EID and the EventType in an Event Record match the EID in the CERT.EUICC.SIG obtained during step (1) and the operationType in the function call, respectively (Case a)). The resulting list MAY be empty.

- If MatchingID has a value of non-zero-length:
  - the EventID2, the EID, and the EventType in an Event Record match the EventID in the MatchingID, the EID in the CERT.EUICC.SIG obtained during step (1), and the operationType in the function call, respectively (case b)).
  - If no Event Record is found, an appropriate error status SHALL be returned to the LPAd. If the eUICC indicated `signedSmdsResponseV3Support`, the LPAd SHALL send "ES10b.CancelSession" to the eUICC with a reason `sessionAborted`. The procedure SHALL stop.

If the LPAd indicated `signedSmdsResponseV3Support` and the SM-DS indicated `eventListSigningV3Support` during Common Mutual Authentication, the procedure continues with step 3; otherwise, it continues with step 8.

3. The SM-DS SHALL generate an smdsSigned2 data structure containing the `TransactionID`, Event Record(s) , optionally the ECID, and `pushServiceRefreshTime` and compute the `smdsSignature2` over `smdsSigned2` and `euiccSignature1` using the SK.DSauth.SIG. The Event Record(s) SHALL be ordered as registered in the SM-DS.
4. The SM-DS SHALL respond to the LPAd with `TransactionID`, `smdsSigned2`, and `smdsSignature2`.
5. If `smdsSigned2` or `smdsSignature2` is missing, then the LPAd SHALL send "ES10b.CancelSession" to the eUICC with a reason `sessionAborted`, and the procedure SHALL stop.
   If the eUICC indicated `signedSmdsResponseV3Support` during Common Mutual Authentication:
6. The LPAd SHALL call the "ES10a.VerifySmdsResponse" function to verify the SM-DS signature.
7. The eUICC SHALL verify `smdsSignature2` and return success or an error status to the LPAd. If the eUICC returns an error the procedure SHALL stop, otherwise the procedure SHALL continue with step 9.

   NOTE: If the eUICC did not indicate `signedSmdsResponseV3Support`, the LPAd MAY verify `smdsSignature2` itself.

If either the LPAd or SM-DS does not support signed Event Records:

8. The SM-DS SHALL respond to the LPAd with `TransactionID` and unsigned Event Records. The Event Record(s) SHALL be ordered as registered in the SM-DS.

Finally:

9. The LPAd SHOULD filter the received Event Record(s) based upon the information contained therein and Profile Metadata of the installed Profile(s). Example of the filtering includes, but not limited to:
   - The LPAd can check if the Profile that will be installed by the Event Record is already installed. If so, the LPAd can ignore the Event Record.
   - The LPAd can check if the target Profile(s) of the RPM Package that will be received by the Event Record is available. If not, the LPAd can ignore the Event Record.

- The LPAd can check if the Service Provider and/or Operator is Profile Owner of the target Profile(s). If not, the LPAd can ignore the Event Record.

10. If the filtered Event Record list is not empty, the LPAd SHALL process the received Event Records in the order received, by sequentially contacting each RSP Server with the corresponding EventID with MatchingID source set to the OID of the SM-DS from which the Events were retrieved.

    NOTE: The LPA MAY also process the Event Record that are filtered out in step (9).

11. If the ECID was received in the response from an SM-DS, the LPAd SHOULD store and associate the ECID with the SM-DS address.

3.6.3 Event Deletion

3.6.3.1 Event Deletion without Cascade

```
@startuml
hide footbox
skinparam sequenceMessageAlign center
skinparam sequenceArrowFontSize 11
skinparam noteFontSize 11
skinparam monochrome true
skinparam lifelinestrategy solid

participant "<b>SM-DP+" as DP
participant "<b>SM-DS" as DS

DP -> DS : [1] ES12.DeleteEvent (EID, EventID)

rnote over DS #FFFFFF
[2]
- Retrieve the Event Record
Endrnote

DS --> DP : [error]

rnote over DS #FFFFFF
[2a]
- Delete the Event Record
endrnote
DS --> DP : [3] Deletion Status

@enduml
```

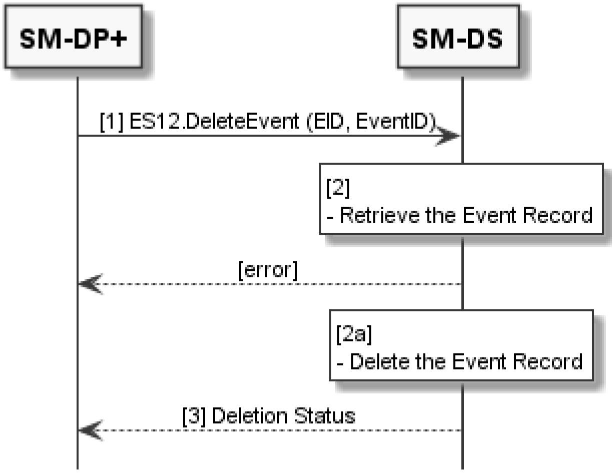

SM-DP+
SM-DS
[1] ES12.DeleteEvent (EID, EventID)
[2]
- Retrieve the Event Record
[error]
[2a]
- Delete the Event Record
[3] Deletion Status

Figure 28: Event Deletion Procedure without Cascade

Start Conditions:

An Event Record is stored in the SM-DS, which is identified by the EID, EventID and SM-DP+ OID.

The related Registration was not cascaded.

The SM-DP+ and SM-DS are mutually authenticated. The SM-DP+ OID has been retrieved from the TLS certificate used for mutual authentication.

Procedure:

1. The SM-DP+ calls "ES12.DeleteEvent" function including the EID and EventID.
2. The SM-DS SHALL verify that there is an Event Record identified by the SM-DP+ OID and EventID. If no Event Record is found, the SM-DS SHALL return the relevant error status and the procedure SHALL stop.

2a. The SM-DS SHALL delete the Event Record identified by the SM-DP+ OID and EventID.

3. The SM-DS SHALL return the deletion status.

3.6.3.2 Event Deletion with Cascade

```
@startuml
hide footbox
skinparam sequenceMessageAlign center
skinparam sequenceArrowFontSize 11
skinparam noteFontSize 11
skinparam monochrome true
skinparam lifelinestrategy solid

participant "<b>SM-DP+" as DP
participant "<b>Alt SM-DS" as ADS
participant "<b>Root SM-DS" as RDS

DP -> ADS : [1] ES12.DeleteEvent (EID, EventID1)

rnote over ADS #FFFFFF
[2]
- Retrieve the Event Record and
  associated Root SM-DS address
endrnote

ADS --> DP : [error]

ADS -> RDS : [3] ES12.DeleteEvent\n(EID, EventID2)

rnote over RDS #FFFFFF
[4]
- Retrieve the Event Record
Endrnote

RDS --> ADS : [error]

rnote over RDS #FFFFFF
[4a]
- Delete the Event Record
```

```
Endrnote

RDS --> ADS : [5] Deletion Status

rnote over ADS #FFFFFF
[6]
- [Delete the Event Record]
endrnote

ADS --> DP : [7] Deletion Status

@enduml
```

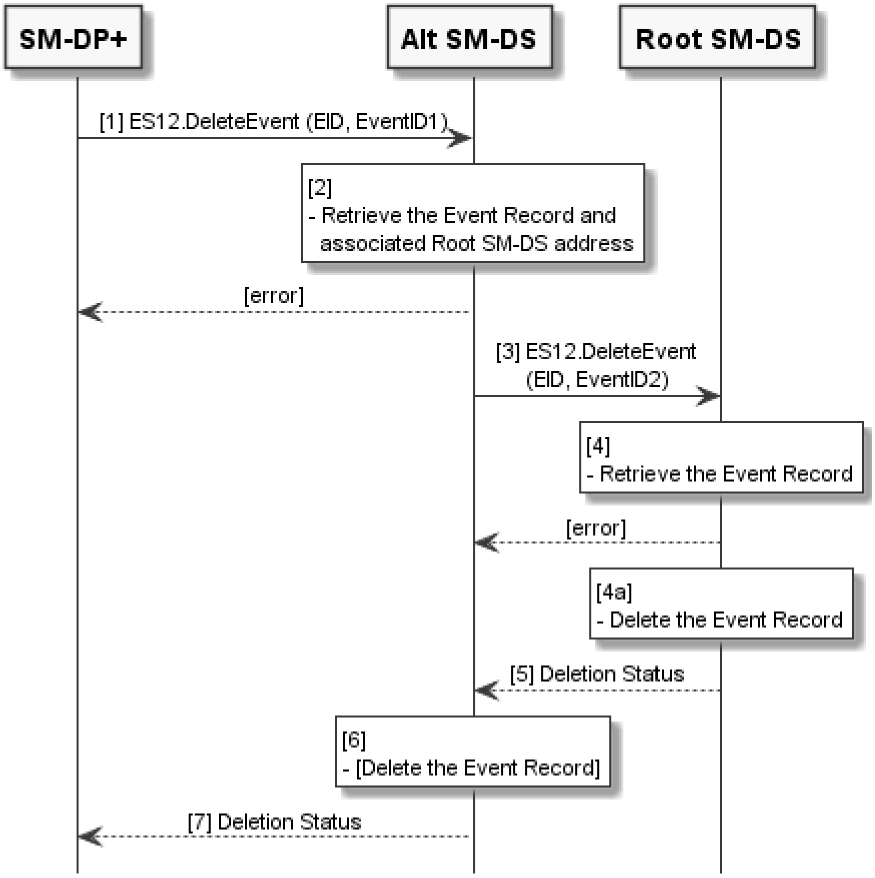

SM-DP+
Alt SM-DS
Root SM-DS
[1] ES12.DeleteEvent (EID, EventID1)
[2]
- Retrieve the Event Record and
associated Root SM-DS address
[error]
[3] ES12.DeleteEvent
(EID, EventID2)
[4]
- Retrieve the Event Record
[error]
[4a]
- Delete the Event Record
[5] Deletion Status
[6]
- [Delete the Event Record]
[7] Deletion Status

Figure 29: Event Deletion Procedure with Cascade

Start Conditions:

Start condition is same as for section 3.6.3.1 except the following:

The related Registration was cascaded. The Alternative SM-DS has stored the Root SM-DS associated with each cascaded Event Record.

Procedure:

1. The SM-DP+ calls "ES12.DeleteEvent" function including the EID and EventID1.
2. The Alternative SM-DS SHALL verify that there is a stored Event Record identified by the SM-DP+ OID and EventID1. If no Event Record is found, the SM-DS SHALL return the relevant error status and the procedure SHALL stop.

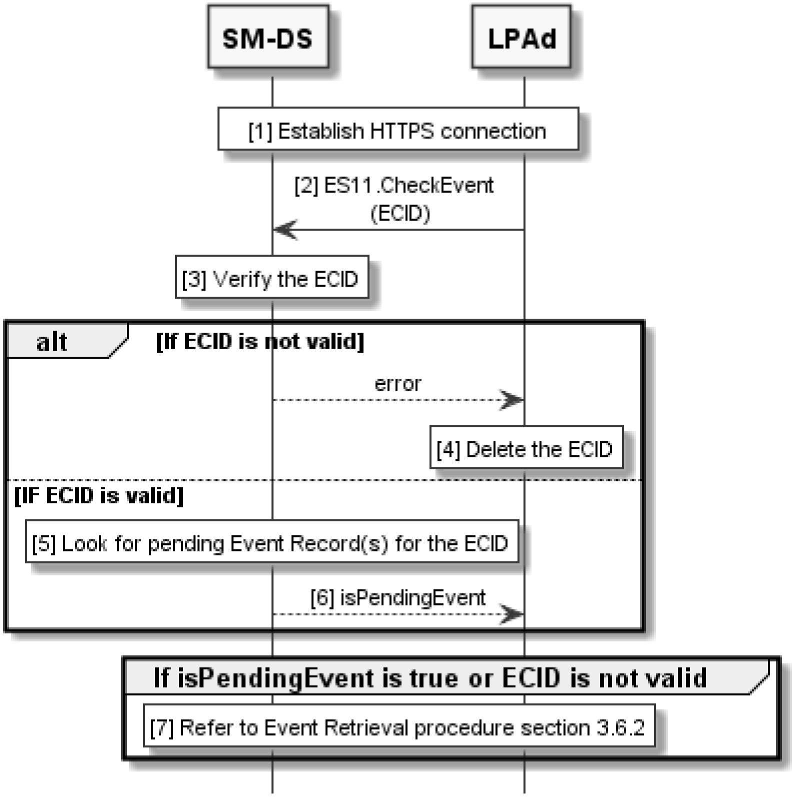
3. As EventID2 is not empty, i.e., the Alternative SM-DS previously cascaded the registration according to 3.6.1.2, the Alternative SM-DS calls "ES12.DeleteEvent" function including the EID and EventID2 which were used in the registration to the Root SM-DS whose address it stored along with the Event Record when the event was registered.
4. The Root SM-DS SHALL verifies that there is a stored Event Record identified by the SM-DS OID and EventID2. If no Event Record is found, the Root SM-DS SHALL return the relevant error status.

4a. Otherwise, the Root SM-DS SHALL delete the Event Record identified by the SM-DS OID and EventID2.

5. The Root SM-DS SHALL return the deletion status.
6. If, at the Root SM-DS, the deletion of the Event Record is successful or the Event Record is not found, the Alternative SM-DS SHALL delete the associated Event Record from its storage.
7. The Alternative SM-DS SHALL return the deletion status.

3.6.4 Event Checking

```
@startuml
hide footbox
skinparam sequenceMessageAlign center
skinparam sequenceArrowFontSize 11
skinparam noteFontSize 11
skinparam monochrome true
skinparam lifelinestrategy solid

participant "<b>SM-DS" as DS
participant "<b>LPAd" as LPA


rnote over DS, LPA #FFFFFF : [1] Establish HTTPS connection

LPA -> DS: [2] ES11.CheckEvent \n (ECID)

rnote over DS #FFFFFF : [3] Verify the ECID
alt If ECID is not valid
DS --> LPA: error
rnote over LPA #FFFFFF : [4] Delete the ECID
else IF ECID is valid
rnote over DS #FFFFFF : [5] Look for pending Event Record(s) for the ECID

DS --> LPA: [6] isPendingEvent
End alt
group If isPendingEvent is true or ECID is not valid
rnote over DS, LPA #FFFFFF : [7] Refer to Event Retrieval procedure section 3.6.2
end
@enduml
```


SM-DS
LPAd
[1] Establish HTTPS connection
[2] ES11.CheckEvent (ECID)
[3] Verify the ECID
alt
[If ECID is not valid]
error
[4] Delete the ECID
[IF ECID is valid]
[5] Look for pending Event Record(s) for the ECID
[6] isPendingEvent
If isPendingEvent is true or ECID is not valid
[7] Refer to Event Retrieval procedure section 3.6.2

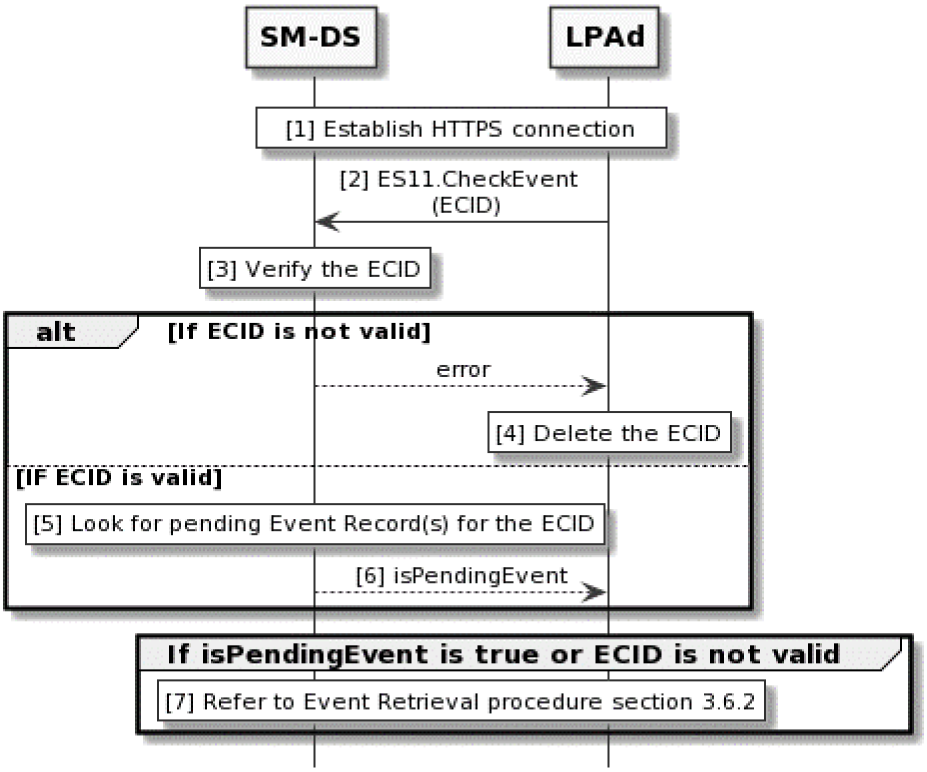

SM-DS
LPAd
[1] Establish HTTPS connection
[2] ES11.CheckEvent (ECID)
[3] Verify the ECID
alt
[If ECID is not valid]
error
[4] Delete the ECID
[IF ECID is valid]
[5] Look for pending Event Record(s) for the ECID
[6] isPendingEvent
If isPendingEvent is true or ECID is not valid
[7] Refer to Event Retrieval procedure section 3.6.2

Figure 29aa: Event Retrieval Procedure

Start Conditions:

The LPAd has determined the set of configured Root SM-DS addresses. It MAY retrieve one or more Root SM-DS addresses configured on the eUICC (using

ES10a.GetEuiccConfiguredData). In addition, it MAY retrieve them from where they are configured on the Device.

As a result of previous Event Retrieval(s), the LPAd received ECID(s) for the corresponding SM-DS(s) and stored them into the Device. If the LPAd cannot determine the ECID for the corresponding SM-DS, the LPAd SHOULD initiate Event Retrieval procedure to the SM-DS as described in section 3.6.2 instead of the Event Checking procedure.

The Event Checking procedure is used to determine the presence of an Event on an SM-DS. This includes, but is not limited to, the following trigger conditions:

- The End User MAY manually query to check the presence of pending Event Records from the configured SM-DS(s). The LUId MAY implement this query in combination with other related operations, for example, as a composite 'Add Profile' operation.
- The LPAd MAY query the configured SM-DS(s) as part of Device power-on Profile discovery as described in section 3.4.4.

Procedure:

1. The LPAd establishes a HTTPS connection with the SM-DS as defined at step (5) of the Common Mutual Authentication procedure in section 3.0.1.
2. The LPAd SHALL call the "ES11.EventCheck" function with the ECID corresponding to the SM-DS.
3. The SM-DS SHALL verify that the received ECID is valid. If the verification fails, the SM-DS SHALL return an error status "ECID – Unknown". Otherwise, the procedure continues at step (5).
4. The LPAd SHALL delete the ECID which was used in step (2) and the procedure continues at step (7).
5. The SM-DS SHALL check if there is a pending Event Record associated with the received ECID. If there is a pending Event Record associated with the ECID, the SM-DS SHALL set `isPendingEvent` to `true`. Otherwise, the SM-DS SHALL set `isPendingEvent` to `false`.
6. The SM-DS SHALL respond to the LPAd with the `isPendingEvent`.
7. The LPAd SHOULD perform the event retrieval procedure as defined in section 3.6.2 if the LPAd has received `isPendingEvent` set to `true` or has received an error status "ECID – Unknown" or "ECID – Expired". Otherwise, the LPAd SHALL terminate the Event Checking procedure.

3.6.5 Push Service Registration

This procedure is used to register to the SM-DS a Push Token for a Push Service, so that the SM-DS can subsequently send push notifications to the LPAd.

```
@startuml

hide footbox
skinparam sequenceMessageAlign center
skinparam sequenceArrowFontSize 11
skinparam noteFontSize 11
skinparam monochrome true
skinparam lifelinestrategy solid

```

```
participant "<b>SM-DS" as DS
participant "<b>LPAd" as LPA
participant "<b>eUICC" as E

rnote over DS, E #FFFFFF : [1] [Refer to Common mutual authentication procedure
section 3.0.1]

LPA -> DS : ES11.AuthenticateClient request \n
(ctxParamsForPushServiceRegistration)

rnote over DS #FFFFFF
[2]
- Verify the pushServiceRegistration
- Store the pushToken and associate the pushToken with the EID
endrnote

group Cond. verification fails
DS --> LPA : error
rnote over LPA, E : Refer to Common Cancel Session procedure section 3.0.2
end

rnote over DS #FFFFFF
[3]
- Build smdsSigned2 = {TransactionID, [pushServiceRefreshTime]}
- Compute smdsSignature2 over smdsSigned2 and euiccSignature1
endrnote

DS --> LPA : [4] TransactionID, smdsSigned2, smdsSignature2

rnote over LPA #FFFFFF : [5] Verify SM-DS response

group Cond. smdsSigned2 or smdsSignature2 is missing
rnote over LPA, E : Refer to Common Cancel Session procedure section 3.0.2
end
group Cond. eUICC supports signed SM-DS response
LPA -> E : [6] ES10a.VerifySmdsResponse
rnote over E #FFFFFF : [7] Verify smdsSignature2
E --> LPA : ok/error
end


rnote over LPA #FFFFFF
[8] Enable the Push Service for corresponding SM-DS
Endrnote

@enduml
```

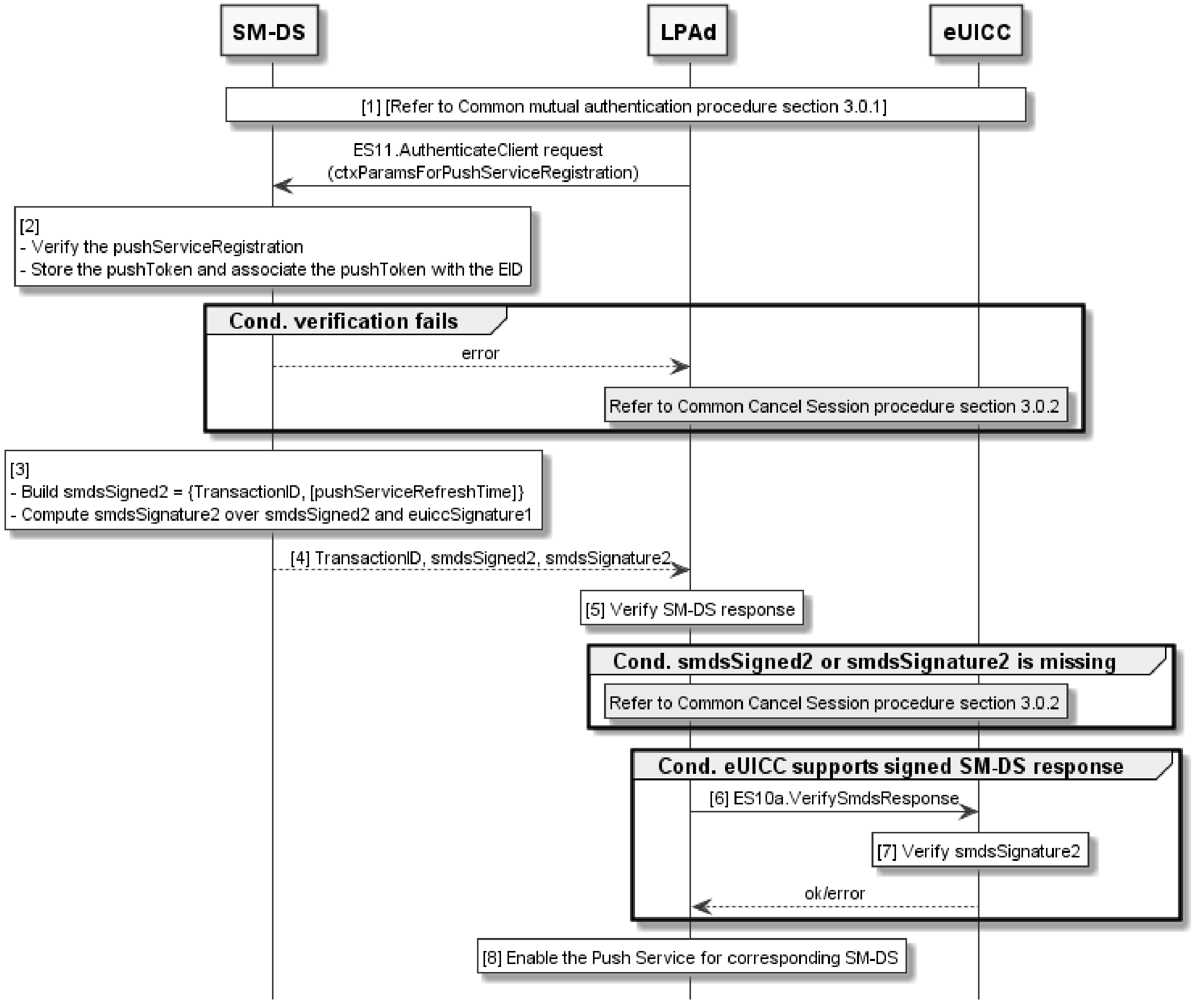

SM-DS
LPAd
eUICC
[1] [Refer to Common mutual authentication procedure section 3.0.1]
ES11.AuthenticateClient request
(ctxParamsForPushServiceRegistration)
[2]
- Verify the pushServiceRegistration
- Store the pushToken and associate the pushToken with the EID
Cond. verification fails
error
Refer to Common Cancel Session procedure section 3.0.2
[3]
- Build smdsSigned2 = {TransactionID, [pushServiceRefreshTime]}
- Compute smdsSignature2 over smdsSigned2 and euiccSignature1
[4] TransactionID, smdsSigned2, smdsSignature2
[5] Verify SM-DS response
Cond. smdsSigned2 or smdsSignature2 is missing
Refer to Common Cancel Session procedure section 3.0.2
Cond. eUICC supports signed SM-DS response
[6] ES10a.VerifySmdsResponse
[7] Verify smdsSignature2
ok/error
[8] Enable the Push Service for corresponding SM-DS

Figure 29ab: Push Service Registration Procedure

Start Conditions:

In addition to the start conditions required by the common mutual authentication procedure defined in section 3.0.1, the LPAd supports a Push Service and the eUICC supports ES10a.VerifySmdsResponse.

The LPAd has retrieved the list of configured Root SM-DS addresses. It MAY retrieve one or more Root SM-DS addresses configured on the eUICC (using ES10a.GetEuiccConfiguredData). In addition, it MAY retrieve them from where they are configured on the Device.

The LPAd SHOULD use the Root SM-DS address(es) configured on a removable eUICC.

Push Service registration MAY be used in following cases:

- To register a new Push Token to the Root SM-DS(s).
- To re-register a Push Token when:

- the LPAd detects the Push Token, which was used for the Push Service, is not valid.
- the time indicated in the `pushServiceRefreshTime` has been reached.

Procedure:

1. The common mutual authentication procedure defined in section 3.0.1 SHALL be executed. When this procedure is used for SM-DS, SM-XX, CERT.XXauth.SIG, PK.XXauth.SIG, SK.XXauth.SIG, and ESXX are SM-DS, CERT.DSauth.SIG, PK.DSauth.SIG, SK.DSauth.SIG, and ES11, respectively.

   During the common mutual authentication procedure at step (10), if both the LPAd and the SM-DS indicated their support of a Push Service, in `lpaRspCapability` and `serverRspCapability` respectively, and the SM-DS provided `supportedPushServices` at step (9), the LPAd SHALL select one Push Service supported by both the LPAd and the SM-DS from the `supportedPushService`.

   NOTE: The LPAd may perform additional operations to retrieve an appropriate Push Token for the selected Push Service, e.g., interaction with the push server, which are out of scope of this specification.

   In addition, the LPAd SHALL build the ctxParams1 data object with `ctxParamsForPushServiceRegistration` comprising the `selectedPushService` and the `pushToken`.

2. After having successfully authenticated the eUICC at the end of the step (1), the SM-DS SHALL verify that the `selectedPushService` indicated in the `pushServiceRegistration` is supported by the SM-DS. If supported, the SM-DS SHALL store the receivied Push Token and associate the Push Token with the EID. Otherwise, an appropriate error status SHALL be returned to the LPAd and the procedure SHALL stop.
3. The SM-DS SHALL generate an `smdsSigned2` data structure containing the `TransactionID` and optionally `pushServiceRefreshTime` and compute the `smdsSignature2` over `smdsSigned2` and `euiccSignature1` using the SK.DSauth.SIG.
4. The SM-DS SHALL respond to the LPAd with `TransactionID`, `smdsSigned2`, and `smdsSignature2`.
5. If `smdsSigned2` or `smdsSignature2` is missing, then the LPAd SHALL send "ES10b.CancelSession" to the eUICC with a reason `sessionAborted`, and the procedure SHALL stop.

If the eUICC indicated `signedSmdsResponseV3Support` during Common Mutual Authentication:

6. The LPAd SHALL call the "ES10a.VerifySmdsResponse" function to verify the SM-DS signature.
7. The eUICC SHALL verify `smdsSignature2` and return success or an error status to the LPAd. If the eUICC returns an error, the procedure SHALL stop.

NOTE: If the eUICC did not indicate `signedSmdsResponseV3Support`, the LPAd MAY verify `smdsSignature2` itself.

Finally:

8. The LPAd enables the selected Push Service for a corresponding SM-DS. If `pushServiceRefreshTime` was received in the response from the SM-DS, the LPA SHALL store the value.

3.7 Remote Profile Management

The Operator initiates Remote Profile Management (RPM) procedure using the ES2+ interface.

RPM is executed by a Managing SM-DP+. A Profile MAY be configured to allow RPM by several Managing SM-DP+s. This specification assumes that a Managing SM-DP+ that prepares an RPM Package has full knowledge of the state of a Profile. How this state is synchronised among multiple Managing SM-DP+s, or procedures where a Managing SM-DP+ does not have full knowledge of the state of a Profile are out of the scope of this specification.

The LPA downloads an RPM Package using the ES9+ interface as described in section 3.7.2. An RPM Package MAY contain one or more RPM Commands.

Each RPM Command SHALL require Confirmation Request enforced by the LPA as described in section 3.7.2. The specific implementation of Confirmation Request by the LPA is out of scope of this specification.

Confirmation Request on multiple RPM Commands for consecutive operations MAY be combined to simplify the user experience and avoid repeated input steps for the End User. For instance, when executing two RPM Commands in an RPM Package, the Strong Confirmation for the first RPM Command and the Simple Confirmation for the second RPM Command MAY be combined. In the case of combined Confirmation Requests, it SHALL be clear to the End User what RPM Commands will be executed, and the highest Confirmation Level SHALL be obtained.

If End User accepts the execution of the RPM Package, the LPA transfers the RPM Package to the eUICC as described in section 3.7.3. The eUICC executes the RPM Command(s) contained in the RPM Package in the received order and generates the Load RPM Package Result specified in section 2.10.2.

Upon completing the execution of an RPM Package, the Operator receives the Load RPM Package Result. Based upon the outcome, the Operator can choose to initiate additional RPM operations. For instance, if a 'Contact PCMP' command resulted in a `noLprConfiguration` error, the Operator could request a new RPM Package containing both an 'Update Metadata' command with `pcmpAddress` and a 'Contact PCMP' command.

3.7.1 RPM Initiation

This procedure is used by Operator to issue an RPM Command(s).

```
@startuml
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
hide footbox

participant "<b>Operator" as OP
participant "<b>SM-DP+" as DP
participant "<b>SM-DS" as DS

rnote over OP : [1] [Generate a MatchingID]
OP -> DP : [2] ES2+.RpmOrder(eid, rpmScript, [MatchingID], [rootSmdsAddress [,
altSmdsAddress]])
rnote left DS
[3] Verify Profile Owner
[4] Prepare RPM Package
[5] [Generate a MatchingID]
endrnote
rnote over DP, DS : [6] [Event Registration]
DP --> OP : [7] MatchingID
@enduml
```

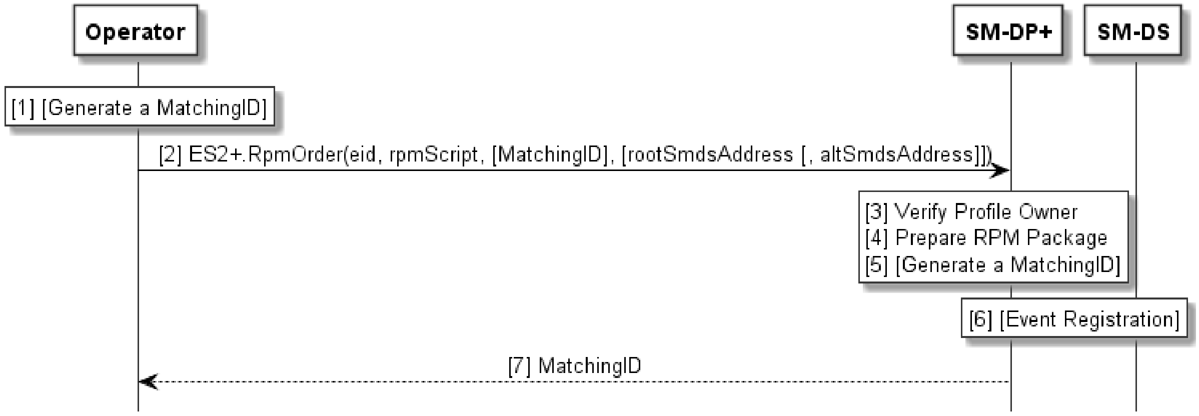

Operator
SM-DP+
SM-DS
[1] [Generate a MatchingID]
[2] ES2+.RpmOrder(eid, rpmScript, [MatchingID], [rootSmdsAddress [, altSmdsAddress]])
[3] Verify Profile Owner
[4] Prepare RPM Package
[5] [Generate a MatchingID]
[6] [Event Registration]
[7] MatchingID

Figure 29a: RPM Initiation

Start Condition:

- The target eUICC has already installed a Profile from the Operator.
- The SM-DP+ is a Managing SM-DP+ of the target Profile.

Procedure:

1. Optionally, the Operator MAY generate a MatchingID.
2. The Operator calls the "ES2+.RpmOrder" (section 5.3.6) function for the SM-DP+ with the relevant input data.

   The Operator SHALL send the rpmScript.

   The Operator MAY send the MatchingID generated in step (1). If a Default SM-DP+ is to be used for the RPM download, then the Operator MAY send an empty string in the MatchingID value field.

   The Operator MAY send the address of a Root SM-DS, and optionally also the address of an Alternative SM-DS, to the SM-DP+ as defined in section 3.6.1. If the Default SM-DP+ is to be used, then the SM-DS address SHALL NOT be present.

   The SM-DP+ SHALL store the functionRequesterIdentifier and functionCallIdentifier values of the "ES2+.RpmOrder" function call, which SHALL be used as notificationReceiverIdentifier and notificationIdentifier, respectively, in subsequent "ES2+.HandleNotification" calls related to this order.
3. If an rpmScript is provided, the SM-DP+ SHALL verify that the Operator is the Profile Owner of all targeted Profiles.

4. The SM-DP+ prepares an RPM Package.
5. If the MatchingID is not provided in step (2), the SM-DP+ SHALL generate a MatchingID in this step. The RPM Package SHALL be associated to the MatchingID.
6. As an optional step, if the SM-DS address is given in step (2), then the SM-DP+ SHALL perform Event Registration to the specified SM-DS.
7. The SM-DP+ returns MatchingID to the Operator. If the Operator provided a MatchingID in step (2), then the returned MatchingID SHALL be the same.

3.7.2 RPM Download

This procedure is used by LPA to download an RPM Package and to get End User consent on RPM Execution.

The procedure applies for SEP and MEP. The following applies for Command Port:

	Command Port
MEP-A1/A2	0
MEP-B	>=0
SEP	Physical interface

All communication between the LPA and the eUICC in the RPM Download procedure SHALL use the Command Port.

```
@startuml
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
hide footbox

participant "<b>Operator" as OP
participant "<b>SM-DP+" as DP
participant "<b>LPAd" as LPA
participant "<b>eUICC" as E

rnote over LPA, E
[1] (a) Get SM-DP+ Address [and CI PKid] from Polling Address, or
     (b) Get SM-DP+ Address and EventID from SM-DS, or
     (c) Get Default SM-DP+ Address [and CI PKID] from eUICC or Device, or
     (d) Reuse SM-DP+ Address [and CI PKID] from the previous RSP session
endrnote

rnote over DP, E : [2] Refer to Common mutual authentication procedure section
3.0.1

rnote over DP
[3] Look for pending RPM Package
endrnote

DP --> LPA : [ERROR]

group Opt.
DP -> OP : [4] ES2+.HandleNotification(...)
OP --> DP : OK
end

rnote over DP
[5]
```

```
- Build smdpSigned3 = {TransactionID, RpmPackage, [rpmPending]}
- Compute smdpSignature3 over smdpSigned3 and euiccSignature1
endrnote

DP -> LPA : [6] TransactionID, smdpSigned3, smdpSignature3

opt RPM disabled, RPM Package contains unsupported command, or invalid RPM Package
rnote over OP, E : Refer to Common Cancel Session procedure section 3.0.2
end

LPA -> E : [7] [ES10c.GetProfilesInfo]
E --> LPA : [ProfileInfoListOk]

rnote over LPA
[8]
- Check RPM Package against PPR(s)
- [User Consent]
- [User Confirmation]
- [Invalidate cached Metadata]
endrnote

alt Download rejection
rnote over OP, E : Refer to Common Cancel Session procedure section 3.0.2
else Download confirmation
rnote over OP, E : Refer to Sub-procedure RPM Execution
end

@enduml
```

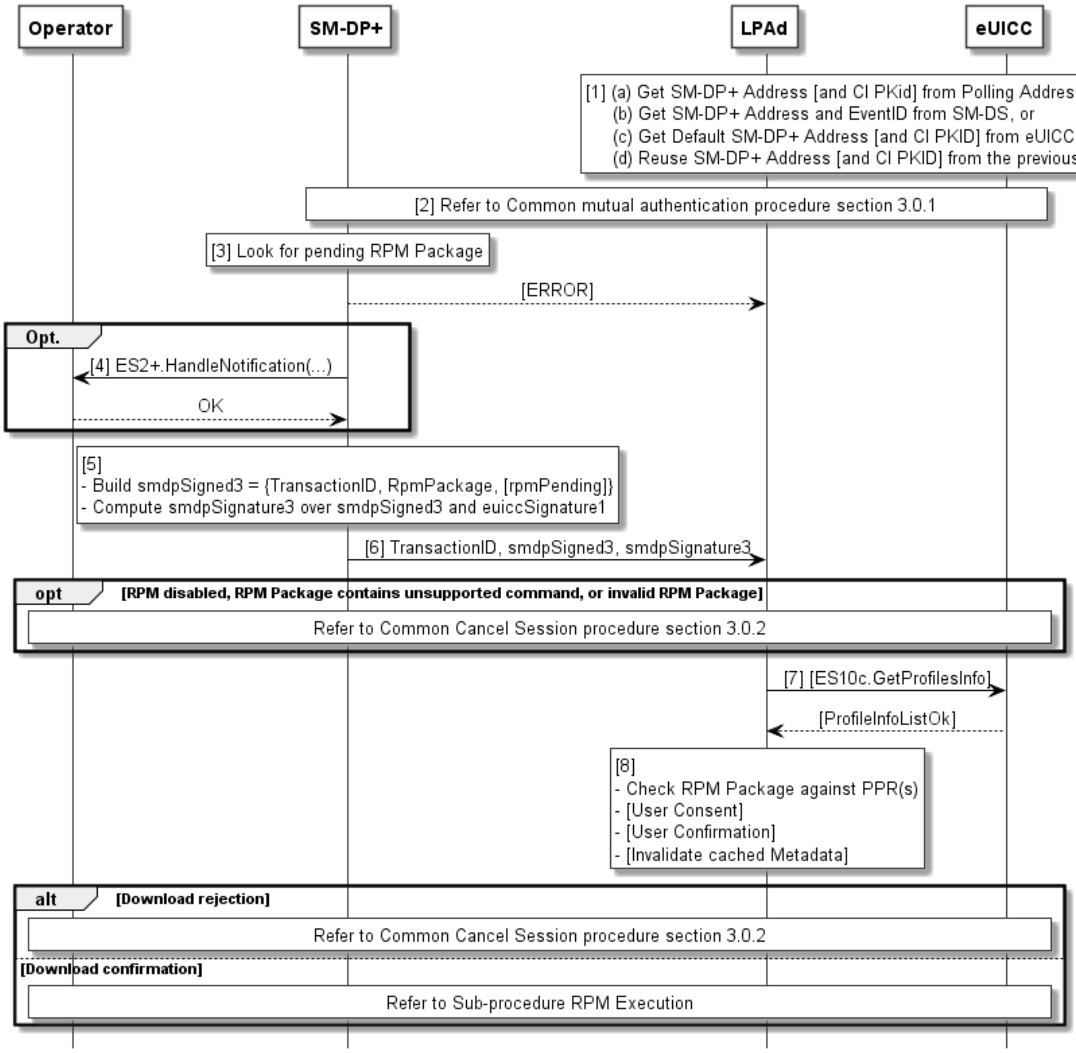

Operator
SM-DP+
LPAd
eUICC
[1] (a) Get SM-DP+ Address [and CI PKid] from Polling Address, or
(b) Get SM-DP+ Address and EventID from SM-DS, or
(c) Get Default SM-DP+ Address [and CI PKID] from eUICC or Device, or
(d) Reuse SM-DP+ Address [and CI PKID] from the previous RSP session
[2] Refer to Common mutual authentication procedure section 3.0.1
[3] Look for pending RPM Package
[ERROR]
Opt.
[4] ES2+.HandleNotification(...)
OK
[5]
- Build smdpSigned3 = {TransactionID, RpmPackage, [rpmPending]}
- Compute smdpSignature3 over smdpSigned3 and euiccSignature1
[6] TransactionID, smdpSigned3, smdpSignature3
opt
[RPM disabled, RPM Package contains unsupported command, or invalid RPM Package]
Refer to Common Cancel Session procedure section 3.0.2
[7] [ES10c.GetProfilesInfo]
[ProfileInfoListOk]
[8]
- Check RPM Package against PPR(s)
- [User Consent]
- [User Confirmation]
- [Invalidate cached Metadata]
alt
[Download rejection]
Refer to Common Cancel Session procedure section 3.0.2
[Download confirmation]
Refer to Sub-procedure RPM Execution

Figure 29b: RPM Download

Start Condition:

In addition to the start conditions required by the common mutual authentication procedure defined in section 3.0.1, this procedure requires the following start conditions depending on options in step (1):

- If this procedure uses a Polling Address (option a):
  - The LPAd has retrieved the SM-DP+ Address and allowed eSIM CA RootCA public key identifier, if any, from the Profile Metadata, if the Polling Address indicates the SM-DP+; or
  - The LPAd has retrieved the SM-DP+ Address and EventID from the SM-DS, if the Polling Address indicates the SM-DS.
  - The LPAd has retrieved the allowed eSIM CA RootCA public key identifier associated with the Polling Address, if any, from the eUICC. If the retrieved data includes an allowed eSIM CA RootCA public key identifier, then the LPAd SHALL restrict the allowed eSIM CA RootCA public key identifiers to that value.
- If this procedure uses an SM-DS (option b):
  - The LPAd has retrieved an SM-DP+ Address and EventID from the SM-DS. If there was a restriction of the eSIM CA RootCA public key identifier for the SM-DS procedure, the LPAd SHALL apply a restriction to the same eSIM CA RootCA public key identifier for the RPM download and installation procedure.
- If this procedure uses a Default SM-DP+ (option c):
  - The LPAd has retrieved the Default SM-DP+ Address and allowed eSIM CA RootCA public key identifier, if any, from the eUICC or the Device. If the retrieved data includes an allowed eSIM CA RootCA public key identifier, then the LPAd SHALL restrict the allowed eSIM CA RootCA public key identifiers to that value.
- If this procedure is triggered by `rpmPending` returned in a previous RSP session (option d):
  - The LPAd SHALL reuse the SM-DP+ Address from the previous RSP session. The LPAd SHALL apply a restriction to the same eSIM CA RootCA public key identifier on the allowed CI public key identifiers.

A Provisioning Profile MAY be enabled by the LPAd upon End User request for RSP operations as defined in SGP.21 [4], which SHALL include End User consent if an Operational Profile is to be disabled and if establishment of the connectivity using the currently Enabled Profile is not successful.

Procedure:

1. The LPAd has retrieved an SM-DP+ address from either SM-DS, eUICC or Polling Address and allowed eSIM CA RootCA Public Key identifier, if any.
2. The common mutual authentication procedure defined in section 3.0.1 SHALL be executed, conditionally restricting the allowed eSIM CA RootCA public key identifiers to a single allowed value as described in the Start Conditions above. In this procedure, SM-XX is SM-DP+. CERT.XXauth.SIG, PK.XXauth.SIG and SK.XXauth.SIG are CERT.DPauth.SIG, PK.DPauth.SIG and SK.DPauth.SIG respectively. ESXX is ES9+.

   During the common mutual authentication procedure at step (10), the LPAd SHALL build the ctxParams1 data object to provide the MatchingID, MatchingID Source, Device Info, operationType and iccid (optional) to the eUICC for signature. The operationType SHALL include 'rpm'. If this function is called in the context of polling RPM for a specific Target Profile, the iccid of the Target Profile SHALL be present; otherwise, it SHALL NOT be present. The value of the MatchingID and MatchingID Source SHALL be set as follows:

   - If a Default SM-DP+ or Polling Address (which indicates an SM-DP+) is used, the MatchingID SHALL be missing and the MatchingID Source value SHALL be set to 'none'.
   - If an SM-DS is used, the MatchingID value SHALL be set to EventID and the MatchingID Source value SHALL be set to the OID of the SM-DS that provided the EventID.
   - If `rpmPending` was received in the previous RSP session, the `matchingId` SHALL be missing and the `matchingIdSource` value SHALL be set to `none`.

3. The SM-DP+ SHALL:

   - Verify that there is a pending RPM Package for the EID of the authenticated eUICC.
   - If there is a pending RPM Package, then verify that the eUICC indicates `rpmSupport` in `EuiccRspCapability`.
   - If MatchingID is provided, then verify that it matches the MatchingID bound to the pending RPM Package.

   If any of these verifications fail, the SM-DP+ SHALL return a relevant error status and the procedure SHALL stop.

4. (Optional step) Depending on the agreed behaviour with the Operator (out of scope of this specification), the SM-DP+ SHALL notify the Operator using the function "ES2+.HandleNotification" with the `notificationEvent` indicating 'Eligibility and attempt limit check.

NOTE: This notification step MAY be done asynchronously.

5. The SM-DP+ SHALL:

   - Determine if another RPM Package for the `eid` is pending.
   - Generate an `smdpSigned3` data structure containing associated data elements.
   - Compute the `smdpSignature3`.

6. The SM-DP+ returns the "ES9+.AuthenticateClient" response to the LPAd. The LPAd SHALL check the following.

- If the RPM operation is disabled in the LPA by the End User, then the LPAd SHALL cancel the RPM download by performing the Common Cancel Session procedure with the reason `rpmDisabled`.
- If the RPM Package violates any of the limitations defined in section 2.10.1, then the LPAd SHOULD cancel the RPM Package download by performing the Common Cancel Session procedure hereunder with the reason `invalidRpmPackage`.
- If the RPM Package contains an RPM Command 'Contact PCMP':
  - If the LPAd does not support the LPRd, then the LPAd SHOULD cancel the RPM download by performing the Common Cancel Session procedure with the reason `lprNotSupported`.
  - If the only available data connection is mobile network (cellular) data, and if the End User has disallowed use of mobile network data for the LPA Proxy, then the LPAd SHALL cancel the RPM download by performing the Common Cancel Session procedure with the reason `lprNetworkDataNotAllowed`.

7. As an optional step, if the LPAd does not already have the Profile Metadata of the Target Profile(s) identified by the ICCID(s) in the RPM Package (if any), then the LPAd SHALL request the information from the eUICC by calling the "ES10c.GetProfilesInfo" function.
8. The LPAd SHALL check the Profile Metadata of the Target Profile(s). If the Profile Metadata of the Target Profile includes any PPR and/or Enterprise Rule, then the LPAd SHOULD ask for the End User consent by showing relevant information concerning the Profile and PPR(s) and/or Enterprise Rule(s). This information MAY include the consequences of the Profile Policy Rule and/or Enterprise Rule to the End User. This message SHALL be formulated in a descriptive and non-discriminatory manner (e.g., for "Non-Disable" Profile Policy Rule: "The RPM Command for Profile X that you are about to allow can be undone only under the terms you have agreed with your service provider. Approve RPM Command YES/NO?").

   The LPAd MAY show relevant information to the End User if the RPM Command results in PPR(s) being unset.

   The LPAd SHALL ask for Confirmation Request(s) on the execution of RPM Commands, either combined or separated, by showing any relevant information in the RPM Package and the Profile Metadata of the Target Profile(s). The Confirmation Request(s) MAY be combined with the End User consent on PPR and/or Enterprise Rule above. The Confirmation Level for each RPM Command SHALL be higher than or equal to the following:

   - No User Confirmation: List Profile Info, Update Metadata (except for the case below), Contact PCMP
   - Simple Confirmation: Enable Profile, Disable Profile, Update Metadata (only if the command sets the Reference Enterprise Rule)
   - Strong Confirmation: Delete Profile

   NOTE: For MEP-A1 and MEP-B, if there are several eSIM Ports available, the LPAd may select an eSIM Port for Profile enabling via RPM. This may include some End User interaction.

If the End User refuses the execution of any RPM Command (e.g., by selecting 'No' or 'Not Now' in combined or separate User Confirmation(s)), the LPAd SHALL continue with the Common Cancel Session procedure with the reason `endUserRejection` or `postponed`.

If the End User does not respond to the LPAd prompt within an implementation-dependent timeout interval, the LPAd SHALL cancel the RPM Package download by performing the Common Cancel Session procedure with the reason `timeout`.

If the RPM Package contains (an) Update Metadata command(s) and the LPAd caches such Profile Metadata objects, the LPAd SHOULD invalidate these objects.

If the execution of the RPM Command(s) has not been rejected in the steps above, the procedure SHALL continue with the sub-procedure "RPM Execution".

3.7.3 RPM Execution

This procedure is used by LPA to transfer RPM Package to eUICC. The eUICC sequentially executes the RPM Command(s) contained in the RPM Package.

The procedure applies for SEP and MEP. The following applies for Command Port and Target Port:

	Command Port	Target Port
MEP-A1/A2	0	>0
MEP-B	As selected in the RPM Download procedure	>=0, identical to or different from Command Port
SEP	Physical interface	Physical interface

The Target Port is selected in the procedure depending on the presence of Disable or Enable commands.

```
@startuml
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
hide footbox

participant "<b>Operator" as OP
participant "<b>SM-DP+" as DP
participant "<b>LPAd" as LPA
participant "<b>eUICC" as E
participant "<b>Device baseband" as DevBB

LPA -> E : [1] CP: ES10b.LoadRpmPackage\n (smdpSigned3, smdpSignature3,
[targetEsimPort])
rnote over E
[2]
- Verify CERT.DPauth.SIG
- Verify smdpSignature3 over smdpSigned3
- Verify smdpSigned3
endrnote
```

```
loop Up to the number of\n transferred RPM Commands
rnote over E
[3] Execute an RPM Command
(see sections 3.7.3.1 to 3.7.3.6)
endrnote
end

rnote over E
[3a] Generate LoadRpmPackageResult
endrnote

E --> LPA : Response APDU\n(LoadRpmPackageResult)

opt Any marked Profile

E -> DevBB: [3b] TP: REFRESH
DevBB -> E: Terminal Response or RESET
rnote over E
[3c] The marked Profile(s) are
     enabled, disabled or deleted
endrnote

rnote over DevBB
[Network attach procedure
with the newly Enabled Profile]
end rnote

end opt
LPA -> DP : [4] ES9+.HandleNotification\n     (LoadRpmPackageResult)
DP --> LPA : OK

rnote over LPA
[Trigger PCM session]
endrnote

rnote over DP
[5] [Terminate RPM order]
endrnote

DP -> OP : [6] [ES2+.HandleNotification]
OP --> DP : OK

rnote over DP
[7] [Delete Event, Refer to Event Deletion section 3.6.3]
endrnote

LPA -> E : [8] CP: ES10b.RemoveNotificationFromList

rnote over E
[9] Delete Notification
endrnote

E --> LPA : OK

rnote over DP, E
[10] [Next RSP session follows]
endrnote

@enduml
```

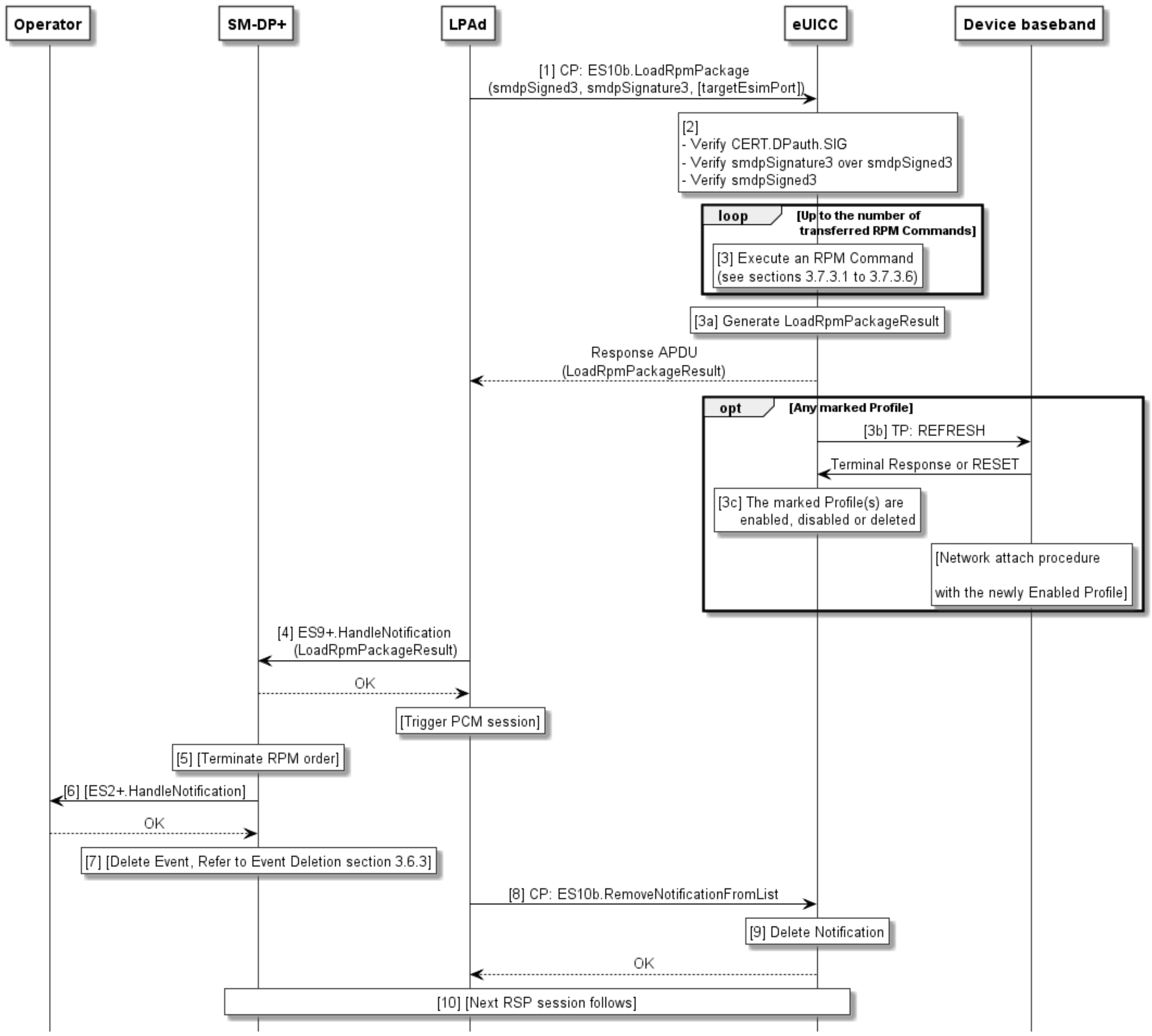

Operator
SM-DP+
LPAd
eUICC
Device baseband
[1] CP: ES10b.LoadRpmPackage
(smdpSigned3, smdpSignature3, [targetEsimPort])
[2]
- Verify CERT.DPauth.SIG
- Verify smdpSignature3 over smdpSigned3
- Verify smdpSigned3
loop
[Up to the number of transferred RPM Commands]
[3] Execute an RPM Command
(see sections 3.7.3.1 to 3.7.3.6)
[3a] Generate LoadRpmPackageResult
Response APDU
(LoadRpmPackageResult)
opt
[Any marked Profile]
[3b] TP: REFRESH
Terminal Response or RESET
[3c] The marked Profile(s) are
enabled, disabled or deleted
[Network attach procedure
with the newly Enabled Profile]
[4] ES9+.HandleNotification
(LoadRpmPackageResult)
OK
[Trigger PCM session]
[5] [Terminate RPM order]
[6] [ES2+.HandleNotification]
OK
[7] [Delete Event, Refer to Event Deletion section 3.6.3]
[8] CP: ES10b.RemoveNotificationFromList
[9] Delete Notification
OK
[10] [Next RSP session follows]

Figure 29c: Sub-procedure RPM Execution

Start Conditions:

The RPM Download procedure was successfully executed.

Procedure:

If any call of "ES10b.LoadRpmPackage" in this sub-procedure returns status words other than '90 00' or '91 XX', the LPAd SHALL perform the Common Cancel Session procedure with reason code 'loadRpmPackageError'.

1. On the Command Port, the LPAd SHALL transfer `smdpSigned3` and `smdpSignature3`, plus for MEP-A1 and MEP-B optionally `targetEsimPort` (the eSIM Port to be used for enabling a Profile via RPM) to the eUICC by calling the "ES10b.LoadRpmPackage" function.

2. The eUICC SHALL:

   - Verify `smdpSignature3`.

- Verify that the `TransactionID` contained in `smdpSigned3` matches the `TransactionID` of the on-going RSP session.

If any of the verifications fails, the eUICC SHALL skip to step 3a and return a relevant error status in the generated result.

3. The eUICC SHALL execute the RPM Commands contained in the RPM Package. Processing of a command MAY involve the Device Baseband. An error within one command MAY terminate processing. No `RpmCommandResult` is generated for the remaining commands.

3a. The eUICC SHALL generate the `LoadRpmPackageResult` data structure comprising the `RpmCommandResult` data structure(s) generated during the processing and execution of the RPM Command(s).

The eUICC SHALL store a signed Load RPM Package Result in its non-volatile memory and return it to the LPAd.

The LPAd MAY inform the End User of the success or error status(es) indicated by the LoadRpmPackageResult.

3b. If any Profile is marked as "to be disabled", "to be disabled and deleted", or "to be enabled", the eUICC SHALL trigger a REFRESH of the Target Port as follows:

- For SEP and MEP-B if the Target Port is identical to the Command Port, the eUICC SHALL send a REFRESH proactive command on the Target Port.
- For MEP-A1, MEP-A2 and MEP-B if the Target Port is not identical to the Command Port, the eUICC SHALL send an LSI COMMAND proactive command with the action "Proactive session request" on the Command Port. This results in the Device checking for pending proactive commands on the Target Port, whereupon the eUICC sends a REFRESH proactive command on the Target Port.

NOTE: Enabling or Disabling via RPM always uses REFRESH. A mode which does not require a REFRESH is only defined for Local Profile Management.

3c Upon reception of the Terminal Response or after the RESET, the ISD-R SHALL:

- If a Profile is marked "to be disabled": disable the marked Profile.
- If a Profile is marked "to be disabled and deleted": disable the marked Profile and then delete it.
- If a Profile is marked "to be enabled": enable the marked Profile.
- If any marked Profile is successfully enabled, disabled or deleted in the previous steps, generate as many Notifications as configured in each Profile Metadata (`notificationConfigurationInfo`) in the format of `OtherSignedNotification`.
- Unmark all marked Profiles.

If a Profile is now in Enabled state, the Device baseband executes the network attach procedure on the Target Port.

4. The LPAd calls the "ES9+.HandleNotification" function in order to deliver the Load RPM Package Result to the SM-DP+. The SM-DP+ SHALL acknowledge the reception of the Notification to the LPAd. If the Load RPM Package Result was not signed by the eUICC, the procedure SHALL stop.

   If the Device and the eUICC support the LPA Proxy and if appropriate connectivity is available:

   - If the Load RPM Package Result includes the FQDN of PCMP server, the LPRd SHALL trigger a Profile Content Management session, with the DPI in the RPM Command (if present), as described in section 3.9.

   - If the Load RPM Package Result includes a successful `EnableProfileResponse` and `lprConfiguration.triggerLprOnEnableProfile` is present in the Profile Metadata of the enabled Profile, the LPAd SHALL trigger a Profile Content Management session using the DPI in the Profile Metadata (if present) as described in section 3.9.

5. The SM-DP+ SHALL:

   - Retrieve the RPM order identified by the TransactionID. If TransactionID is unknown, the SM-DP+ SHALL terminate its processing.
   - (Conditional) Terminate the RPM order.

6. (Conditional) The SM-DP+ SHALL call the "ES2+.HandleNotification" with:

   - `notificationReceiverIdentifier` reflecting the functionRequesterIdentifier value of the associated "ES2+.RpmOrder";
   - `notificationIdentifier` reflecting the functionCallIdentifier value of the associated "ES2+.RpmOrder";
   - `notificationEvent` indicating 'RPM execution';
   - `notificationEventStatus` reflecting the value received in "ES9+.HandleNotification";
   - `resultData` containing the result of the execution of the RPM Package.

   NOTE: The content of `resultData` is generated by the eUICC while processing the ES10b.LoadRpmPackage command, and before the Device Baseband processes a potential REFRESH proactive command. This implies that `resultData`, and accordingly, `notificationEventStatus`, cannot reflect any error that may occur afterwards, such as, network attachment failure.

7. (Conditional) If this procedure is executed in the context of option (b), the SM-DP+ SHALL execute the SM-DS event deletion procedure (section 3.6.3).

8. On the Command Port, the LPAd SHALL call "ES10b.RemoveNotificationFromList" with the corresponding seqNumber. For MEP-B, this function MAY use a different Command Port.

9. The eUICC SHALL delete the Load RPM Package Result from its non-volatile memory.

10. (Conditional) If the LPAd has received `rpmPending` in the response of "ES9+.AuthenticateClient" function call, the LPAd SHOULD initiate an additional RSP Session with the SM-DP+, setting the `operationType` to indicate `rpm`. If this RSP session was triggered by an Event Record from an SM-DS, the pending RSP session with the SM-DP+ SHOULD be executed before continuing processing any remaining Event Records from that SM-DS. For MEP-B, a subsequent RPM session MAY use a different Command Port.

3.7.3.1 Enable Profile

This procedure is used to remotely enable a Profile already downloaded and installed on an eUICC.

```
@startuml
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
hide footbox

participant "<b>eUICC\n<b>LPA Services (ISD-R)" as LPAServices

rnote over LPAServices
[1] Find the Target Profile
[2] Verify authorisation of the SM-DP+
[3] Verify Profiles type[4] Verify Profile state[5] Enforce Profile Policy Rules
and Enterprise Rules[6] Verify preceding Disable or available Port
[6a] [Set Target Port]end rnoternote over LPAServices
[7] Mark Target Profile "to be enabled"
endrnote

rnote over LPAServices
[8] Generate an RpmCommandResult
     data structure
endrnote

rnote over LPAServices
[9] [If required:
     Stop processing of the RPM Package]
endrnote

@enduml
```

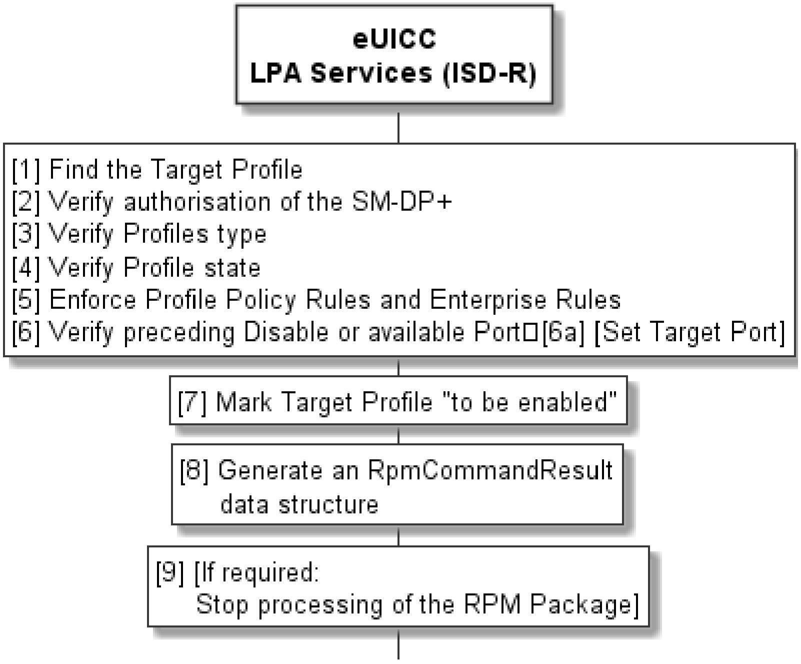

eUICC
LPA Services (ISD-R)
[1] Find the Target Profile
[2] Verify authorisation of the SM-DP+
[3] Verify Profiles type
[4] Verify Profile state
[5] Enforce Profile Policy Rules and Enterprise Rules
[6] Verify preceding Disable or available Port□[6a] [Set Target Port]
[7] Mark Target Profile "to be enabled"
[8] Generate an RpmCommandResult data structure
[9] [If required: Stop processing of the RPM Package]

Figure 29d: Enable Profile

Start Conditions:

In addition to the start conditions described in section 3.2.1 "Enable Profile", an RPM Command 'Enable Profile' is received within an "ES10b.LoadRpmPackage".

Procedure:

1. The ISD-R SHALL find the Target Profile with the ICCID. If the Target Profile is not found, the ISD-R SHALL proceed to step (8) with a result indicating a failure.
2. The ISD-R SHALL verify the authorisation of the SM-DP+ for the RPM Command. If the verification fails, then the ISD-R SHALL proceed to step (8) with a result indicating a failure.
3. The ISD-R SHALL verify the type of the Target Profile and the currently Enabled Profile (if any). If at least one of them is a Test Profile, the ISD-R SHALL proceed to step (8) with a result indicating a failure.
4. The ISD-R SHALL verify the state of the Target Profile. If the Target Profile is not in Disabled state, the ISD-R SHALL proceed to step (8) with a result indicating a failure.
5. The ISD-R SHALL check the Profile Policy Rules of the currently Enabled Profile (if any) and the Reference Enterprise Rule (if any). If the target Profile cannot be enabled, the ISD-R SHALL proceed to step (8) with a result indicating a failure.
6. For SEP, the ISD-R SHALL verify that either no Profile is currently Enabled or the currently Enabled Profile is marked "to be disabled" or "to be disabled and deleted". Otherwise, the ISD-R SHALL proceed to step (8) with a result indicating a failure.

6a. For MEP, if the Target Port was not already set in a previous RPM Disable command:

- The ISD-R SHALL verify that at least one eSIM Port is available (i.e., has no Enabled Profile assigned). Otherwise, the ISD-R SHALL proceed to step (8) with a result indicating a failure.

- For MEP-B, if the LPAd did not indicate an eSIM Port to be used in the "ES10b.LoadRpmPackage" and if the Command Port is available, the eUICC SHALL select the Command Port as Target Port.
- For MEP-A1 and MEP-B, if the LPAd indicated an eSIM Port to be used in the "ES10b.LoadRpmPackage" and if the indicated eSIM Port is available, the eUICC SHALL select this eSIM Port as Target Port.
- For other cases for MEP-A1 and MEP-B, the behaviour of the eUICC is implementation specific. (E.g., no eSIM Port or a non-available eSIM Port indicated for MEP-A1 and the eUICC selecting the eSIM Port or the eUICC generating an error.) However, it SHALL NOT result in the implicit disabling of an Enabled Profile.
- For MEP-A2, the eUICC SHALL select an available eSIM Port as Target Port.

7. The eUICC SHALL mark the Target Profile "to be enabled".
8. The eUICC SHALL generate an `RpmCommandResult` data structure indicating the result of the RPM Command 'Enable Profile'.
9. If the execution of this RPM Command fails and `continueOnFailure` is not present, the ISD-R SHALL stop the execution of the remaining RPM Command(s).

End Conditions:

The Target Profile is marked "to be enabled" and, for MEP, the Target Port is defined. An `RpmCommandResult` data structure containing the result of Enable Profile is stored in the eUICC.

3.7.3.2 Disable Profile

This procedure is used to remotely disable an Enabled Profile already downloaded and installed on an eUICC.

```
@startuml
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
hide footbox

participant "<b>eUICC\n<b>LPA Services (ISD-R)" as LPAServices

rnote over LPAServices[1] Find the Target Profile
[2] Verify authorisation of the SM-DP+
[3] Verify Profile type and state
[4] Enforce Profile Policy Rulesend rnoternote over LPAServices[5] Mark Target
Profile "to be disabled"
[5a] [Set Target Port]
end rnote

rnote over LPAServices
[6] Generate an RpmCommandResult
     data structure
endrnote

rnote over LPAServices
[7] [If required:
Stop processing of the RPM Package]
```

```
endrnote

@enduml
```

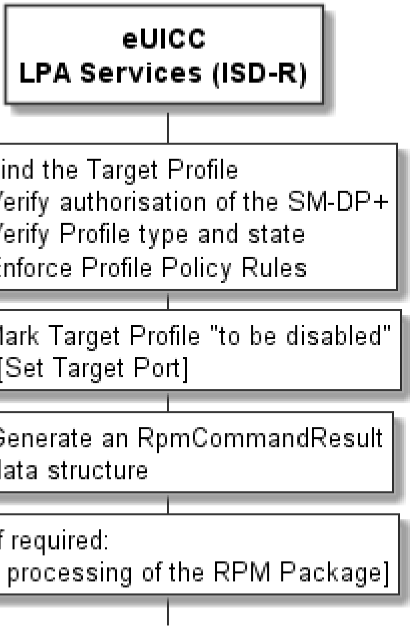

eUICC
LPA Services (ISD-R)
[1] Find the Target Profile
[2] Verify authorisation of the SM-DP+
[3] Verify Profile type and state
[4] Enforce Profile Policy Rules
[5] Mark Target Profile "to be disabled"
[5a] [Set Target Port]
[6] Generate an RpmCommandResult data structure
[7] [If required:
Stop processing of the RPM Package]

Figure 29e: Disable Profile

Start Conditions:

In addition to the start conditions described in section 3.2.2 "Disable Profile", an RPM Command 'Disable Profile' is received over "ES10b.LoadRpmPackage".

Procedure:

1. The ISD-R SHALL find the Target Profile with the ICCID. If the Target Profile is not found, the ISD-R SHALL proceed to step (6) with a result indicating a failure.
2. The ISD-R SHALL verify the authorization of the SM-DP+ for the RPM Command. If the verification fails, then the ISD-R SHALL proceed to step (6) with a result indicating a failure.
3. The ISD-R SHALL verify the type and the state of the Target Profile. If the Target Profile is a Test Profile or if it is not in the Enabled state, the ISD-R SHALL proceed to step (6) with a result indicating a failure.
4. The ISD-R SHALL check the Profile Policy Rules of the currently Enabled Profile. If it has a Profile Policy Rule "Disabling not allowed", the ISD-R SHALL proceed to step (6) with a result indicating a failure.

NOTE: The eUICC may check in addition if the eSIM Port of the Target Profile matches the eSIM Port indicated in the "ES10b.LoadRpmPackage". The response to a mismatch is implementation specific. It may silently be ignored or result in an error.

5. The eUICC SHALL mark the Target Profile "to be disabled" and, for MEP, set the Target Port to the eSIM Port where the Target Profile is currently Enabled.
6. The eUICC SHALL generate an `RpmCommandResult` data structure indicating the result of the RPM Command 'Disable Profile'.

7. If the execution of this RPM Command fails and `continueOnFailure` is not present, the ISD-R SHALL stop the execution of the remaining RPM Command(s).

End Conditions:

The Target Profile is marked "to be disabled" and, for MEP, the Target Port is defined. An `RpmCommandResult` data structure containing the result of Disable Profile is stored in the eUICC.

3.7.3.3 Delete Profile

This procedure is used to remotely delete a Profile already downloaded and installed on an eUICC.

The conditions under which the RPM Command MAY delete a Provisioning Profile are implementation-dependent and out of the scope of this specification. The eUICC implementation MAY not support deletion of a Provisioning Profile or a preloaded Test Profile.

```
@startuml
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
hide footbox

participant "<b>eUICC\n<b>LPA Services (ISD-R)" as LPAServices

rnote over LPAServices[1] Find the target Profile
[2] Verify authorisation of the SM-DP+
[3] Enforce Profile Policy Rulesend rnotealt If the Profile is in Disabled state
and\n is not marked "to be enabled"
rnote over LPAServices
[4a] Delete the target Profile
endrnote
else If the Profile is marked "to be disabled"
rnote over LPAServices
[4b] Mark the target Profile "to be disabled and deleted"
endrnote
else Otherwise
rnote over LPAServices
[4c] Failure
endrnote

end

rnote over LPAServices
[5] Generate an RpmCommandResult
     data structure
endrnote

rnote over LPAServices
[6] [If required:
Stop processing of the RPM Package]
endrnote

@enduml
```

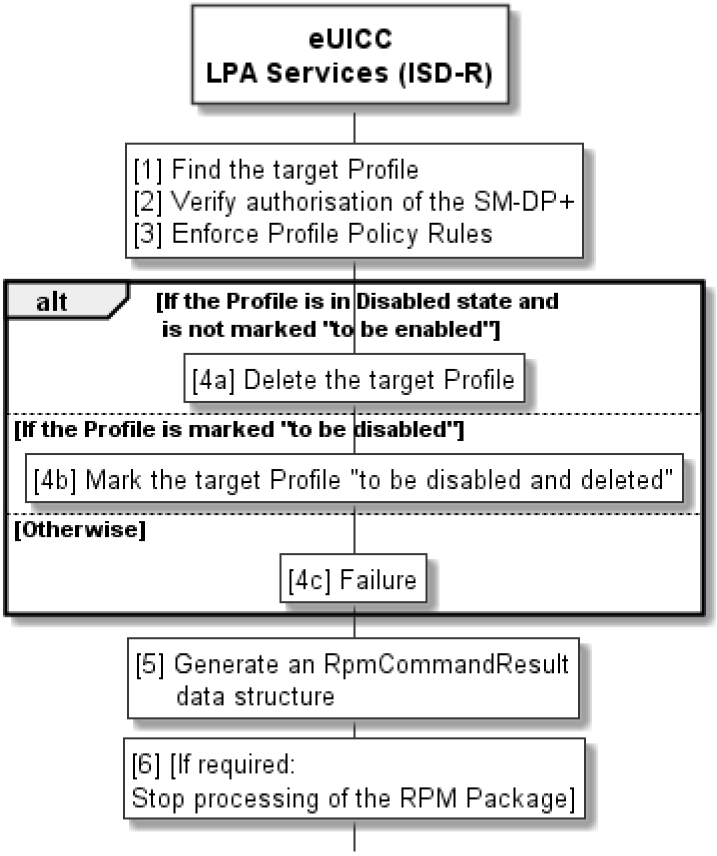

eUICC
LPA Services (ISD-R)
[1] Find the target Profile
[2] Verify authorisation of the SM-DP+
[3] Enforce Profile Policy Rules
alt
[If the Profile is in Disabled state and is not marked "to be enabled"]
[4a] Delete the target Profile
[If the Profile is marked "to be disabled"]
[4b] Mark the target Profile "to be disabled and deleted"
[Otherwise]
[4c] Failure
[5] Generate an RpmCommandResult data structure
[6] [If required: Stop processing of the RPM Package]

Figure 29f: Delete Profile

Start Conditions:

In addition to the start conditions described in section 3.2.3 "Delete Profile", an RPM Command 'Delete Profile' is received over "ES10b.LoadRpmPackage".

Procedure:

1. The ISD-R SHALL find the target Profile with the ICCID. If the target Profile is not found, the ISD-R SHALL proceed to step (5) with a result indicating a failure.
2. The ISD-R SHALL verify the authorisation of the SM-DP+ for the RPM Command. If the verificationfails then the ISD-R SHALL proceed to step (5) with a result indicating a failure.
3. The ISD-R SHALL check the Profile Policy Rules of the target Profile. If it does not allow deletion, the ISD-R SHALL proceed to step (5) with a result indicating a failure.

4a. If the target Profile is in Disabled state and not marked "to be enabled": The eUICC SHALL delete the Profile. If the target Profile is successfully deleted, the eUICC SHALL generate as many Notifications as configured in its Profile Metadata (`notificationConfigurationInfo`) in the format of `OtherSignedNotification`.

4b. If the Profile is in Enabled state and marked "to be disabled": The eUICC SHALL re-mark the target Profile "to be disabled and deleted".

4c. If the target Profile is in Enabled state and is not marked "to be disabled", or the target Profile is in Disabled state and is marked "to be enabled", the ISD-R SHALL proceed to step (5) with a result indicating a failure.

5. The eUICC SHALL generate an `RpmCommandResult` data structure indicating the result of the RPM Command 'Delete Profile'.
6. If the execution of this RPM Command fails and `continueOnFailure` is not present, the ISD-R SHALL stop the execution of the remaining RPM Command(s).

End Conditions:

The target Profile is deleted. An `RpmCommandResult` data structure containing the result of Delete Profile is stored in the eUICC.

3.7.3.4 List Profile Info

This procedure is used to list information about Profiles on an eUICC.

```
@startuml
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
hide footbox

participant "<b>eUICC\n<b>LPA Services (ISD-R)" as LPAServices

rnote over LPAServices
[1] Find the target Profile(s)
[2] Verify authorisation of the SM-DP+
endrnote
rnote over LPAServices
[3] Generate an RpmCommandResult     with the ProfileInfo of the target Profile(s)
endrnote

rnote over LPAServices
[4] [If required:
Stop processing of the RPM Package]
endrnote

@enduml
```

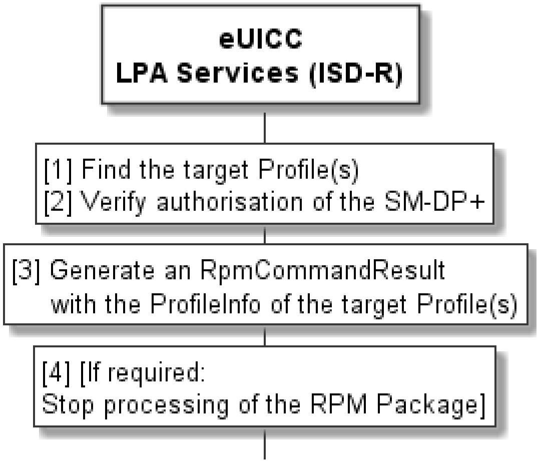

eUICC
LPA Services (ISD-R)
[1] Find the target Profile(s)
[2] Verify authorisation of the SM-DP+
[3] Generate an RpmCommandResult
with the ProfileInfo of the target Profile(s)
[4] [If required:
Stop processing of the RPM Package]

Figure 29g: List Profiles Info

Start Conditions:

An RPM Command 'List Profile Info' is received over ES10b.LoadRpmPackage.

Procedure:

1. The ISD-R SHALL find the target Profile(s) with the ICCID or the Profile Owner OID. If no matching Profile is found, the ISD-R SHALL generate an empty response, and proceed to step (3).
2. For all identified Profiles, the ISD-R SHALL verify the authorisation of the SM-DP+ for the RPM Command. If the verification fails, the ISD-R SHALL remove this Profile from the list of identified Profiles.
3. The ISD-R SHALL generate an RpmCommandResult data structure for the found Profile(s).
4. If the execution of this RPM Command fails and `continueOnFailure` is not present, the ISD-R SHALL stop the execution of the remaining RPM Command(s).

End Conditions:

An `RpmCommandResult` data structure containing the `ProfileInfo` data object(s) that a Managing SM-DP+ is authorised to receive.

3.7.3.5 Update Metadata

This procedure is used to remotely update the Profile Metadata of a Profile already downloaded and installed on an eUICC.

```
@startuml
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
hide footbox

participant "<b>eUICC\n<b>LPA Services (ISD-R)" as LPAServices

rnote over LPAServices
[1] Find the target Profile
[2] Verify authorisation of the SM-DP+
[3] [Verify Enterprise Configuration]
rnote over LPAServices
[4] Update the Profile Metadata
    of the target Profile
[5] [Remove referenceEnterpriseRule]
endrnote

rnote over LPAServices
[6] Generate an RpmCommandResult
     data structure
endrnote

rnote over LPAServices
[7] [If required:
Stop processing of the RPM Package]
endrnote

@enduml
```

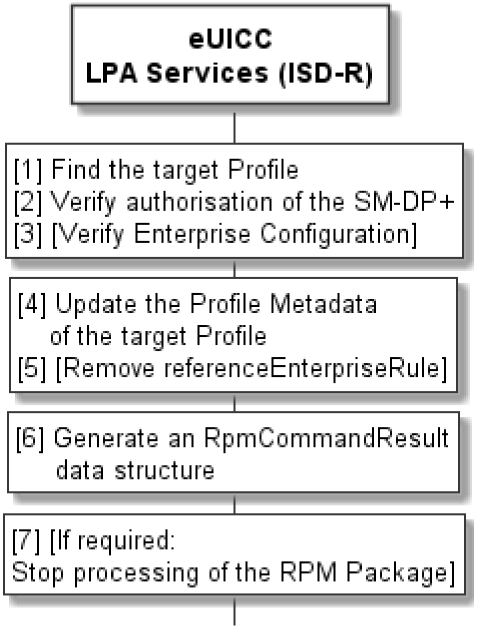

eUICC
LPA Services (ISD-R)
[1] Find the target Profile
[2] Verify authorisation of the SM-DP+
[3] [Verify Enterprise Configuration]
[4] Update the Profile Metadata of the target Profile
[5] [Remove referenceEnterpriseRule]
[6] Generate an RpmCommandResult data structure
[7] [If required: Stop processing of the RPM Package]

Figure 29h: Update Metadata

Start Conditions:

An RPM Command 'Update Metadata' is received over "ES10b.LoadRpmPackage".

Procedure:

1. The ISD-R SHALL find the target Profile with the ICCID. If the target Profile is not found, the ISD-R SHALL proceed to step (6) with a result indicating a failure.
2. The ISD-R SHALL verify the authorisation of the SM-DP+ for the RPM Command.If the verification fails, then the ISD-R SHALL proceed to step (6) with a result indicating a failure.
3. If an Enterprise Configuration is to be updated: The ISD-R SHALL verify that this update is permitted as defined in section 5.4.1. If the verification fails, the ISD-R SHALL proceed to step (6) with a result indicating a failure.
4. The ISD-R SHALL update the Profile Metadata of the target Profile by using the UpdateMetadataRequest specified in the RPM Command.
5. If the UpdateMetadataRequest contains an Enterprise Rule with the referenceEnterpriseRule bit set: the ISD-R SHALL unset the referenceEnterpriseRule bit of the Enterprise Profile for which it is currently set, if any.
6. The eUICC SHALL generate an `RpmCommandResult` data structure indicating the result of the RPM Command 'Update Metadata'.
7. If execution of this RPM Command fails and `continueOnFailure` is not present, the ISD-R SHALL stop the execution of the remaining RPM Command(s).

End Conditions:

The Profile Metadata of the target Profile is updated, and optionally the target Profile is enabled. An `RpmCommandResult` data structure containing the result of RPM Command 'Update Metadata' is stored in the eUICC.

3.7.3.6 Contact PCMP

The support of the RPM Command 'Contact PCMP' is optional for the eUICC.

This procedure is used to fetch the PCMP address configured in the enabled Profile.

```
@startuml
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
hide footbox

participant "<b>eUICC\n<b>LPA Services (ISD-R)" as LPAServices

rnote over LPAServices
[0] Verify eUICC support of LPA Proxy
[1] Verify ICCID
[2] Verify authorisation of the SM-DP+
[3] Verify Profile state
[4] Fetch PCMP Address
[5] Generate an RpmCommandResult data structure
endrnote
@enduml
```

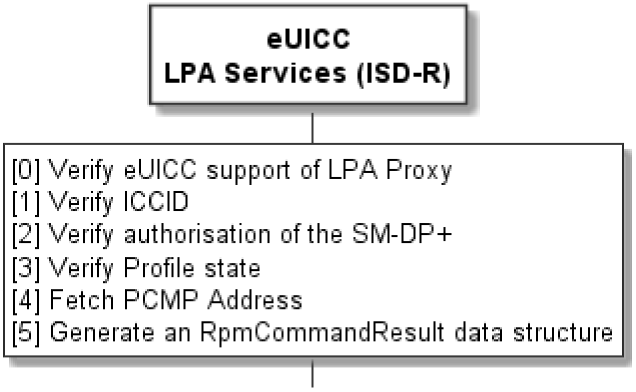

eUICC
LPA Services (ISD-R)
[0] Verify eUICC support of LPA Proxy
[1] Verify ICCID
[2] Verify authorisation of the SM-DP+
[3] Verify Profile state
[4] Fetch PCMP Address
[5] Generate an RpmCommandResult data structure

Figure 29i: Contact PCMP

Start Conditions:

An RPM Command 'Contact PCMP' is received over "ES10b.LoadRpmPackage".

Procedure:

0. If the eUICC does not support the RPM Command 'Contact PCMP', the ISD-R SHALL proceed to step (5) with a result indicating `unknownOrDamagedCommand`.
1. If the target Profile cannot be identified by the ICCID, the ISD-R SHALL proceed to step (5) with a result indicating `commandError`.
2. The ISD-R SHALL verify the following:

   - The SM-DP+ OID specified in the RPM Command is included in the `managingDpList` and is authorised to perform the RPM Command. Otherwise, the ISD-R SHALL proceed to step (5) with a result indicating `commandError`.
   - If the Profile Metadata specifies an allowed CI public key identifier: the Subject Key Identifier of the CI corresponding to CERT.DPauth.SIG matches that value.

Otherwise, the ISD-R SHALL proceed to step (5) with a result indicating `commandError`.

3. If the target Profile is not currently enabled, the ISD-R SHALL proceed to step (4) with a result indicating `profileNotEnabled`.
4. The ISD-R SHALL fetch the LPR Configuration of the target Profile to include the PCMP Address in the `RpmCommandResult`. If the target Profile does not contain an LPR Configuration, the ISD-R SHALL proceed to step (5) with a result indicating `noLprConfiguration`.
5. The eUICC SHALL generate an `RpmCommandResult` data structure indicating the result of the RPM Command 'Contact PCMP'.

End Conditions:

An `RpmCommandResult` data structure containing the result of Contact PCMP is stored in the eUICC.

NOTE: See section 3.7.3 for triggering of a PCM session.

3.8 Remote Management by the Operator

3.8.1 Metadata Update via ES6

This procedure is used by the Profile Owner to update Metadata of an enabled Profile.

```
@startuml
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
hide footbox

participant "<b>Operator" as OP
participant "<b>Device" as Device
participant "<b>eUICC" as eUICC

OP -> eUICC: [1] ES6.UpdateMetadata(...)
rnote over eUICC
[2] Verify command and update Metadata
endrnote
Opt Metadata update alerting supported
eUICC -> Device: [3] REFRESH (tag list)
rnote over Device, eUICC : [4] [Invalidate or update\n     cached Profile Metadata]
end opt

@enduml
```

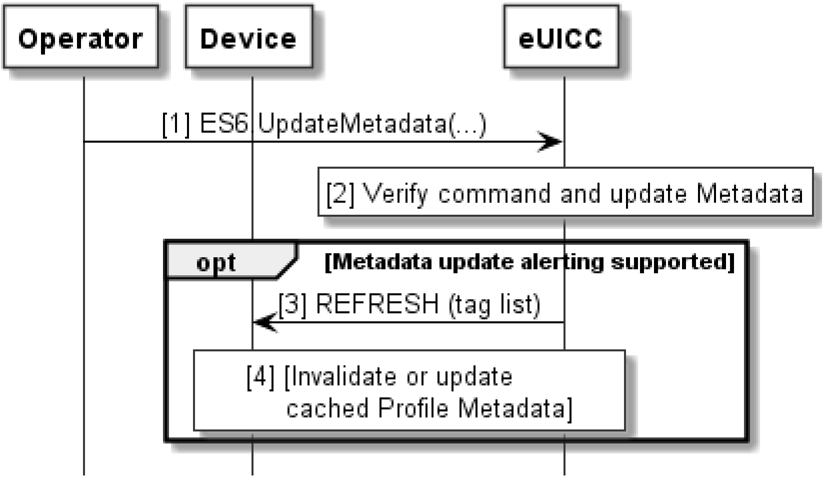

Operator
Device
eUICC
[1] ES6 UpdateMetadata(...)
[2] Verify command and update Metadata
opt
[Metadata update alerting supported]
[3] REFRESH (tag list)
[4] [Invalidate or update
cached Profile Metadata]

Figure 29j: Metadata Update

Start Condition:

The Profile owned by the Operator is enabled.

Procedure:

1. The Operator sends an UpdateMetadata via ES6 to the enabled Profile.
2. The eUICC verifies the command and updates the Metadata as defined in section 5.4.1.

Steps 3 and 4 apply for a Device that supports Metadata update alerting:

3. The eUICC alerts the Device as follows:

- For SEP and MEP-B, the eUICC sends a REFRESH proactive command on the eSIM Port of the Profile with mode "Application Update" and providing the tag list of all Metadata objects updated or deleted by the UpdateMetadata.
- For MEP-A1 and MEP-A2, the eUICC first sends an LSI COMMAND proactive command with the action "Proactive session request" for eSIM Port 0 on the eSIM Port of the Profile. This results in the Device checking for pending proactive commands on eSIM Port 0, whereupon the eUICC sends a REFRESH proactive command on eSIM Port 0 with mode "Application Update", providing the ICCID of the Profile and the tag list of all Metadata objects updated or deleted by the UpdateMetadata.
- The internal processing of the "Application Update" REFRESH command in the Device, especially how the LPA is alerted, is out of scope.

4. For Profile Metadata objects that are cached in the LPA, the Device invalidates or updates these objects.

NOTE: A Device not supporting Metadata update alerting SHOULD NOT cache Profile Metadata to avoid using outdated information after Metadata updates via ES6.

3.8.2 Pending Operation Alerting

This procedure is used by the Profile Owner to alert the LPA of pending RSP operations.

```
@startuml
```

```
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
hide footbox

participant "<b>SM-DS\n<b>SM-DP+" as SM
participant "<b>Operator" as OP
participant "<b>Device" as Device
participant "<b>LPA" as LPA
participant "<b>eUICC" as eUICC
participant "<b>Profile" as P

OP -> P: [1] Application specific message

P -> Device: [2] REFRESH (server to check)
alt Default SM-DP+
rnote over SM, eUICC #FFFFFF
[3]
- Retrieve Default SM-DP+ address
- Profile Download and Installation Procedure
  or RPM Download and Execution Procedure
end rnote
else Root SM-DS
rnote over SM, eUICC #FFFFFF
[4] For all configured Root SM-DSs:
- Event Retrieval Procedure
- Process Events
end rnote
else Polling Address
rnote over SM, eUICC #FFFFFF
[5]
- Retrieve Polling Address
- [Event Retrieval procedure]
- Profile Download and Installation Procedure
  or RPM Download and Execution Procedure
end rnote
else explicit address
rnote over SM, eUICC #FFFFFF
[6]
- [Event Retrieval procedure]
- Profile Download and Installation Procedure
  or RPM Download and Execution Procedure
end rnote
end

@enduml
```

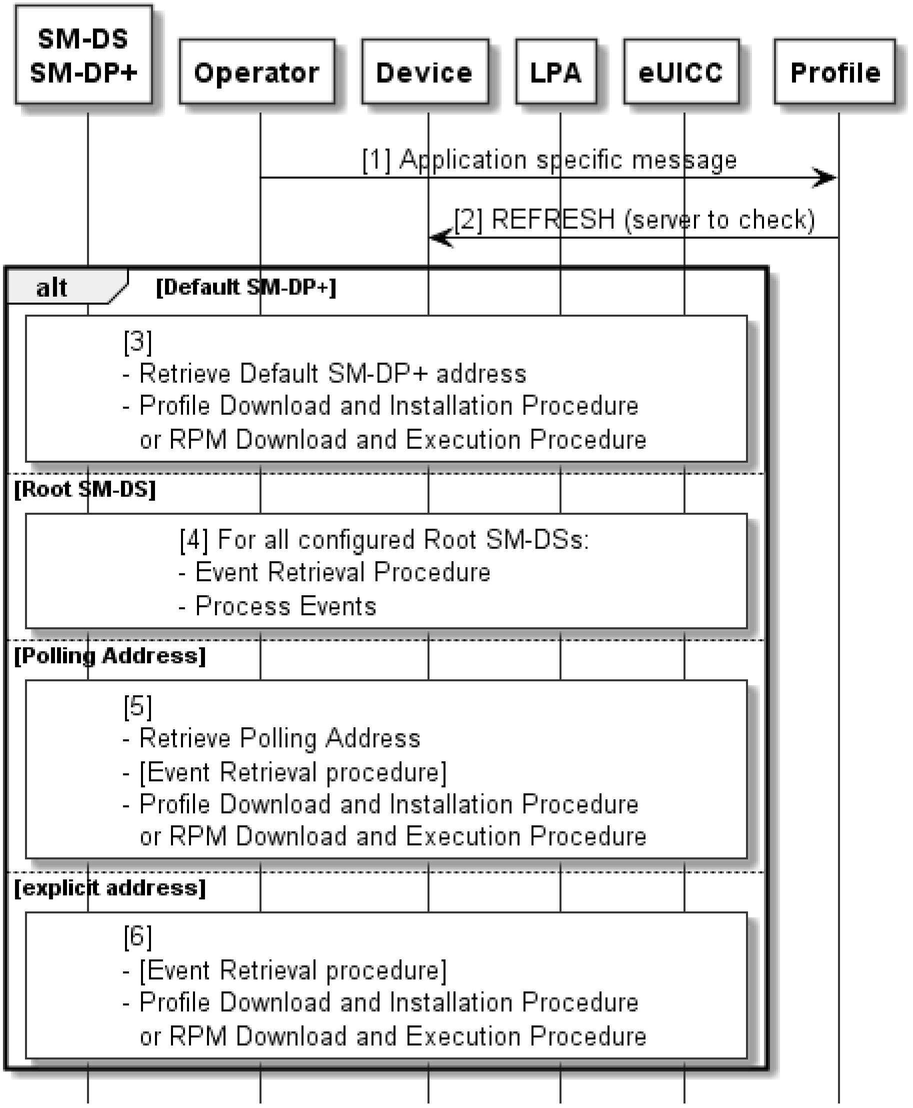

SM-DS
SM-DP+
Operator
Device
LPA
eUICC
Profile
[1] Application specific message
[2] REFRESH (server to check)
alt
[Default SM-DP+]
[3]
- Retrieve Default SM-DP+ address
- Profile Download and Installation Procedure
or RPM Download and Execution Procedure
[Root SM-DS]
[4] For all configured Root SM-DSs:
- Event Retrieval Procedure
- Process Events
[Polling Address]
[5]
- Retrieve Polling Address
- [Event Retrieval procedure]
- Profile Download and Installation Procedure
or RPM Download and Execution Procedure
[explicit address]
[6]
- [Event Retrieval procedure]
- Profile Download and Installation Procedure
or RPM Download and Execution Procedure

Figure 29k: Pending Operation Alerting

Start Condition:

The Profile owned by the Operator is enabled.

The Device supports pending operation alerting.

NOTE: An SM-DP+ can know from `LpaRspCapability.pendingOperationAlertingSupport` whether this feature is supported on the Device and can use this feature.

Procedure:

1. The Operator sends an application specific message to an application in the enabled Profile.

NOTE: For remote management, ETSI TS 102 226 [39] specifies a command 'Immediate Action' that can be used to trigger a proactive REFRESH command. However, the support of this command by the eUICC is not mandatory and error responses are not specified.

2. The application triggers the LPA by sending a REFRESH proactive command on the eSIM Port of the enabled Profile with mode "Application Update" to the Device, providing the indication of the server to be checked for pending operations. (The internal processing of this command in the Device, especially how the alert is forwarded to the LPA, is out of scope). When connecting to a Server as described below, the operation type SHALL indicate all possible operations.
3. If the command indicates the Default SM-DP+, the LPAd SHOULD retrieve the Default SM-DP+ address(es) from the eUICC and the Device to download Profile(s) or RPM package(s) from the SM-DP+ as defined in section 3.1.3 or 3.7.2.
4. If the command indicates the Root SM-DS, the LPAd SHOULD retrieve pending events from all configured SM-DSs using the Event Retrieval Procedure defined in section 3.6.2 and process the events accordingly.
5. If the command indicates the Polling Address of the Profile, the LPAd SHOULD retrieve the Polling Address of the Enabled Profile from the eUICC. If the Polling Address is an SM-DS, the LPAd performs the Event Retrieval procedure as described in section 3.6.2 and processes the events accordingly. If the Polling Address is an SM-DP+, the LPAd downloads a Profile or an RPM package from the SM-DP+ as defined in section 3.1.3 or 3.7.2.
6. If the command indicates an explicit address: if the address is an SM-DS, the LPAd performs the Event Retrieval procedure as described in section 3.6.2 and processes the events accordingly. If the address is an SM-DP+, the LPAd downloads a Profile or an RPM package from the SM-DP+ as defined in section 3.1.3 or 3.7.2.

3.9 Profile Content Management

This procedure applies when a PCM session is triggered.

```
@startuml
hide footbox
skinparam sequenceMessageAlign center
skinparam sequenceArrowFontSize 11
skinparam noteFontSize 11
skinparam monochrome true
skinparam lifelinestrategy solid
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF

autonumber "[0]"

participant "<b>Device\n<b>App" as App
participant "<b>LPRd" as LPR
participant "<b>PCMAA" as AA
participant "<b>PCMP" as PCMP
participant "<b>Profile" as P

App -> LPR : [ES21.InitiateProfileContentManagement]

rnote over LPR
     PCM trigger received
     Calculate initial URI
end rnote

LPR -> AA: Start PCM (initial URI)

loop until done
    AA -> PCMP: HTTP request
    AA -> LPR : notify(HTTP request)
    LPR -> App : [ES21.PcmProgressInformation(HTTP request)]

    PCMP --> AA: HTTP response
    AA -> LPR : notify(HTTP response)
    LPR -> App : [ES21.PcmProgressInformation(HTTP response)]

    loop for each part of the multipart response
        alt progress message
            AA -> LPR : notify(progress message)
            LPR -> App : [ES21.PcmProgressInformation(progress message)]
        else script part
            loop for each APDU in the script part
                AA -> P : APDU header and data
                P --> AA : SW1,SW2 and response data
            end loop
            AA -> LPR : notify(script part delivery)
            LPR -> App: [ES21.PcmProgressInformation(script part delivery)]
        end alt
    end loop

end loop

AA -> LPR: notify(end of PCM session)
LPR -> App: [ES21.PcmProgressInformation(end of PCM session)]

@enduml
```

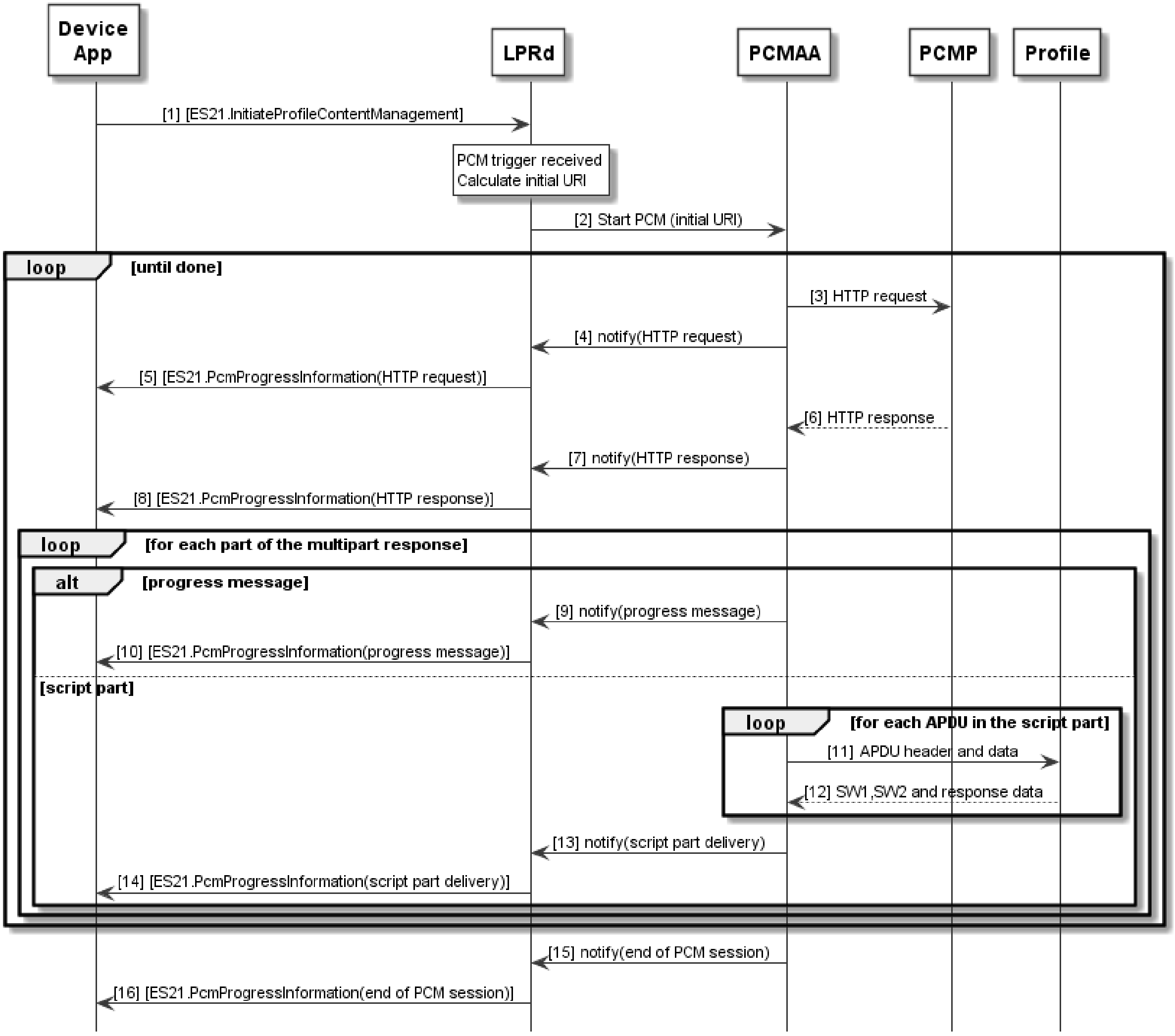

Device App
LPRd
PCMAA
PCMP
Profile
[1] [ES21.InitiateProfileContentManagement]
PCM trigger received
Calculate initial URI
[2] Start PCM (initial URI)
loop
[until done]
[3] HTTP request
[4] notify(HTTP request)
[5] [ES21.PcmProgressInformation(HTTP request)]
[6] HTTP response
[7] notify(HTTP response)
[8] [ES21.PcmProgressInformation(HTTP response)]
loop
[for each part of the multipart response]
alt
[progress message]
[9] notify(progress message)
[10] [ES21.PcmProgressInformation(progress message)]
[script part]
loop
[for each APDU in the script part]
[11] APDU header and data
[12] SW1,SW2 and response data
[13] notify(script part delivery)
[14] [ES21.PcmProgressInformation(script part delivery)]
[15] notify(end of PCM session)
[16] [ES21.PcmProgressInformation(end of PCM session)]

Figure 29l: Profile Content Management

This section is normative with respect to the LPRd and – if applicable – its interaction with a Device Application. The behaviours of the PCMAA, PCMP, and Enabled Profile within a

Profile Content Management session are specified in GP SERAM [74]; the description of their behaviours in this specification is informative.

Start Conditions:

Three methods may trigger PCM:

- A Profile is enabled which has PCM triggering enabled. The optional DPI is configured in the Profile Metadata.
- An RPM command sent from a SM-DP+ to the LPRd, which MAY contain a DPI.
- An API command sent from a Device Application to the LPRd, which MAY contain a DPI.

The LPRd SHALL calculate the initial URI by concatenating the PCMP address it retrieved from the Metadata of the enabled Profile and the DPI if provided.

Additionally, at least one of the following conditions SHALL be satisfied:

- The End User has not disallowed mobile network data (cellular data) to be used for the LPA Proxy/PCM session, and mobile network data is available; or
- Some other data connectivity is available (e.g., WiFi).

Procedure:

1. A Device Application MAY initiate the PCM session by invoking ES21.InitiateProfileContentManagement. The Device Application MAY request the LPRd to send it progress notifications during the PCM session. The PCM session MAY also be triggered by a Profile enabling or a 'Contact PCMP' RPM command. Upon receipt of a trigger, the LPRd SHALL calculate the initial URI.
2. The LPRd SHALL request the PCMAA to start a new PCM session, passing it the initial URI.

   The PCMAA repeatedly sends HTTP requests to the PCMP and processes the HTTP responses until all work within the PCM session is completed. For each iteration:

3. The PCMAA sends an HTTP request to the PCMP. Dependant on information in the previous HTTP response, this request may go to a different server.
4. The PCMAA sends a notification to the LPRd of the HTTP request it sent to the PCMP.
5. If a Device Application has requested progress notifications, the LPRd SHALL send an ES21.PcmProgressInformation notification to the Device Application, indicating an HTTP request along with the targeted URI.
6. The PCMP sends an HTTP response to the PCMAA.
7. The PCMAA sends a notification to the LPRd that it has received the HTTP response from the PCMP.
8. If a Device Application has requested progress notifications, the LPRd SHALL send an ES21.PcmProgressInformation notification to the Device Application, indicating an HTTP response was received from the PCMP along with the number of script parts it contains.

The HTTP response from the PCMAA may contain multiple HTTP parts, each of which may either contain a progress message or a script part. For each HTTP part:

If the HTTP part contains a progress message:

9. The PCMAA sends a notification to the LPRd containing the progress message.
10. If a Device Application has requested progress notifications, the LPRd SHALL send an ES21.PcmProgressInformation notification to the Device Application containing the progress message.

Otherwise (the HTTP part contains a script part):

For each APDU within the script part:

11. The PCMAA sends the APDU to the Enabled Profile.
12. The Enabled Profile returns status words and response data.

After all APDUs in the script part have been sent:

13. The PCMAA sends a notification to the LPRd containing the result of the script part.
14. If a Device Application has requested progress notifications, the LPRd SHALL send an ES21.PcmProgressInformation notification to the Device Application containing the script part number, the delivery status of the script part, and the status words of all APDUs that were processed by the Enabled Profile.

Upon completion of the PCM session:

15. The PCMAA send a notification to the LPRd that the PCM session has ended.
16. If a Device Application has requested progress notifications, the LPRd SHALL send an ES21.PcmProgressInformation notification to the Device Application indicating the termination status of the session.

If the PCM session was triggered via RPM, the LPA SHALL start any subsequent RSP session only after the PCM session ended.

3.10 eUICC Root Public Key Update

The Device and eUICC MAY provide a mechanism to update the set of eSIM CA RootCA Public Keys in the ECASD of the eUICC. The mechanism, where provided, is Device Manufacturer/EUM-specific and SHALL be secure.

A Certificate installed into the ECASD SHALL be from an eSIM CA.

This LPAd MAY start this procedure when either:

- The LPAd has determined that the eSIM CA RootCA Public Key indicator in an Activation Code is not supported by the eUICC, or
- During the Common Mutual Authentication procedure, the SM-XX has returned a CERT.XXauth.SIG that chains to a Root Certificate that is not supported by the eUICC.

If supported, the LPAd SHOULD initiate the secure update mechanism (section 2.4.2) with the eUICC indicating the requested eSIM CA RootCA Public Key indicator or identifier, which MAY include communication with an external EUM-specific server. This MAY provision additional related eSIM CA RootCA Public Keys (e.g., one for each curve supported by the requested eSIM CA). As part of this process, the Device SHALL inform the End User of the proposed update and SHALL obtain End User consent.

Upon successful completion of the credential update, the LPAd MAY retry the Profile Download procedure again.

3.11 Device Change and Profile Recovery

3.11.1 Device Change

This procedure will allow the End User having a Profile in the old Device to add a Profile related to the same Subscription in the new Device. This procedure can further enable the downloaded Profile upon End User consent, which consequently MAY disable a currently Enabled Profile of the new Device (if any). Network connectivity is assumed.

```
@startuml
hide footbox
skinparam sequenceMessageAlign center
skinparam sequenceArrowFontSize 12
skinparam noteFontSize 12
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
skinparam boxPadding 6
hide footbox

participant "End User" as EU
box "Old Device"
participant "LPAd" as LPA1
participant "eUICC\nLPA Services (ISD-R)" as EUICC1
end box
participant "SM-DP+" as DP
box "New Device"
participant "LPAd" as LPA2
participant "eUICC\nLPA Services (ISD-R)" as EUICC2
end box
participant "Service Provider" as SP

rnote over EU, LPA1 : [1] End User interactions

LPA1 -> EUICC1 : [2] ES10c.GetProfilesInfo
EUICC1 --> LPA1 : Profile Metadata
rnote over LPA1 : Check Device Change Configuration

LPA1 -> LPA1 : [3] Identify the SM-DP+ Address
opt If new Device's information is required
LPA1 <- LPA2 : [4] Retrieval from new device ([EID], [TAC])
end
rnote over LPA1, DP : [5] Common Mutual Authentication Procedure, see 3.0.1
LPA1 -> DP : ES9+.AuthenticateClient request (ctxParamsForDeviceChange)
Opt If ES2+.HandleDeviceChangeRequest is configured by the Service Provider
DP -> SP : [6] [ES2+.HandleDeviceChangeRequest (ICCID, [EID], [TAC])]
SP --> DP : [OK (isNewProfileRequired, [Service Provider Message for Device
Change], [Confirmation Code])]
end
alt Device Change is not supported or not allowed
DP --> LPA1 : [7] Error
else Otherwise - Device Change is allowed
Opt If ES2+.HandleNotification is configured by the Service Provider
DP -> SP : [8] [ES2+.HandleNotification (Device Change Request)]
```

```
SP --> DP : [OK]
end
DP --> LPA1 : [9] ES9+.AuthenticateClient response \n (transactionId, smdpSigned4,
smdpSignature4, [Service Provider Message for Device Change])

opt If isNewProfileRequired = TRUE or depending on a service agreement
rnote over DP, SP : [10] Prepare Profile Download section 3.1.1\n (this process can
be performed in parallel to steps 6 to 19)
end

rnote over EU, LPA1 : [11] \n - [Display Service Provider Message for Device
Change] \n - Confirmation Request\n - [Confirmation Code entry]

LPA1 -> EUICC1 : [12] ES10b.PrepareDeviceChange \n (smdpSigned4, smdpSignature4,
Hashed Confirmation Code)
EUICC1 --> LPA1 : ES10b.PrepareDeviceChange response
LPA1 -> DP : [13] ES9+.ConfirmDeviceChange request \n (transactionId,
prepareDeviceChangeResponse)
DP -> SP : [14] [ES2+.HandleNotification (Device Change Confirmation or
Confirmation Failure)]
SP --> DP : [OK]
alt If End User confirmed
opt If isNewProfileRequired=FALSE or if configured by the Service Provider
rnote over DP : [15] Prepare the Profile \n identified by ICCID
end
Opt If ES2+.HandleNotification is configured by the Service Provider
DP -> SP : [ES2+.HandleNotification (Profile preparation for Device Change)]
SP --> DP : [OK]
end
end
DP --> LPA1 : [16] ES9+.ConfirmDeviceChange response
rnote over LPA1, EUICC1 : [Disable the target Profile]
LPA1 -> EUICC1 : ES10b.VerifyDeviceChange (deviceChangeConfirmation)
EUICC1 --> LPA1 : [OK]
end
opt If the installed Profile has to be deleted
rnote over LPA1, EUICC1 : [17] Delete the installed Profile
LPA1 -> DP : [18] [ES9+.HandleNotification (Delete Notification)]
opt If the SM-DP+ supports the Delete Notification for Device Change and \n the
Delete Notification was not transmitted
LPA1 -> LPA1 : Generate an Activation Code containing the Delete Notification for
Device Change
end
end

LPA1 -> LPA2 : [19] Provide the Activation Code (e.g., via LUI)
rnote over EUICC2, DP : [20] Profile Download and Installation Procedure, see 3.1.3
@enduml
```

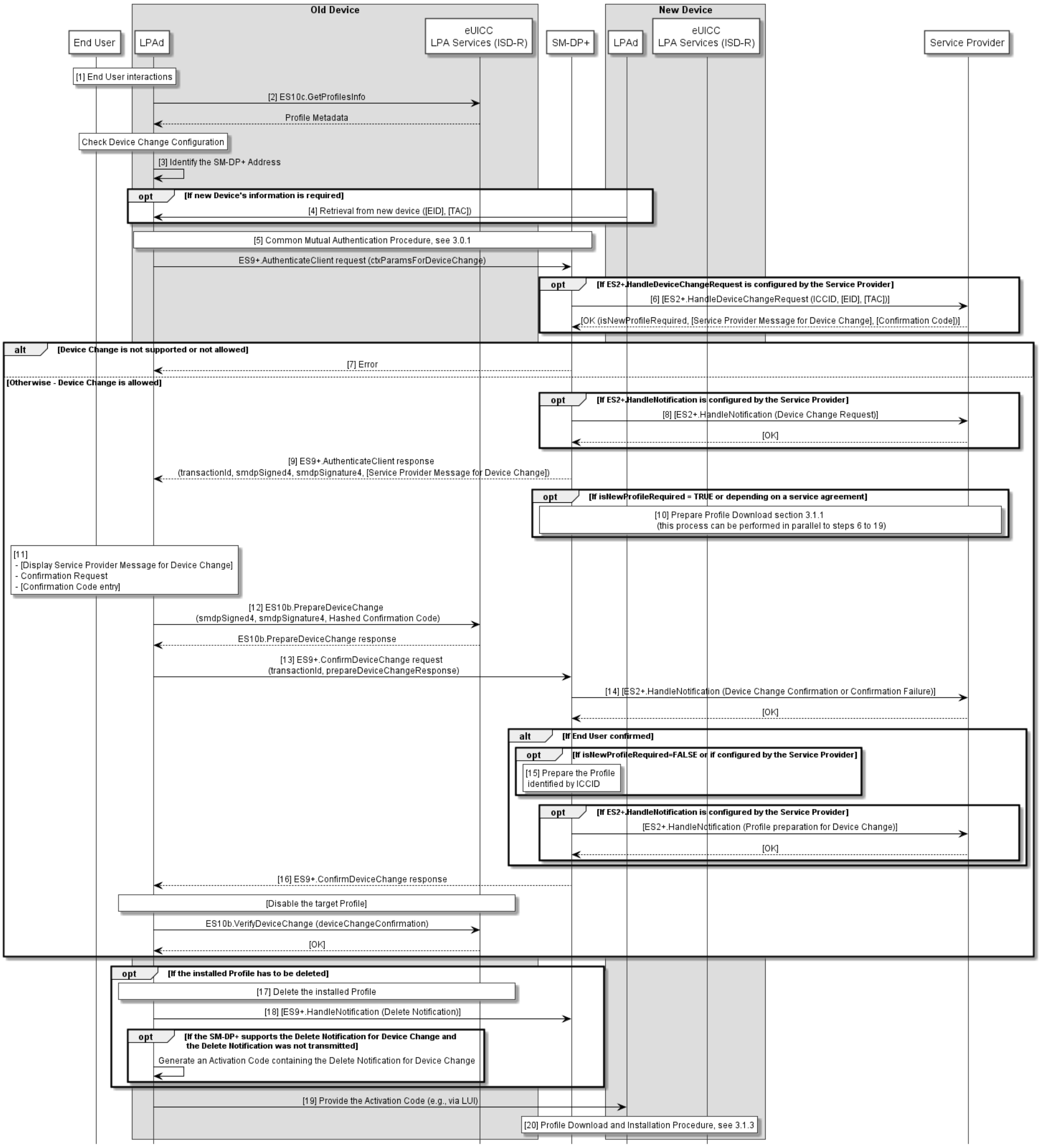

Old Device
New Device
End User
LPAd
eUICC
LPA Services (ISD-R)
SM-DP+
LPAd
eUICC
LPA Services (ISD-R)
Service Provider
[1] End User interactions
[2] ES10c.GetProfilesInfo
Profile Metadata
Check Device Change Configuration
[3] Identify the SM-DP+ Address
opt
[If new Device's information is required]
[4] Retrieval from new device ([EID], [TAC])
[5] Common Mutual Authentication Procedure, see 3.0.1
ES9+.AuthenticateClient request (ctxParamsForDeviceChange)
opt
[If ES2+.HandleDeviceChangeRequest is configured by the Service Provider]
[6] [ES2+.HandleDeviceChangeRequest (ICCID, [EID], [TAC])]
[OK (isNewProfileRequired, [Service Provider Message for Device Change], [Confirmation Code])]
alt
[Device Change is not supported or not allowed]
[7] Error
[Otherwise - Device Change is allowed]
opt
[If ES2+.HandleNotification is configured by the Service Provider]
[8] [ES2+.HandleNotification (Device Change Request)]
[OK]
[9] ES9+.AuthenticateClient response
(transactionId, smdpSigned4, smdpSignature4, [Service Provider Message for Device Change])
opt
[If isNewProfileRequired = TRUE or depending on a service agreement]
[10] Prepare Profile Download section 3.1.1
(this process can be performed in parallel to steps 6 to 19)
[11]
- [Display Service Provider Message for Device Change]
- Confirmation Request
- [Confirmation Code entry]
[12] ES10b.PrepareDeviceChange
(smdpSigned4, smdpSignature4, Hashed Confirmation Code)
ES10b.PrepareDeviceChange response
[13] ES9+.ConfirmDeviceChange request
(transactionId, prepareDeviceChangeResponse)
[14] [ES2+.HandleNotification (Device Change Confirmation or Confirmation Failure)]
[OK]
alt
[If End User confirmed]
opt
[If isNewProfileRequired=FALSE or if configured by the Service Provider]
[15] Prepare the Profile
identified by ICCID
opt
[If ES2+.HandleNotification is configured by the Service Provider]
[ES2+.HandleNotification (Profile preparation for Device Change)]
[OK]
[16] ES9+.ConfirmDeviceChange response
[Disable the target Profile]
ES10b.VerifyDeviceChange (deviceChangeConfirmation)
[OK]
opt
[If the installed Profile has to be deleted]
[17] Delete the installed Profile
[18] [ES9+.HandleNotification (Delete Notification)]
opt
[If the SM-DP+ supports the Delete Notification for Device Change and
the Delete Notification was not transmitted]
Generate an Activation Code containing the Delete Notification for Device Change
[19] Provide the Activation Code (e.g., via LUI)
[20] Profile Download and Installation Procedure, see 3.1.3

Figure 29m: Device Change

Start Conditions:

- The Service Provider has provided to the SM-DP+ the relevant information and configuration for the Device Change (see Annex O).
- The End User has an old Device containing a Profile.
- The eUICC and the LPAd of the old Device support Device Change.
- The Profile on the old Device contains a Device Change Configuration with the information as provided by the Service Provider.
- None of the Profile Policy Rules is set for the Profile with a Device Change Configuration.
- The End User gets a new Device.

- User Intent is acquired as defined in SGP.21 [4] in the old Device.

Procedure:

1. The End User initiates the Device Change operation from the LPAd of the old Device and selects the Profile to be installed in their new Device.
2. The LPAd of the old Device retrieves the `DeviceChangeConfiguration` from the Profile Metadata of the selected Profile. The LPAd of the old Device SHALL check the retrieved `DeviceChangeConfiguration` and proceed based upon its value as follows:

   - If the `DeviceChangeConfiguration` indicates `requestToDp`, the procedure continues with step (3).

   - If the `DeviceChangeConfiguration` indicates `usingStoredAc`, the procedure continues with step (17).

3. The LPAd of the old Device SHALL determine the SM-DP+ address from `smdpAddressForDc` in `DeviceChangeConfiguration` of the Profile.
4. If the `DeviceChangeConfiguration` indicates any of the EID and/or TAC of the new Device is required, the LPAd of the old Device SHALL retrieve the required information from the new Device. The detailed interface and mechanism to retrieve the required information is out of scope of the specification.

   NOTE: For instance, the LPAd of the old Device can guide the End User to use "Show EID" menu of the LPAd of the new Device, and then provide a means to scan/input the EID in a QR code format or human-readable text format.

   If the `DeviceChangeConfiguration` indicates that any of the EID and/or TAC of the new Device is required but the LPAd of the old Device cannot retrieve the required information, the LPAd SHALL display an appropriate error state to the End User and stop the procedure.

5. The LPAd of the old Device initiates the Common Mutual Authentication procedure defined in section 3.0.1 to the retrieved SM-DP+ address. During the Common Mutual Authentication procedure, if the `DeviceChangeConfiguration` includes an allowed eSIM CA RootCA public key identifier, the LPAd SHALL restrict the allowed eSIM CA RootCA public key identifiers to that value.
   During the Common Mutual Authentication procedure at step (10), the LPAd SHALL build the `ctxParams1` data object with `ctxParamsForDeviceChange` comprising the ICCID of the selected Profile and, if indicated as required in `DeviceChangeConfiguration`, any of the EID and/or TAC of the new Device.

6. If configured by the Service Provider, the SM-DP+ SHALL call ES2+.HandleDeviceChangeRequest function comprising the ICCID and, if present in the `ctxParamsForDeviceChange` data object, the EID and/or TAC of the new Device. The Service Provider SHALL provide `isNewProfileRequired` and optionally

a Service Provider Message for Device Change in the response of ES2+.HandleDeviceChangeRequest function.
If it is required for the End User to enter a Confirmation Code in order to proceed with the Device Change of the Profile, the Service Provider SHALL provide the value of the Confirmation Code in the response of ES2+.HandleDeviceChangeRequest function.

7. If the SM-DP+ does not support Device Change, the SM-DP+ SHALL return an error status "Device Change – Unsupported" and the procedure SHALL stop. If the Device Change is not allowed for the Profile identified by the ICCID, the SM-DP+ SHALL return an error status "Device Change – Not Allowed" and the procedure SHALL stop.

   If the LPAd of the old Device receives any error status, the LPAd of the old Device MAY display an appropriate error state to the End User and SHALL stop the procedure.

   NOTE: This provides compatibility with SM-DP+ that does not understand or cannot appropriately process the Device Change request (e.g., v3 SM-DP+ not supporting the Device Change feature or v2 SM-DP+).

   NOTE: If the procedure stopped due to an error, the LPAd MAY send "ES10b.CancelSession" to the eUICC with a reason `sessionAborted`.

8. If configured by the Service Provider, the SM-DP+ SHALL notify the Service Provider of the Device Change request by calling ES2+.HandleNotification function.
9. If the Device Change is allowed for the Profile identified by the ICCID, the SM-DP+ returns ES9+.AuthenticateClient response comprising `transactionId`, `smdpSigned4`, `smdpSignature4` and optionally Service Provider Message for Device Change.
10. If `isNewProfileRequired` was set to TRUE in the response to ES2+.HandleDeviceChangeRequest function and/or if there is an agreed behaviour between the Service Provider and the SM-DP+ on the Profile identified by the ICCID, the Service Provider SHALL run the Download Preparation Process, as defined in 3.1.1.2 and optionally the Subscription Activation Process, as defined in 3.1.1.4.

NOTE: This process can be performed in parallel to steps 6 to 19. The Profile identified by the ICCID has to be in 'Released' state before step 20.

11. The LPAd of the old Device SHALL ask for the Strong Confirmation on the Device Change. If Service Provider Message for Device Change was provided in the ES9+.AuthenticateClient response, it SHOULD be presented to the End User.

    If `ccRequiredFlag` is set to TRUE in `smdpSigned4`, the LPAd of the old Device SHALL ask for the End User to enter the Confirmation Code which was provided by the Operator that MAY be considered as a Strong Confirmation.

    The Confirmation Requests described above MAY:

    - display `profileName` or any relevant information contained in the Profile Metadata and `smdpSigned4` to the End User.

- be combined, if prompted, into a single prompt with the highest Confirmation Level therefore requiring a single confirmation by the End User.

If the End User does not confirm the Device Change of the Profile, the LPAd SHALL continue with the Common Cancel Session procedure with reason code 'endUserRejection'. If the End User does not respond to the LPAd prompt within an implementation-dependent timeout interval, the LPAd SHALL cancel the Profile download by performing the Common Cancel Session procedure with the reason 'timeout'. For both cases, the notificationEvent SHALL be set to 'Device Change confirmation failure' if a notification is sent to the Service Provider.

12. The LPA of the old Device SHALL call the "ES10b.PrepareDeviceChange" function including the smdpSigned4, smdpSignature4 and optionally the Hashed Confirmation Code. The Hashed Confirmation Code SHALL be calculated with the UTF-8-encoded representation of the Confirmation Code as follows:

    Hashed Confirmation Code = SHA256 (SHA256(Confirmation Code) | TransactionID), where '|' means concatenation of data

13. The LPAd of the old Device SHALL call ES9+.ConfirmDeviceChange function comprising transactionId, prepareDeviceChangeResponse.
14. If configured by the Service Provider or if isNewProfileRequired was set to TRUE in the response to ES2+.HandleDeviceChangeRequest function, the SM-DP+ SHALL notify the Service Provider of the End User's confirmation result by calling ES2+.HandleNotification function. If the End User accepted the Device Change, the procedure continues with the next step. Otherwise, the procedure continues with step (16).
15. If isNewProfileRequired was set to FALSE in the response to ES2+.HandleDeviceChangeRequest function or if configured by the Service Provider, the SM-DP+ SHALL prepare a Profile for download and the associated MatchingID. If an EID was provided in the Device Change Request in the step 5, the SM-DP+ SHALL link the prepared Profile download with the EID. The SM-DP+ SHALL determine the deletion of the Profile on the old Device as per Service Provider's configuration and SHALL generate the associated Activation Code. If the Activation Code is to be encrypted as per section 5.6.6, the SM-DP+ SHALL use a MatchingID that has not previously been used in the Activation Code. The SM-DP+ SHALL notify the Service Provider of the Profile preparation result by calling ES2+.HandleNotification function if configured by the Service Provider.
16. If the End User accepted the Device Change, the SM-DP+ SHALL return the ES9+.ConfirmDeviceChange response comprising the Device Change response. Upon receiving the response, the LPAd of the old Device SHOULD disable the installed Profile if the response contains encryptedDeviceChangeData, and SHALL call "ES10b.VerifyDeviceChange" function comprising deviceChangeConfirmation to verify the SM-DP+ signature and optionally decrypt the Device Change Response via eUICC as described in section 5.7.27. If the eUICC returns a profileNotInDisabledState error, the LPA MAY disable the installed Profile and retry the "ES10b.VerifyDeviceChange" function call. If the eUICC returns any other

error or the LPA does not retry the "ES10b.VerifyDeviceChange" function call, the procedure SHALL stop.

NOTE 1: The use of an SM-DS in the context of Device Change is FFS.

NOTE 2: If the LPA does not retry the "ES10b.VerifyDeviceChange" function call, the LPA can terminate the RSP session by calling "ES10b.CancelSession" with the reason `sessionAborted`.

If the End User rejected the Device Change, the SM-DP+ SHALL return the ES9+.ConfirmDeviceChange response without the Device Change response, and the procedure SHALL stop.

17. If the LPAd of the old Device has been instructed in the Device Change Response to delete the installed Profile or the `DeviceChangeConfiguration` indicates that the deletion of the installed profile is required, the LPAd of the old Device SHALL delete the installed Profile from the eUICC using ES10c.DeleteProfile and retrieve the corresponding Delete Notification from the eUICC. Additionally, if the `DeviceChangeConfiguration` indicates `requestToDp` and the SM-DP+ has indicated in the Device Change Response that it supports the recovery of the deleted Profile, the LPAd of the old Device SHOULD store the following values of the deleted Profile:
    - ICCID, and
    - from `DeviceChangeConfiguration:` the `smdpAddressForDc` and, if present, the `allowedCiPKId`.

    If the deletion of the installed Profile is not required, the procedure continues with step (19).

    NOTE: The LPA of the old Device should store the Profile Recovery Information until the expiration of time indicated in `profileRecoveryValidityPeriod` in the `deviceChangeResponse` or successful Profile Recovery, whichever comes first.

18. The LPAd of the old Device SHALL send the Delete Notification of the deleted Profile to the corresponding Recipient Address. For that, the LPAd MAY perform one of the following:

    - The LPAd MAY call ES9+.HandleNotification function (as defined in section 3.5) and receive the acknowledgement of the Delete Notification.
    - The LPAd MAY send the Delete Notification to the LPAd of the new Device via implementation-specific channel. In this case the LPAd of the new Device SHALL relay the Delete Notification by calling ES9+.HandleNotification function (as defined in section 3.5) before executing step (20).
    - If the SM-DP+ has indicated that it supports the Delete Notification for Device Change of the deleted Profile in the Device Change Response, the LPAd MAY embed the Delete Notification for Device Change corresponding to the `notificationAddress` in the Device Change Response in an Activation Code (as defined in section 4.1 and 4.1.3).

The procedure SHALL stop if the LPAd of the old Device cannot send the Delete Notification.

NOTE1: The Recipient Address may not be the FQDN of the SM-DP+ in figure 21b. In such a case, it is out of scope of this document how the Notification receiver delivers the Delete Notification to the SM-DP+ in figure 21b.

NOTE2: The LPAd of the old Device MAY send additional Notifications to the SM-DP+(s) other than the SM-DP+ in figure 21b.

NOTE3: Execution of this step MAY require an extended period of time. For instance, if the LPAd does not presently have network connectivity, it can wait until connectivity is available in order to deliver the notification.

19. The LPAd of the old Device provides the Activation Code to the LPAd of the new Device.

    - If the `DeviceChangeConfiguration` indicates `requestToDp`, the LPAd SHALL use the Activation Code in the ES9+.ConfirmDeviceChange response or the Activation Code at step (18), if generated.

    - If the `DeviceChangeConfiguration` indicates `usingStoredAc`, the LPAd SHALL use the Activation Code from the `activationCodeForDc`.

    The LPAd of the old Device SHOULD provide to the new Device the state of the Profile during the procedure in order to allow the LPAd of the new Device to restore the state of the Profile of the old Device.

    The LPAd of the old Device MAY present the information via the LUI. Additional means to provide the information to the LPAd of the new Device is out of scope of this specification.

20. The Profile is downloaded from the SM-DP+ to the new Device via the Profile download and installation procedure as defined in section 3.1.3, based upon the Activation Code.

    - If the Activation Code, provided at step (19), contains the Delete Notification for Device Change, the LPAd of the new Device SHALL provide the Delete Notification for Device Change by calling ES9+.AuthenticateClient function comprising a `deleteNotificationForDc`.
    - If the Activation Code, provided at step (19), indicates Confirmation Code Required Flag, the Confirmation Code that was used in this procedure has to be used at step (8) of the Profile download and installation procedure defined in section 3.1.3.

End Conditions:

The Profile and its associated Profile Metadata have been installed on the End User's eUICC of the new Device.

3.11.1.1 Multiple Profiles Device Change

The LPA supporting Device Change SHALL be able to allow the End User to perform the Device Change procedure for one or more Profiles installed in the eUICC.

The LUId SHOULD present to the End User all applicable Profiles and allow their selection for the Device Change operation.

For each selected Profile, steps from 2 to 20 of section 3.11.1 are repeated. When multiple Profiles have been selected for Device Change, if one of these steps fails due to an error, the LPAd SHOULD continue the Device Change procedure with the other Profiles.

If more than one Profile is selected for Device Change, the LPAd of the old Device SHOULD provide all Activation Codes to the LPAd of the new Device in one step. In this case, step 19 of section 3.11.1 is put on hold until all Activation Codes are ready to be provided to the new Device.

3.11.2 Profile Recovery

This procedure will allow the End User to recover the deleted Profile on the old Device.

```
@startuml
hide footbox
skinparam sequenceMessageAlign center
skinparam sequenceArrowFontSize 12
skinparam noteFontSize 12
skinparam monochrome true
skinparam ArrowColor Black
skinparam lifelinestrategy solid
skinparam sequenceMessageAlign center
skinparam noteBackgroundColor #FFFFFF
skinparam participantBackgroundColor #FFFFFF
skinparam boxPadding 6
hide footbox

participant "End User" as EU
box "Old Device"
participant "LPAd" as LPA1
participant "eUICC\nLPA Services (ISD-R)" as EUICC1
end box
participant "SM-DP+" as DP
participant "Service Provider" as SP

rnote over EU, LPA1 : [1] End User interactions

rnote right EU
[2]
Get ICCID, SM-DP+ Address,
[allowedCiPkId]
endrnote

rnote over LPA1, DP: [3] Common Mutual Authentication Procedure, see 3.0.1

LPA1 -> DP: ES9+.AuthenticateClient request \n (ctxParamsForProfileRecovery)

rnote left SP
[4] Verify the Profile Installation Result of
    the new Device for Device Change
endrnote
DP --> LPA1: [error]
rnote over DP, SP: [5] Prepare Profile Download and Activation Code
DP --> LPA1: [6] ES9+.AuthenticateClient response \n (transactionId, smdpSigned4,
smdpSignature4)

Opt If eUICC supports Device Change
LPA1 -> EUICC1: [7] ES10b.VerifyDeviceChange\n(profileRecoveryResponse)
```

```
alt If verification fails
EUICC1 --> LPA1 : error
else Otherwise
EUICC1 --> LPA1 : OK
end
end
rnote over DP, LPA1
[8] Profile Download and Installation Procedure, see 3.1.3
endrnote
@enduml
```

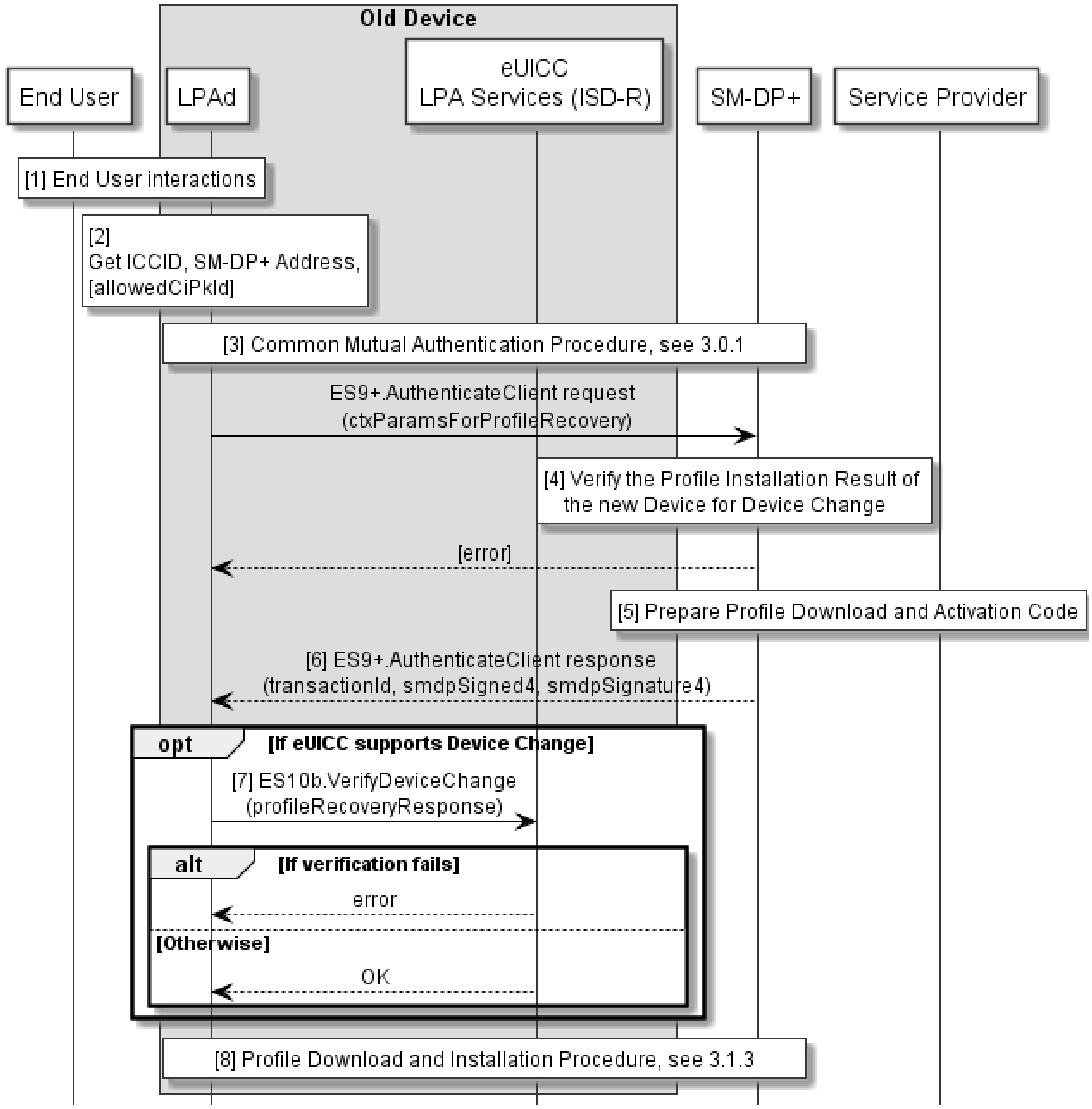

Old Device
End User
LPAd
eUICC
LPA Services (ISD-R)
SM-DP+
Service Provider
[1] End User interactions
[2]
Get ICCID, SM-DP+ Address,
[allowedCiPkId]
[3] Common Mutual Authentication Procedure, see 3.0.1
ES9+.AuthenticateClient request
(ctxParamsForProfileRecovery)
[4] Verify the Profile Installation Result of
the new Device for Device Change
[error]
[5] Prepare Profile Download and Activation Code
[6] ES9+.AuthenticateClient response
(transactionId, smdpSigned4, smdpSignature4)
opt
[If eUICC supports Device Change]
[7] ES10b.VerifyDeviceChange
(profileRecoveryResponse)
alt
[If verification fails]
error
[Otherwise]
OK
[8] Profile Download and Installation Procedure, see 3.1.3

Figure 29n: Profile Recovery procedure

Start Conditions:

- The LPAd of the old Device has deleted the installed Profile as instructed by the SM-DP+ during the Device Change procedure.
- The SM-DP+ has indicated to the LPAd of the old Device the support of the recovery of the deleted Profile in the Device Change Response.

- The relevant information for the recovery of the deleted Profile has been stored and validity period of the Profile Recovery is not expired in the LPAd of the old Device.
- The Profile Installation on the eUICC of the new Device has failed due to a permanent error described in section 2.5.6.1.
- The LPAd of the new Device has delivered the Profile Installation Result to the SM-DP+ by calling the "ES9+.HandleNotification".

Procedure:

1. The End User initiates the Profile Recovery operation within the LUId of the old Device by selecting the Profile that was deleted for Device Change.
2. The LPAd of the old Device retrieves the SM-DP+ address and an optional allowed eSIM CA RootCA Public Key identifier for recovery of the selected Profile. If the LPAd of the old Device cannot retrieve the SM-DP+ address, the procedure SHALL stop.
3. The LPAd of the old Device initiates the Common Mutual Authentication procedure defined in section 3.0.1 to the retrieved SM-DP+ address. If an allowed eSIM CA RootCA public key identifier was retrieved, the LPAd SHALL restrict the allowed eSIM CA RootCA public key identifiers to that value.
   During the Common Mutual Authentication procedure at step (10), the LPAd SHALL build the `ctxParams1` data object with `ctxParamsForProfileRecovery` comprising the ICCID of the selected Profile to be recovered.
4. On reception of the "ES9+.AuthenticateClient" function call comprising `ctxParamsForProfileRecovery`, the SM-DP+ SHALL verify that there was a permanent error in installing the prepared Profile on the new Device for Device Change, corresponding to the ICCID therein. If verification fails, the SM-DP+ SHALL return a status code "Profile – Not allowed".
5. The SM-DP+ SHALL prepare a Profile for recovery and the associated Activation Code for the old Device. The SM-DP+ MAY interact with the Service Provider for the Profile preparation.
6. The SM-DP+ SHALL return the ES9+.AuthenticateClient response comprising `transactionId, smdpSigned4` and `smdpSignature4`.
7. Upon receiving the response, if the eUICC supports Device Change, the LPAd of the old Device SHALL call "ES10b.VerifyDeviceChange" function comprising `profileRecoveryResponse` to verify the SM-DP+ signature via eUICC as described in section 5.7.27. If the eUICC returns an error, the procedure SHALL stop.
8. The LPAd of the old Device initiate Profile download and installation procedure, as defined in section 3.1.3, by opening a new RSP session to the SM-DP+ identified by the Activation Code contained in the `smdpSigned4` (see step 6 above).

End Conditions:

The Profile for recovery and its associated Profile Metadata have been installed on the End User's eUICC of the old Device.

4 Data Elements

4.1 Activation Code

The Activation Code SHALL be coded to be the concatenation of the strings listed in the following table:

Name	Description	MOC
AC_Format	Format of the Activation Code. SHALL be set to "1" for this format of the Activation Code and any subsequent backward compatible format NOTE: The value "2" designates an Advanced Activation Code, as described in section 4.1a	M
Delimiter	SHALL be set to "$"	M
SM-DP+ Address	FQDN (Fully Qualified Domain Name) of the SM-DP+ (e.g., SMDP.EXAMPLE.COM) restricted to the Alphanumeric mode character set defined in table 5 of ISO/IEC 18004 [15] excluding "$"	M
Delimiter	SHALL be set to "$"	M
AC_Token	MatchingID as described in section (0)	M
Delimiter	SHALL be present and set to "$" if any of the following optional parameters is present	C
SM-DP+ OID	SM-DP+ OID in the CERT.DPauth.SIG	O
Delimiter	SHALL be present and set to "$" if any of the following optional parameters is present	C
Confirmation Code Required Flag	SHALL be present and set to "1" if Confirmation Code is required; otherwise it SHALL be absent	O
Delimiter	SHALL be present and set to "$" if any of the following optional parameters is present	C
CI Public Key indicator	#SupportedFromV3.0.0# If present, it specifies an indicator of an eSIM CA RootCA public key as specified below.	O
Delimiter	SHALL be present and set to "$" if any of the following optional parameters is present	C
Delete Notification for Device Change	#SupportedForDcV3.0.0# If present in an Activation Code, it specifies the Delete Notification for Device Change. See section 4.1.3.	O

Table 8: Activation Code Structure

The maximum length of an Activation Code which does not contain a Delete Notification for Device Change SHALL be 255 characters, but in practise it is recommended to consider the user experience when choosing the length. When the Activation Code contains a Delete Notification for Device Change, the length of the Activation Code MAY be longer than 255 characters.

To support extension by future versions of this specification, the Device SHALL ignore a delimiter and any further parameters following those defined in Table 8.

The Device SHALL treat an AC_Format other than "1" or "2" as invalid.

Examples of the Activation Code are as follows:

- 1$SMDP.EXAMPLE.COM$04386-AGYFT-A74Y8-3F815
(if SM-DP+ OID and Confirmation Code Required Flag are not present)
- 1$SMDP.EXAMPLE.COM$04386-AGYFT-A74Y8-3F815$$1
(if SM-DP+ OID is not present and Confirmation Code Required Flag is present)
- 1$SMDP.EXAMPLE.COM$04386-AGYFT-A74Y8-3F815$1.3.6.1.4.1.31746$1
(if SM-DP+ OID and Confirmation Code Required flag are present)
- 1$SMDP.EXAMPLE.COM$04386-AGYFT-A74Y8-3F815$1.3.6.1.4.1.31746
(If SM-DP+ OID is present and Confirmation Code Required Flag is not present)
- 1$SMDP.EXAMPLE.COM$$1.3.6.1.4.1.31746
(If SM-DP+ OID is present, Activation token is left blank and Confirmation Code Required Flag is not present)
- 1$SMDP.EXAMPLE.NET$KL14XA-8C7RLY$1.3.6.1.4.1.31746$$A14D8-971
(If SM-DP+ OID and CI Public Key indicator are present)

When entered manually, the Activation Code SHALL be used as defined above.

When provided in a QR code according to ISO/IEC 18004 [15], the Activation Code SHALL be prefixed with "LPA:".

4.1.1 Matching ID

The MatchingID is mandatory information (but MAY be zero-length) that SHALL be set-up between the Operator and the SM-DP+, to identify the context of a specific management order given to the SM-DP+. The MatchingID is generated during the download initiation procedure (section 3.1.1) or RPM initiation procedure (section 3.7.1).

The MatchingID included in an Activation Code is equivalent to the "Activation Code Token" as defined is SGP.21 [4].

The format and content of the MatchingID is subject to the following constraints:

The MatchingID, when not a zero-length value, SHALL be a unique identifier in the context of the Operator and the SM-DP+ to:

- Match a download order initiated by the Operator with a Profile Download or RPM request coming from an LPD.
- As a protection for the SM-DP+: the SM-DP+ SHALL only process requests containing a MatchingID known to the SM-DP+ (and therefore inherently valid).

It SHALL consist only of upper case alphanumeric characters (0-9, A-Z) and the "-" in any combination.

NOTE: This selection allows more compact alphanumeric QR code encoding and is expected to be supported for manual entry.

On ES9+ and ES11, the data object containing the MatchingID MAY also be missing completely. Whenever this specification refers to a "missing MatchingID", this refers to this data object either to be missing or to having a length of zero.

In turn, a MatchingID being present refers to the data object having a non-zero-length value.

4.1.2 eSIM CA RootCA Public Key indicator

The eSIM CA RootCA Public Key indicator is the potentially truncated hexadecimal representation of the eSIM CA RootCA public key identifier. It SHALL consist of an even number of hexadecimal characters with letters written in upper case and no inserted spaces. "-" characters MAY be inserted at any place for better readability.

NOTE: As the number of eSIM CA RootCAs is limited, a short string can be sufficient to uniquely identify one of these.

4.1.3 Delete Notification for Device Change

Delete Notification for Device Change contains a part of Delete Notification of the deleted Profile excluding EUM Certificate, Sub-EUM Certificate (if any), and eUICC Certificate therein.

`DeleteNotificationForDc` is defined as follows:

```
-- ASN1START
DeleteNotificationForDc ::= SEQUENCE {
  notificationMetadata NotificationMetadata,
  euiccNotificationSignature EuiccSign
}
-- ASN1STOP
```

The `notificationMetadata` and `euiccNotificationSignature` data object SHALL contain the value of `tbsOtherNotification` and `euiccNotificationSignature` data object in the retrieved the `OtherSignedNotification` TLV of the deleted Profile, respectively.

When the LPA of the old Device generates an Activation Code containing a Delete Notification for Device Change, the LPA SHALL use the ASN.1 DER encoded `DeleteNotificationForDc` TLV in hexadecimal byte expression using the character set 0 to 9 and A to F in any combination.

4.2 Device Information

Device Information is mainly in destination of the SM-DP+ for the purpose of Device eligibility check. The SM-DP+/Operator is free to use or ignore this information at their discretion.

Device Information includes:

- TAC
- Device capabilities: The Device SHALL set all the capabilities it supports
  - Radio access technologies, including release.
  - Contactless: the SWP and HCI interfaces as well as the associated APIs
  - RSP CRL SVN
  - LPA SVN
  - Card Application Toolkit support.

  - eUICC form factor type

- IMEI (optional)
- Preferred languages (optional)
- Device Test Mode
- RSP functions supported:

  - CRL stapling
  - RSP Server certificate chain variant A, B, and C
  - APDU API
  - Enterprise Capable Device
  - LPA Proxy (Profile Content Management)
  - Signed SM-DS responses
  - CI update on the eUICC
  - Event Checking
  - Push Service
  - Pending operations alerting

Device Information

`DeviceInfo` is defined as follows:

```
-- ASN1START
DeviceInfo ::= SEQUENCE {
   tac Octet4,
   deviceCapabilities DeviceCapabilities,
   imei Octet8 OPTIONAL,
   preferredLanguages SEQUENCE OF UTF8String OPTIONAL, --
#DeviceInfoExtensibilitySupported#
   deviceTestMode NULL OPTIONAL, -- #DeviceInfoExtensibilitySupported# if present
the Device is operating in Device Test Mode
   lpaRspCapability LpaRspCapability OPTIONAL -- #DeviceInfoExtensibilitySupported#
}

DeviceCapabilities ::= SEQUENCE { -- Highest fully supported release for each
definition
  -- The device SHALL set all the capabilities it supports
   gsmSupportedRelease VersionType OPTIONAL,
   utranSupportedRelease VersionType OPTIONAL,
   cdma2000onexSupportedRelease VersionType OPTIONAL,
   cdma2000hrpdSupportedRelease VersionType OPTIONAL,
   cdma2000ehrpdSupportedRelease VersionType OPTIONAL,
   eutranEpcSupportedRelease VersionType OPTIONAL,
   contactlessSupportedRelease VersionType OPTIONAL,
   rspCrlSupportedVersion VersionType OPTIONAL, -- #SupportedOnlyBeforeV3.0.0#
   nrEpcSupportedRelease VersionType OPTIONAL, --
#DeviceInfoExtensibilitySupported#
   nr5gcSupportedRelease VersionType OPTIONAL, --
#DeviceInfoExtensibilitySupported#
   eutran5gcSupportedRelease VersionType OPTIONAL, --
#DeviceInfoExtensibilitySupported#
   lpaSvn VersionType OPTIONAL, -- #DeviceInfoExtensibilitySupported# provided for
information only
   catSupportedClasses CatSupportedClasses OPTIONAL, --
#DeviceInfoExtensibilitySupported#
   euiccFormFactorType EuiccFormFactorType OPTIONAL --
#DeviceInfoExtensibilitySupported#
}
```

```
CatSupportedClasses ::= BIT STRING {
    a(0), b(1), c(2), d(3), e(4), f(5), g(6), h(7), i(8), j(9),
    k(10), l(11), m(12), n(13), o(14), p(15), q(16), r(17), s(18), t(19),
    u(20), v(21), w(22), x(23), y(24), z(25), aa(26), ab(27), ac(28), ad(29)
}

-- Definition of EuiccFormFactorType
EuiccFormFactorType ::= INTEGER {
   removableEuicc (0), -- eUICC can be removed
   nonRemovableEuicc (1) -- eUICC cannot be removed
}

-- Definition of LpaRspCapability
LpaRspCapability ::= BIT STRING {
   crlStaplingV3Support (0),
   certChainV3Support (1),
   apduApiSupport (2),
   enterpriseCapableDevice (3),
   lpaProxySupport (4),
   signedSmdsResponseV3Support (5),
   euiccCiUpdateSupport (6),
   eventCheckingSupport (7),
   pushServiceSupport (8),
   pendingOperationAlertingSupport (9)
}
-- ASN1STOP
```

The TAC and IMEI are defined in 3GPP TS 23.003 [35].

The TAC SHALL be represented as a string of 4 octets that is coded as a Telephony Binary Coded Decimal String as defined in 3GPP TS 29.002 [63].

The IMEI (including the check digit) SHALL be represented as a string of 8 octets that is coded as a Telephony Binary Coded Decimal String as defined in 3GPP TS 29.002 [63], except that the last octet contains the check digit (in low nibble) and an 'F' filler (in high nibble). The check digit SHALL be computed according to 3GPP TS 23.003 [35]. The IMEI SHOULD be present if the Device contains a non-removable eUICC.

Example: If IMEI (14 digits) is: 12345678901234, then the check digit is 7 and the value part of the `imei` ASN.1 object is '21 43 65 87 09 21 43 F7'. The value part of the `tac` ASN.1 object is '21 43 65 87'.

NOTE: The SM-DP+ and the Operator should be aware that the IMEI in version 2 of this specification defines an encoding where the order of the nibbles in the last octet is different.

The Device Information MAY include preferred languages. If provided, the preferred languages SHALL be represented as a sequence of language tags as defined by RFC 5646 [70] in decreasing order of preference.

NOTE: The method by which the Device determines the preferred languages is out of scope of this specification.

The Device SHALL indicate whether it is currently operating in Device Test Mode.

The capabilities SHALL be represented as follows:

- **gsmSupportedRelease** – if GSM/GERAN is supported, this SHALL be the highest 3GPP release *N* fully supported by the device, encoded as the octet string {*N*, 0, 0}. If GSM/GERAN is not supported this SHALL NOT be present.
- **utranSupportedRelease** – if UMTS/UTRAN is supported, this SHALL be the highest 3GPP release *N* fully supported by the device, encoded as the octet string {*N*, 0, 0}. If UMTS/UTRAN is not supported this SHALL NOT be present.
- **cdma2000onexSupportedRelease** – if cdma2000 1X is supported, this SHALL be encoded as the octet string {1, 0, 0}. If cdma2000 1X is not supported this SHALL NOT be present.
- **cdma2000hrpdSupportedRelease** – if cdma2000 HRPD is supported, this SHALL be encoded as the octet string {R, 0, 0}. If cdma2000 HRPD is not supported this SHALL NOT be present. The value *R* SHALL represent the EVDO revision as follows:

  - Rev 0 SHALL be encoded as 1
  - Rev A SHALL be encoded as 2
  - Rev B SHALL be encoded as 3

- **cdma2000ehrpdSupportedRelease** – if cdma2000 eHRPD, is supported this SHALL be the highest 3GPP release *N* fully supported by the device, encoded as the octet string {*N*, 0, 0}. If cdma2000 eHRPD is not supported this SHALL NOT be present.
- **eutranEpcSupportedRelease** – if LTE/E-UTRAN using a 4G core network (Evolved Packet Core) is supported, this SHALL be the highest 3GPP release *N* fully supported by the device, encoded as the octet string {*N*, 0, 0}. If LTE/E-UTRAN is not supported this SHALL NOT be present.
- **contactlessSupportedRelease** – if NFC is supported, this SHALL be the highest (*version*, *revision*) number of TS.26 [40], encoded as the octet string {*version*, *revision*, 0}. If NFC is not supported this SHALL NOT be present.
- **rspCrlSupportedVersion** – if load eUICC CRL as it was defined in section 5.7.12 of version 2 of this specification is supported, this SHALL be the highest SGP.22 version number supported by the Device for this function. If this function is not supported, this field SHALL NOT be present.

LPA SHOULD NOT send the following fields to the eUICC unless the eUICC indicates `deviceInfoExtensibilitySupport`.

NOTE: If LPA sends the following fields to an eUICC that does not indicate `deviceInfoExtensibilitySupport`, the eUICC may reject them (see also section 2.4a).

- **nrEpcSupportedRelease** – if NR (5G New Radio) using a 4G core network (Evolved Packet Core) is supported, this SHALL be the highest 3GPP release *N* fully supported by the device, encoded as the octet string {*N*, 0, 0}. If NR using a 4G core network is not supported this SHALL NOT be present.
- **nr5gcSupportedRelease** – if NR using a 5G core network is supported, this SHALL be the highest 3GPP release *N* fully supported by the device, encoded as the octet string {*N*, 0, 0}. If NR using a 5G core network is not supported this SHALL NOT be present.

- **eutran5gcSupportedRelease** – if LTE/E-UTRAN using a 5G core network is supported, this SHALL be the highest 3GPP release *N* fully supported by the device, encoded as the octet string {*N*, 0, 0}. If LTE/E-UTRAN using a 5G core network is not supported this SHALL NOT be present.
- **lpaSvn –** indicates the highest Specification Version Number of this specification supported by the LPA. The SVN SHALL have the same three digit number as the highest supported specification version. Example of value: '3.0.0'. A version 3, or higher, Device SHALL include this information. The `lpaSvn` is provided for information only (see Annex M).
- **catSupportedClasses** – indicates the set of supported Card Application Toolkit letter classes as defined in [31] and [76]. A version 3, or higher, Device SHALL include this information.
- **euiccFormFactorType** – indicates whether the eUICC is removable or non-removable. A version 3, or higher, Device SHALL include this information. If this element is omitted the form factor type of the eUICC is unspecified.

Description of `LpaRspCapability`:

A version 3 or higher Device SHALL include this data object in `DeviceInfo`. Refer to Annex M that describes how a version 3 LPA SHALL be configured.

- The `crlStaplingV3Support` bit SHALL be set to '1' if and only if the LPA supports the CRL stapling during the Common Mutual Authentication procedure.

- The `certChainV3Support` bit SHALL be set to '1' if and only if the LPA supports RSP Server certificate chain Variant A, B and C.

- The `apduApiSupport` bit SHALL be set to '1' if and only if the Device supports the APDU API.

- The `enterpriseCapableDevice` bit SHALL be set to '1' if and only if the Device is an Enterprise Capable Device. The bit SHALL be identical to the corresponding setting in the RSP Device Capabilities.

- The `lpaProxySupport` bit SHALL be set to '1' if and only if the Device supports the LPA Proxy. The bit SHALL be identical to the corresponding setting in the RSP Device Capabilities.

NOTE: The LPA Proxy is usable only if the Local Proxy Configuration (section 2.4a.1.5) is also present in the Profile Metadata of the Enabled Profile.

- The `signedSmdsResponseV3Support` bit SHALL be set to '1' if and only if the LPA supports the handling of SM-DS signed Event Records.

- The `euiccCiUpdateSupport` bit SHALL be set to '1' if and only if the Device together with the eUICC supports update of eSIM CA RootCA Public Keys on the eUICC. The method by which the Device determines its compatibility with the eUICC for update is proprietary to the Device Manufacturer and EUM.

- The `eventCheckingSupport` bit SHALL be set to '1' if and only if the LPA supports the Event Checking procedure as defined in section 3.6.4.

- The `pushServiceSupport` bit SHALL be set to '1' if and only if the LPA supports the Push Service.

- The `pendingOperationAlertingSupport` bit SHALL be set to '1' if and only if the LPA supports the pending operations alerting defined in this specification. If this bit is set to '1', the Device SHALL also indicate support for REFRESH with "Application Update" mode in the Terminal Profile according to ETSI TS 102 223 [31] on each eSIM Port where a Profile is enabled.

4.3 eUICC Information

The eUICC information comprises EUICCInfo1 and EUICCInfo2. EUICCInfo1 is a subset of EUICCInfo2, designed to be sent to an RSP Server before it is authenticated. EUICCInfo2 contains the full eUICC information and is disclosed to an RSP Server only after it is authenticated.

The eUICC information SHALL be generated by the eUICC and MAY be requested by the LPA at any point in time.

The eUICC information includes:

- Profile Package Versions
- Specification Version Numbers
- Firmware version
- Available amount of non-volatile memory
- UICC capabilities
- ETSI TS 102 241 version
- GlobalPlatform version
- RSP capabilities
- Lists of supported eSIM CA RootCA Public Key Identifiers
- eUICC Category
- Forbidden PPRs
- Protection Profile version
- SAS Accreditation Number
- Certification Data Object
- TRE properties
- TRE product reference
- LPA Mode
- EUM specific information

eUICC Information

EUICCInfo1 and EUICCInfo2 are defined as follows:

```
-- ASN1START
EUICCInfo1 ::= [32] SEQUENCE { -- Tag 'BF20'
   lowestSvn [2] VersionType,
   euiccCiPKIdListForVerification [9] SEQUENCE OF SubjectKeyIdentifier, -- List of
eSIM CA RootCA Public Key Identifiers supported on the eUICC for signature
verification
   euiccCiPKIdListForSigning [10] SEQUENCE OF SubjectKeyIdentifier, -- List of eSIM
CA RootCA Public Key Identifier supported on the eUICC for signature creation that
can be verified by a certificate chain Variant O
   euiccCiPKIdListForSigningV3 [17] SEQUENCE OF SubjectKeyIdentifier OPTIONAL, --
#SupportedFromV3.0.0# List of eSIM CA RootCA Public Key Identifiers supported on
the eUICC for signature creation that can be verified by a certificate chain
different from Variant O.
   euiccRspCapability [8] EuiccRspCapability OPTIONAL, -- #MandatoryFromV3.0.0#
   highestSvn [19] VersionType OPTIONAL -- #SupportedFromV3.0.0#
}

EUICCInfo2 ::= [34] SEQUENCE { -- Tag 'BF22'
   baseProfilePackageVersion [1] VersionType,  -- Base eUICC Profile package
version supported
   lowestSvn [2] VersionType,
   euiccFirmwareVersion [3] VersionType,       -- eUICC Firmware version
   extCardResource [4] OCTET STRING,     -- Extended Card Resource Information
according to ETSI TS 102 226
   uiccCapability [5] UICCCapability,
   ts102241Version [6] VersionType OPTIONAL,
   globalplatformVersion [7] VersionType OPTIONAL, -- #MandatoryFromV3.0.0#
   euiccRspCapability [8] EuiccRspCapability,
   euiccCiPKIdListForVerification [9] SEQUENCE OF SubjectKeyIdentifier, -- List of
eSIM CA RootCA Public Key Identifiers supported on the eUICC for signature
verification
   euiccCiPKIdListForSigning [10] SEQUENCE OF SubjectKeyIdentifier, -- List of eSIM
CA RootCA Public Key Identifier supported on the eUICC for signature creation that
can be verified by a certificate chain Variant O
   euiccCategory [11] INTEGER {
      other(0),
      basicEuicc(1),
      mediumEuicc(2),
      contactlessEuicc(3)
   } OPTIONAL, -- #MandatoryFromV3.0.0#
   forbiddenProfilePolicyRules [25] PprIds OPTIONAL, -- Tag '99'
   ppVersion VersionType, -- Protection Profile version
   sasAcreditationNumber UTF8String (SIZE(0..64)),
   certificationDataObject [12] CertificationDataObject OPTIONAL, --
#MandatoryFromV3.0.0#
   treProperties [13] BIT STRING {
      isDiscrete(0),
      isIntegrated(1),
      usesRemoteMemory(2) -- refers to the usage of remote memory protected by
                 -- the Remote Memory Protection Function described in SGP.21 [4]
   } OPTIONAL, -- #MandatoryFromV3.0.0#
   treProductReference [14] UTF8String OPTIONAL, -- #SupportedFromV3.0.0#
Platform_Label as defined in GlobalPlatform DLOA specification [57]
   additionalProfilePackageVersions [15] SEQUENCE OF VersionType OPTIONAL, --
#SupportedFromV3.0.0#
   lpaMode [16] LpaMode OPTIONAL, -- #MandatoryFromV3.0.0# active LPA
   euiccCiPKIdListForSigningV3 [17] SEQUENCE OF SubjectKeyIdentifier OPTIONAL, --
#SupportedFromV3.0.0# List of eSIM CA RootCA Public Key Identifiers supported on
the eUICC for signature creation that can be verified by a certificate chain
different from Variant O.
   additionalEuiccInfo [18] OCTET STRING (SIZE(0..32)) OPTIONAL,   --
#SupportedFromV3.0.0# EUM specific eUICC information
   highestSvn [19] VersionType OPTIONAL -- #SupportedFromV3.0.0#
}

-- Definition of EuiccRspCapability
EuiccRspCapability ::= BIT STRING {
   additionalProfile(0), -- at least one more Profile can be installed
   loadCrlSupport(1), -- #SupportedOnlyBeforeV3.0.0# Support for ES10b.LoadCRL
   rpmSupport(2), -- Remote Profile Management
   testProfileSupport (3), -- support for test profile
   deviceInfoExtensibilitySupport (4),  -- #SupportedFromV2.2.2# support for ASN.1
extensibility in the Device Info
   serviceSpecificDataSupport (5), -- #SupportedFromV2.4.0# support for Service
Specific Data in the Profile Metadata
   hriServerAddressSupport (6), -- #SupportedFromV3.0.0# support for storing HRI
server address
   serviceProviderMessageSupport (7), -- #SupportedFromV3.0.0# Service Provider
message is allowed within Profile metadata
   lpaProxySupport (8), -- #SupportedForLpaProxyV3.0.0# support for LPA Proxy
   enterpriseProfilesSupport (9), -- #SupportedForEnterpriseV3.0.0# support for
enterprise profiles
   serviceDescriptionSupport (10), -- #SupportedFromV3.0.0# support for storing
Service Description
   deviceChangeSupport (11), -- #SupportedFromV3.0.0# support for Device change
   encryptedDeviceChangeDataSupport (12), -- #SupportedFromV3.0.0# support for
encrypted Device Change data in Device Change response
   estimatedProfileSizeIndicationSupport (13), -- #SupportedFromV3.0.0# support for
including estimated profile size
   profileSizeInProfilesInfoSupport (14), -- #SupportedFromV3.0.0# support for
profile size in GetProfilesInfo
   crlStaplingV3Support (15), -- #SupportedFromV3.0.0# support for CRL stapling
   certChainV3VerificationSupport (16), -- #SupportedFromV3.0.0# support for
certificate chain verification Variant A, B and C
   signedSmdsResponseV3Support (17), -- #SupportedFromV3.0.0# support for SM-DS
signed response
   euiccRspCapInInfo1 (18), -- #SupportedFromV3.0.0# EUICCInfo1 includes
euiccRspCapability
   osUpdateSupport (19), -- #SupportedFromV3.0.0# support for eUICC OS Update
   cancelForEmptySpnPnSupport (20), -- #SupportedFromV3.0.0# support for cancel
session reasons empty SPN and empty Profile Name
   updateNotifConfigInfoSupport (21), -- #SupportedFromV3.0.0# support for updating
NotificationConfigurationInfo as defined in section 5.4.1
   updateMetadataV3Support (22) -- #SupportedFromV3.0.0# support for the modified
update metadata mechanism defined in section 5.4.1
}

-- Definition of CertificationDataObject
CertificationDataObject ::= SEQUENCE {
   platformLabel UTF8String,      -- Platform_Label as defined in GlobalPlatform
DLOA specification [57]
   discoveryBaseURL UTF8String    -- Discovery Base URL of the SE default DLOA
Registrar as defined in GlobalPlatform DLOA specification [57]
}

-- Definition of LpaMode
LpaMode ::= INTEGER {
   lpad (0), -- LPAd is active
   lpae (1) -- LPAe is active
}
-- ASN1STOP
```

The `baseProfilePackageVersion` field indicates:

- the lowest major version number and
- the associated highest minor version number

of the TCA eUICC Profile Package: Interoperable Format Technical Specification [5] supported by the eUICC. In order to provide backward compatibility, the major version number SHALL indicate 2 and the associated highest minor version number SHALL indicate at least 3.1 (e.g., `baseProfilePackageVersion` = 2.3.1).

NOTE: Backward compatibility may be removed in future version of this specification.

The additionalProfilePackageVersions field lists additional major versions including the associated highest minor version number of the TCA eUICC Profile Package: Interoperable Format Technical Specification [5] supported by the eUICC. This field SHALL be present only when the eUICC supports additional versions of [5] with the major version higher than the one indicated in baseProfilePackageVersion field. This sequence, if present, SHALL contain version 3.1 or higher and SHOULD contain version 3.2 or higher.

The lowestSvn field is deprecated and is only present for backward compatibility with the previous version of this specification. The highestSvn field indicates the highest Specification Version Number of this specification supported by the eUICC and is provided for information only. These fields SHALL be set as defined in Annex M.

NOTE: The lowestSvn field was called svn in the previous versions of the specification.

The euiccFirmwareVersion field indicates the version information of the eUICC's platform and the OS. This value is EUM specific.

The extCardResource field is defined in ETSI TS 102 226 [39]. This field includes the current total available memory, expressed in bytes, for Profile download and installation. The "number of installed application" value field of extCardResource SHALL be set to '00'.

The uiccCapability field contains the UICC capabilities supported by the eUICC. The related type definition SHALL be imported from the following version of the TCA eUICC Profile Package: Interoperable Format Technical Specification [5]:

- for the eUICC, the highest version declared in additionalProfilePackageVersions, the minimum being version 3.2;
- for the SM-DP+, the highest version supported by this server, the minimum being version 3.2; and
- for other entities, version 3.2 or higher.

The ts102241Version field indicates the latest version of ETSI TS 102 241 [53] supported by the eUICC. This field SHALL NOT be present if the eUICC doesn't support this Java Card^{TM} API.

The globalplatformVersion field SHALL indicate the latest version of GlobalPlatform Card Specification [8] supported by the eUICC.

The euiccRspCapability field contains the optional RSP capabilities supported by the eUICC.

The euiccCiPKIdListForVerification field indicates the list of eSIM CA RootCA Public Key Identifiers supported on the eUICC for RSP Server signature verification.

The euiccCiPKIdListForSigning data object contains the list of eSIM CA RootCA Public Key Identifiers supported on the eUICC for signature creation that can be verified by a certificate chain Variant O. The euiccCiPKIdListForSigningV3 contains the list of eSIM CA RootCA Public Key Identifiers supported on the eUICC for signature creation that can be verified by a certificate chain of one of the other Variants. A version 3 or higher eUICC not supporting signature creation that can be verified by a certificate chain Variant O, SHALL include an empty `euiccCiPKIdListForSigning` data object. A version 3 or higher eUICC not supporting signature creation that can be verified by one of the other certificate chain Variants SHALL omit the `euiccCiPKIdListForSigningV3` data object. A version 3 or higher eUICC SHALL provide at least one identifier in one of the two lists.

Elements in `euiccCiPKIdListForVerification`, `euiccCiPKIdListForSigning` and `euiccCiPKIdListForSigningV3` SHALL be set in decreasing order of priority by the eUICC, where the first element in the list is the most preferred and the last element in the list is the least preferred.

NOTE: By indicating a Public Key identifier in the `euiccCiPKIdListForVerification` for supporting signature verification, the eUICC also indicates support for key agreement associated to the Public Key identifier, see section 2.6.5.

The `euiccCategory` field indicates the eUICC category as described below. A version 3 or higher eUICC SHALL include this data object; it was optional before version 3.

Category
Basic eUICCs SHALL be compliant with at least the following features: • Memory size available when no Profiles are installed (EEPROM): 64kB • ISO interface PPS 96 • BIP over HTTPS features
Medium eUICCs SHALL be compliant with at least the following features: • Memory size available when no Profiles are installed (EEPROM): 384kB • ISO interface PPS 97 • BIP over HTTPS features • Processor >= 25MHz • Crypto processor >= 100MHz • Memory Protection Unit
Contactless eUICCs SHALL be compliant with at least the following features: • Memory size available when no Profiles are installed (EEPROM): 1024kB • ISO interface PPS 97 • BIP over HTTPS features • Processor >= 25MHz • Crypto processor >= 100MHz • Memory Protection Unit • In combination with an enabled Operational NFC Profile, a contactless eUICC SHALL support all requirements specified in the SGP.03 GSMA NFC UICC Requirements Specification [55]

The `forbiddenProfilePolicyRules` data object SHALL contain the list of PPRs that are 'forbidden' to be set in any Profile (the `PprIds` type is defined in section 2.8.1.1). A PPR is 'forbidden' when there is no PPAR related to this PPR. In addition, PPR1 is 'forbidden' if an Operational Profile is currently installed on the eUICC.

The information contained in forbiddenProfilePolicyRules data object SHALL be used during the eligibility check performed by the SM-DP+: the SM-DP+ SHALL NOT deliver a Profile containing a PPR 'forbidden' by the eUICC.

The ppVersion data object indicates the version of the GSMA eUICC Protection Profile for Consumer Devices [92] against which the eUICC has been certified. ppVersion V255.255.255 indicates a Field-Test eUICC.

NOTE: An eUICC certified during the interim period, when certification against the GSMA eUICC Protection Profile for Consumer Devices [92] was not available, has a ppVersion of the form V0.X.Y.

The sasAcreditationNumber data object indicates the SAS for RSP accreditation number obtained by the EUM (according to the SAS UP specification [77]).

Description of EuiccRspCapability:

A version 3 or higher eUICC SHALL include this data object with identical content in euiccInfo1 and euiccInfo2. Refer to Annex M that describes how a version 3 eUICC SHALL be configured.

- The additionalProfile bit SHALL be set to '1' if and only if at least one more Profile can be installed.
- The loadCrlSupport bit SHALL be set to '1' if and only if the eUICC supports the optional function ES10b.LoadCRL.

NOTE: Prior to version 3, this bit indicated that the eUICC implemented the ES10b.LoadCRL function which is no longer supported. Setting this bit to 0 in a eUICC version 3 prevents the LPAd, in case a removable eUICC version 3 is inserted in a Device prior to version 3, to use the ES10b.LoadCRL function.

- The rpmSupport bit SHALL be set to '1' if and only if the eUICC supports the Remote Profile Management feature.
- The testProfileSupport bit SHALL be set to '1' if and only if the eUICC supports the Test Profile feature.
- The deviceInfoExtensibilitySupport bit SHALL be set to '1' if and only if the eUICC supports the extensibility in the DeviceInfo.
- The serviceSpecificDataSupport bit SHALL be set to '1' if and only if the eUICC supports the Service Specific Data in the Profile Metadata.
- The hriServerAddressSupport bit SHALL be set to '1' if and only if the eUICC supports the HRI server address in the Profile Metadata.
- The serviceProviderMessageSupport bit SHALL be set to '1' if and only if the eUICC can accept a Service Provider message within the Profile Metadata.

- The lpaProxySupport bit SHALL be set to '1' if and only if the eUICC supports the features for the LPA Proxy.

- The enterpriseProfilesSupport bit SHALL be set to '1' if and only if the eUICC supports the features for Enterprise Profiles.

- The serviceDescriptionSupport bit SHALL be set to '1' if and only if the eUICC supports the service description in the Profile Metadata.

- The deviceChangeSupport bit SHALL be set to '1' if and only if the eUICC supports the features defined for Device Change and Profile Recovery (tagged with #SupportedForDcV3.0.0#) except the encrypted device change data.

- The encryptedDeviceChangeDataSupport bit SHALL be set to '1' if and only if the eUICC supports the encrypted Device Change data in Device Change response.

  NOTE: this bit can be set to '1' if and only if deviceChangeSupport is set to '1'.

- The estimatedProfileSizeIndicationSupport bit SHALL be set to '1' if and only if the eUICC accepts an estimated Profile size in the Profile Metadata.

- The profileSizeInProfilesInfoSupport bit SHALL be set to '1' if and only if the eUICC provides an estimated Profile size in ES10c.GetProfilesInfo.

- The crlStaplingV3Support bit SHALL be set to '1' if and only if the eUICC supports the CRL stapling and certificate revocation status verification during the Common Mutual Authentication procedure.

- The certChainV3VerificationSupport bit SHALL be set to '1' if and only if the eUICC supports the verification of RSP Server certificate chain Variant A, B and C.

- The signedSmdsResponseV3Support bit SHALL be set to '1' if and only if the eUICC supports the handling of SM-DS signed response, e.g., Event Records, i.e., implements the function ES10a.VerifySmdsResponse.

- The euiccRspCapInInfo1 bit SHALL be set to '1' if and only if euiccRspCapability is present in EUICCInfo1.

- The osUpdateSupport bit SHALL be set to '1' if and only if the eUICC supports the OS Update capability.

- The cancelForEmptySpnPnSupport bit SHALL be set to '1' if and only if the eUICC supports cancel session reason code "emptyProfileOrSpName".

- The updateNotifConfigInfoSupport bit SHALL be set to '1' if and only if the eUICC supports updating of the NotificationConfigurationInfo as defined in section 5.4.1.

- The updateMetadataV3Support bit SHALL be set to '1' if and only if the eUICC supports the modified update metadata mechanism defined in section 5.4.1.

Within the `CertificationDataObject`, the `platformLabel` SHALL identify the DLOA in the DLOA Registrar. This value SHALL be coded as defined in GlobalPlatform DLOA specification [57] section 7.1.1. The `discoveryBaseURL` MAY be an empty string or contain a value allowing to discover alternate DLOA Registrar(s) (in complement to the well-known DLOA Registrar, see section 2.2) where the corresponding DLOA(s) MAY be retrieved. This value SHALL be coded as defined in GlobalPlatform DLOA specification [57] section 7.1.2.

The `lpaMode` indicates whether the LPAd or the LPAe is active.

The `additionalEuiccInfo` contains information about the eUICC as defined by the EUM. It MAY correlate with the additional issuer information contained in the EID.

The `treProperties` field describes properties of the TRE that the eUICC is based upon. It SHALL be present for an Integrated eUICC and MAY be present for a Discrete eUICC. This field SHALL contain one of the following settings:

- `isDiscrete`
- `isIntegrated`
- `isIntegrated, usesRemoteMemory`

The `treProductReference` SHALL be present for an Integrated eUICC. This value SHALL be coded as defined in GlobalPlatform DLOA specification [57] section 7.1.1 and contain an unique reference of the Integrated TRE product that the eUICC is based upon. This field SHALL NOT be modifiable.

4.3.1 eUICC identifier (EID)

The EID SHALL uniquely identify an eUICC. The owner of the EIN SHALL guarantee the uniqueness of the EID, also with respect to eUICCs produced according to previous versions of this specification and to all versions of SGP.02 [2].

This section specifies the general structure of the EID. The different parts of the EID may have different sizes depending on the EID assignment scheme.

The EID SHALL have the following general structure:

- The EID SHALL be 32 digits long
- The EID SHALL be built of
  - An EUM Identification Number (EIN) of M digits.
  - An EUM-Specific Identification Number (ESIN) of 30-M digits.
  - A final two digits (31st to 32nd digits) containing check digits calculated over all 32 digits as specified below.

When stored as a byte string, the first digit SHALL be put into the highest four bits of the first byte.

The EUM SHALL construct the EIN and ESIN, according to the requirements of one of the following EID assignment schemes:

- The assignment scheme defined in SGP.29 [89]
- The E.118-based assignment scheme specified in 4.3.1.1

The two check digits are calculated as follows:

1. Replace the two check digits by two digits of 0,

2. Using the resulting 32 digits as a decimal integer, compute the remainder of that number on division by 97,

3. Subtract the remainder from 98, and use the decimal result for the two check digits,

    - If the result is one digit long, its value SHALL be prefixed by one digit of 0.

Verification of the check digits of an EID is performed as follows:

- Using the 32 digits as a decimal integer, compute the remainder of that number on division by 97.
- If the remainder of the division is 1, the verification is successful; otherwise the EID is invalid.

NOTE: Examples of valid EIDs are:

- 8900 1012 0123 4123 4012 3456 7890 1224
- 8900 1567 01020304 0506 0708 0910 1152
- 8904 4011 1122 3344 1122 3344 1122 3321
- 0090 9231 2229 9992 3581 2903 6544 3008

4.3.1.1 Details of the EID structure according to the E.118-based assignment scheme

The EIN SHALL have the following structure:

- A Major Industry Identifier digit of 8 (1st digit), as defined in ISO/IEC 7812 **Error! Reference source not found.**.
- An additional digit of 9 specifying telecommunications, as defined in ISO/IEC 7812 **Error! Reference source not found.**,
- An additional three digits for country code (3rd to 5th digits).
    - If the country code is one digit long, its value SHALL be prefixed by two digits of 0,
    - If the country code is two digits long, its value SHALL be prefixed by one digit of 0.
- An additional three digits for issuer identifier (6th to 8th digits).
    - If the issuer identifier is one digit long, its value SHALL be prefixed by two digits of 0,
    - If the issuer identifier is two digits long, its value SHALL be prefixed by one digit of 0.

The country code and issuer identifier SHALL be assigned as specified in ITU E.118 **Error! Reference source not found.**.

NOTE: The EIN according to this scheme was called IIN in former versions of this specification.

The ESIN SHALL have the following structure:

- Ten digits for issuer specific information (9th to 18th digits),

  - An additional twelve digits for the individual identification number (19th to 30th digits).

4.4 Profile Metadata

During the Profile download and installation procedure, Profile Metadata needs to be provided to the LPAd for display and to the eUICC. Profile Metadata is generated by the SM-DP+ in plain text to be readable by the LPA. Profile Metadata is also contained protected in BPP to be loaded into the eUICC, so that the LPA will be able to access the same information any time after the Profile has been successfully loaded into the eUICC, using the "ES10c.GetProfilesInfo" function.

Profile Metadata values, like any other Profile data, are under the responsibility of, and defined by, the Profile owner. Profile Metadata is communicated to the SM-DP+ by means which are out of scope of this specification.

Profile Metadata includes:

- ICCID of the Profile
- Profile Name (corresponds to "Short description" in SGP.21 [4]) as a plain text information: content free information defined by the Operator
- Service Provider name, as a plain text information: content free information defined by the Operator/Service Provider (e.g., 'Orange', 'AT&T'…)
- End User’s Profile Nickname
- Icon
- Profile Class: indicates the sort of profile among the defined values: 'Test', 'Operational' and 'Provisioning' (section 4.4.1)
- Notification Configuration Information, defined in section 3.5 "Notifications"
- Profile owner, including MCC, MNC, GID1 and GID2 if the Profile is not PIN protected
- Profile Policy Rules (PPRs) (section 2.9.1 and 4.4.2)
- The address of the HRI Server (section 4.4.3)
- Enterprise Configuration (section 4.4.4)
- RPM Configuration (section 2.4a.1.3)
- Service Provider Message (section 2.4a.1.4)
- Local Proxy Configuration (section 2.4a.1.5)
- Service Description (section 4.4.5)

Profile Metadata is available to the LPA during Profile download to provide information to the End User about the Profile to be installed. But it is out of scope of this implementation what the LPA does exactly with this Profile Metadata, e.g., the LPA can display all or only part of this information.

4.4.1 Profile Class

A Profile can be defined as a Test Profile, an Operational Profile or a Provisioning Profile. The Profile Class is set in the Profile Metadata and indicates to the LPA and the eUICC which rules to apply.

4.4.2 Profile Policy Rules

The PPRs are provided within the ES8+.StoreMetadata function of the Bound Profile Package. The `pol` field of the `ProfileHeader` PE of the TCA Profile Package (UPP in section 2.5.2) SHALL NOT be used.

The PPRs defined in this document are coded using the ASN.1 data type `PprIds`, see section 2.4a.1.1. `pprUpdateControl` has no meaning when provided in ES8+.StoreMetadata.

4.4.3 High Resolution Icons

In addition to the (standard) icon provided directly in the Metadata, which has only limited graphical capabilities, a second mechanism is defined, which allows for better graphics.

The Metadata includes the address of an HRI Server, which together with some other parameters can be used by the LPA to retrieve an icon with higher resolution and a different shape. Such an icon SHOULD be used by the LPA instead of the standard icon. Different icons can be retrieved by the LPA for different usages: During Profile download, for Profile selection, etc.

The LPA MAY store icons it retrieved in local memory for later re-use.

See section 5.11.2 for handling by the LPAe.

4.4.4 Enterprise Configuration

The Enterprise Configuration SHALL be provided if and only if a Profile is an Enterprise Profile. The Enterprise Configuration includes the OID and name of the Enterprise, and optionally the Enterprise Rules associated with the Enterprise Profile.

4.4.5 Service Description

The Service Description MAY be provided by the Profile Owner to indicate the services offered by its associated subscription. This field is for information only and no action is specified in this document based on this field.

4.5 Keys and Certificates

4.5.1 Keys and Certificates Naming Conventions

The keys and Certificates used in this specification are named according to the conventions described in this section.

The general name structure is: <XX>.<YY>.<ZZ>

Where:

- <XX> designates the nature of the element, the following values are defined:
    - PK: the public key of an asymmetric key pair
    - SK: the private key of an asymmetric key pair
    - CERT: a Certificate containing a public key
    - otPK: a public key of an asymmetric one-time key pair
    - otSK: a private key of an asymmetric one-time key pair

- <YY> designates the owner of the element, the following values are defined:
    - CI: an eSIM CA
    - CISubCA: an eSIM CA SubCA
    - DP: an SM-DP+ when no further qualification is required
    - DPauth: the Authentication function of an SM-DP+
    - DPpb: the Profile Package Binding function of an SM-DP+
    - DPSubCA: an SM-DP+ SubCA
    - DS: an SM-DS when no further qualification is required
    - DSauth: the Authentication function of an SM-DS
    - DSSubCA: an SM-DS SubCA
    - EUICC: an eUICC
    - EUM: an EUM
    - EUMSubCA: an EUM SubCA
- <ZZ> designates the usage of the element, the following values are defined:
    - SIG: for a digital signature
    - KA: for a key agreement
    - TLS: for TLS connection establishment

Examples:

- PK.EUICC.SIG: Public key of an eUICC, used to verify an eUICC signature.
- CERT.DP.TLS: Certificate of the SM-DP+, used to establish TLS connection
- CERT.DPauth.SIG: Certificate of the SM-DP+, used to verify an SM-DP+ signature for its authentication.

Table 9: Void

4.5.2 Certificates

A Certificate Issuer issues certificates for Remote SIM Provisioning system entities and acts as a trusted root for the purpose of authentication of the entities of the system. The specification supports X.509 certificate format as defined in Section 4.5.2.1.

Certificates used in this specification all chain to an eSIM CA RootCA Certificate, except TLS Certificates that MAY chain to a Public CA Certificate.

Certificates according to v2 of this specification have the following drawbacks:

- The values used in the Certificate Policies extension are not assigned as specified in RFC 5280 [17].

- The Name Constraints extension for EUM and eUICC Certificates are not used as specified in RFC 5280 [17].

- EUM, SM-DP+ and SM-DS Certificates are directly signed by the offline eSIM CA RootCA, which is not the best practice.

For backwards compatibility, v2 Certificates are still covered as Variant O in this specification and eSIM CAs that issued v2 Certificates may still need to support them.

However, for new eSIM CAs where support of v2 Certificates is not required (e.g., if they use an SM2 Signature), it is strongly recommended to only issue Certificates according to Variants Ov3, A, B or C.

4.5.2.0a eUICC Certificate chains

The eUICC Certificate chains are described in the figure below.

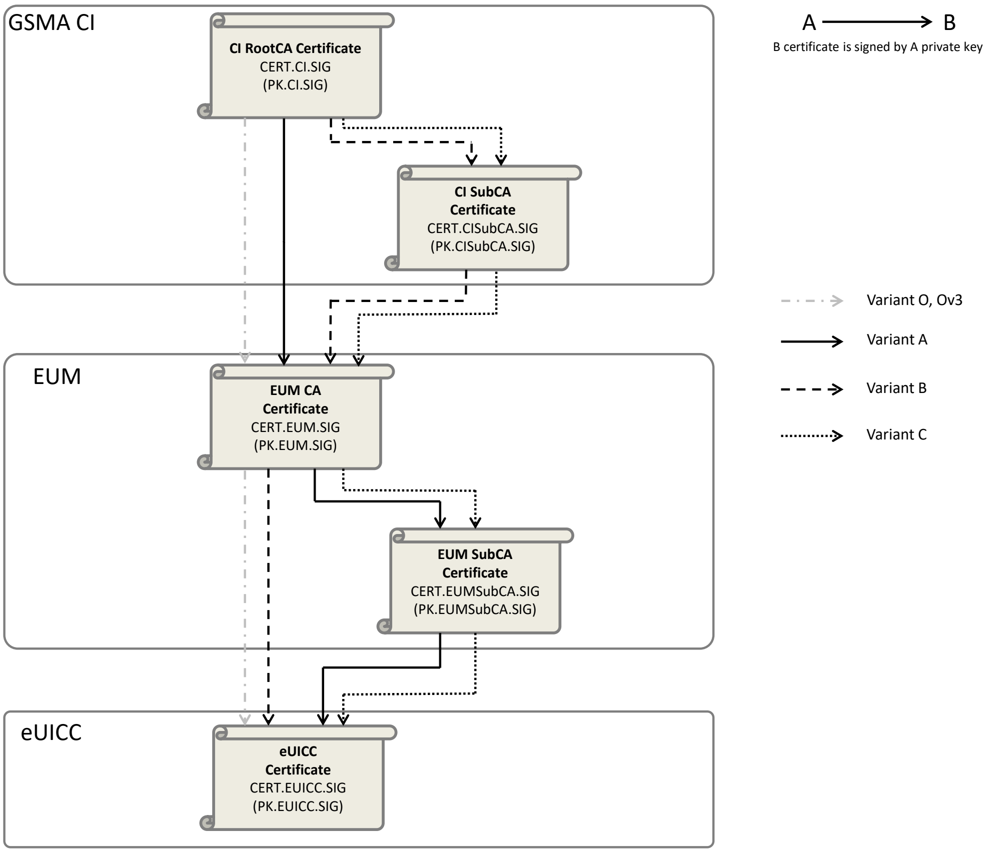


Figure 30: eUICC Certificate Chain

- Variant O (Original) and Variant Ov3: the eUICC Certificate chains to the eSIM CA RootCA Certificate through only the EUM Certificate.
- Variant A: the eUICC Certificate chains to the eSIM CA RootCA Certificate through the EUM SubCA and EUM Certificates.
- Variant B: the eUICC Certificate chains to the eSIM CA RootCA Certificate through the EUM and eSIM CA SubCA Certificates.
- Variant C: the eUICC Certificate chains to the eSIM CA RootCA Certificate through the EUM SubCA, EUM and eSIM CA SubCA Certificates.

Variant O eUICC and EUM Certificates contain Certificate Policies and Name Constraints extensions as defined in v2 of this specification.

All Certificates of the eUICC Certificate chains of all other Variants (including Variant Ov3) contain Certificate Policies, Name Constraints and Permitted EINs extensions as defined in this version of this specification.

Variants A and C may be used, for example, to provide to the EUM the capability to split its eUICC production under different EUM SubCA Certificates while not involving the eSIM CA for that (see section 2.7). If this additional level of EUM SubCA Certificate is used, the EUM SHALL manage its revocation status.

Even though each eUICC SHALL support at least two sets of elliptic curve parameters (section 2.6.5), which can be chosen from by an RSP server for its signatures and key agreement, an eUICC SHALL have at least one CERT.EUICC.SIG.

4.5.2.0b RSP Servers certificate chains

The RSP Server certificate chains are described in the figure below.

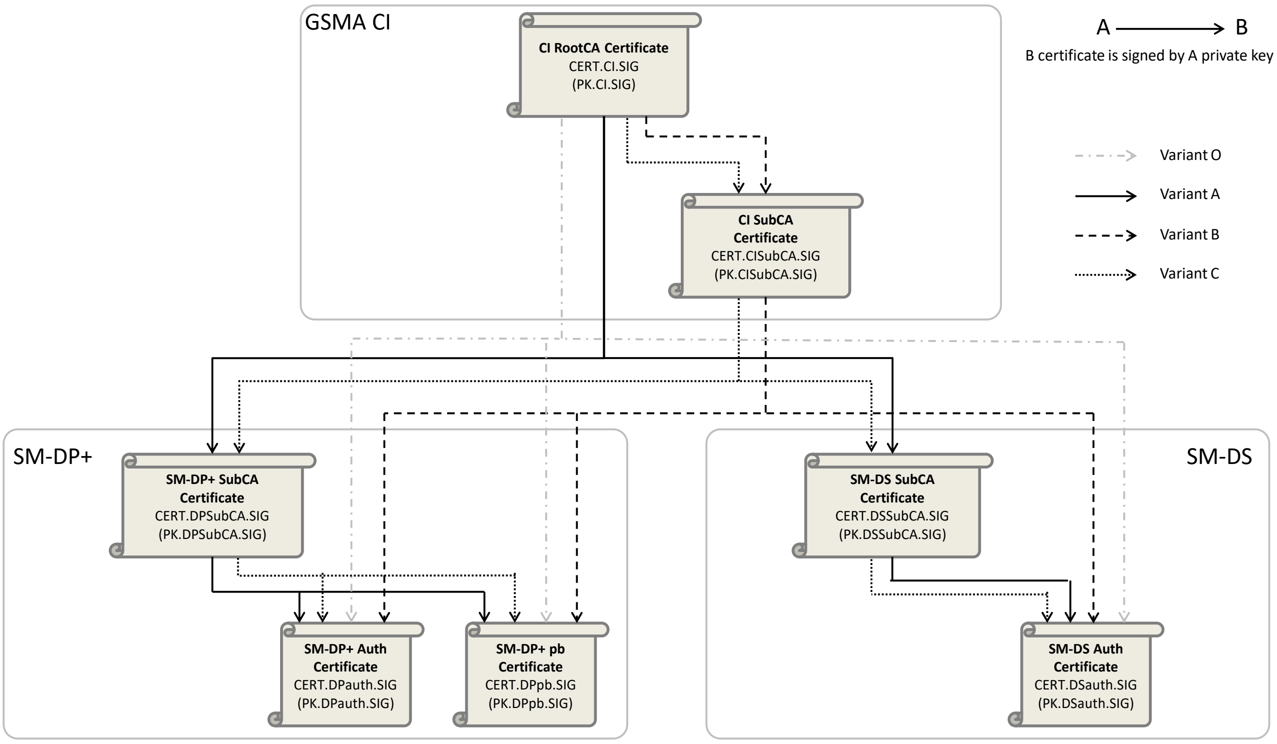


Figure 30a: RSP Server Certificate Chains

- Variant O (Original): the RSP Server Certificates chain directly to the eSIM CA RootCA Certificate (no intermediate SubCA), and containing a Certificate Policies extension as defined in v2 of this specification.
- Variant A: the RSP Server Certificates chain to the eSIM CA RootCA Certificate through intermediate SubCA Certificates under the control of the RSP Server

providers (CERT.DPSubCA.SIG or CERT.DSSubCA.SIG), each containing a Certificate Policies extension as defined in this version of this specification.

- Variant B: the RSP Server Certificates chain to the eSIM CA RootCA Certificate through an intermediate SubCA Certificate under the control of the eSIM CA (CERT.CISubCA.SIG), each containing a Certificate Policies extension as defined in this version of this specification.
- Variant C: the RSP Server Certificates chain to the eSIM CA RootCA Certificate through two levels of intermediate SubCA Certificates, each containing a Certificate Policies extension as defined in this version of this specification.

At least one of the variants A, B or C SHALL be supported by the eSIM CA. Variant O MAY be supported. The SM-DS and SM-DP+ SHALL be able to request their certificate(s) following any of the variants supported by the eSIM CA.

If the Variant A or C is used, the RSP Server MAY choose not to manage revocation status of its leaf certificates. In that case the RSP Server SHALL adopt an appropriate renewal policy in order to mitigate the risk of a compromised certificate being used (i.e., limit their validity periods).

The RSP Server TLS certificate chains are described in the figure below.

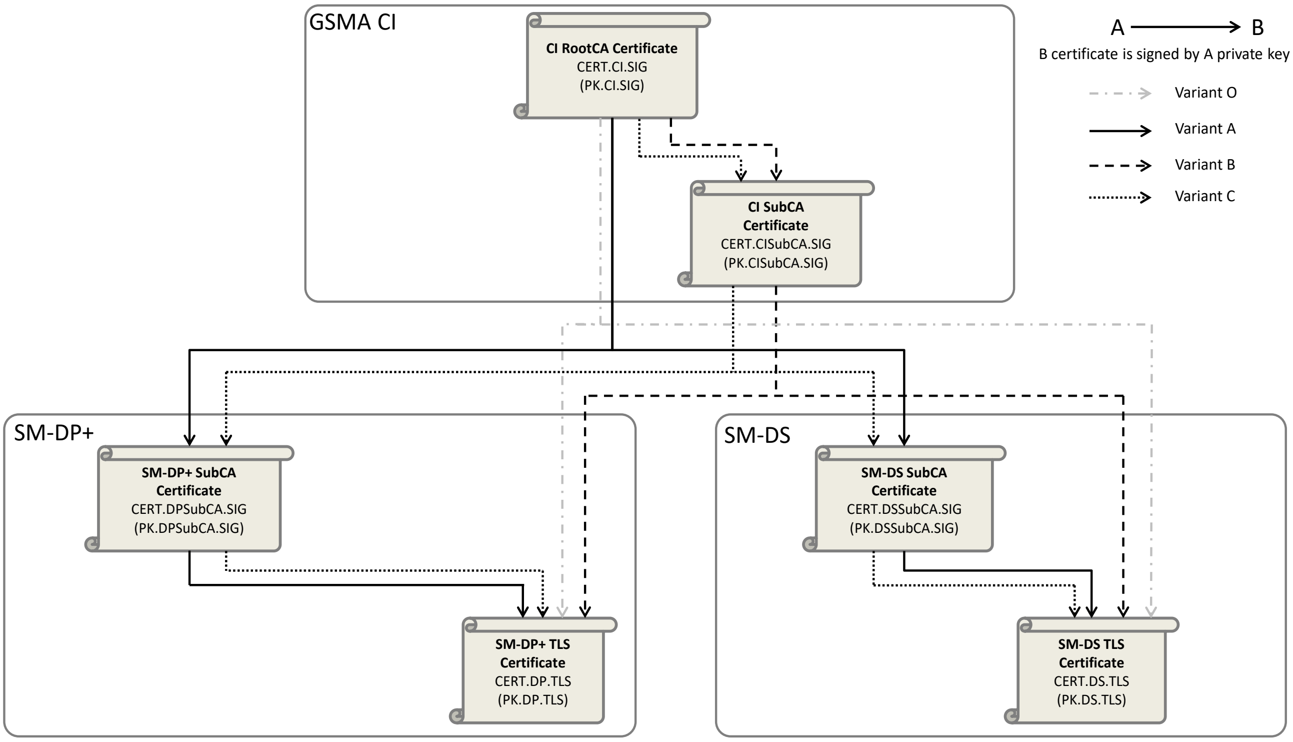


Figure 30b: RSP Server TLS Certificate Chains with eSIM CA

- Variant O (Original): the RSP Server TLS certificates chain directly to the eSIM CA RootCA Certificate (no intermediate SubCA), and containing a Certificate Policies extension as defined in v2 of this specification.

- Variant A: the RSP Server TLS certificates chain to the eSIM CA RootCA Certificate through intermediate SubCA certificates under the control of the RSP Servers (CERT.DPSubCA.SIG or CERT.DSSubCA.SIG), each containing a Certificate Policies extension as defined in this version of this specification.

- Variant B: the RSP Server TLS certificates chain to the eSIM CA RootCA Certificate through an intermediate SubCA certificate under the control of the eSIM CA (CERT.CISubCA.SIG), each containing a Certificate Policies extension as defined in this version of this specification.

- Variant C: the RSP Server TLS certificates chain to the eSIM CA RootCA Certificate through two levels of intermediate SubCA certificates, each containing a Certificate Policies extension as defined in this version of this specification.

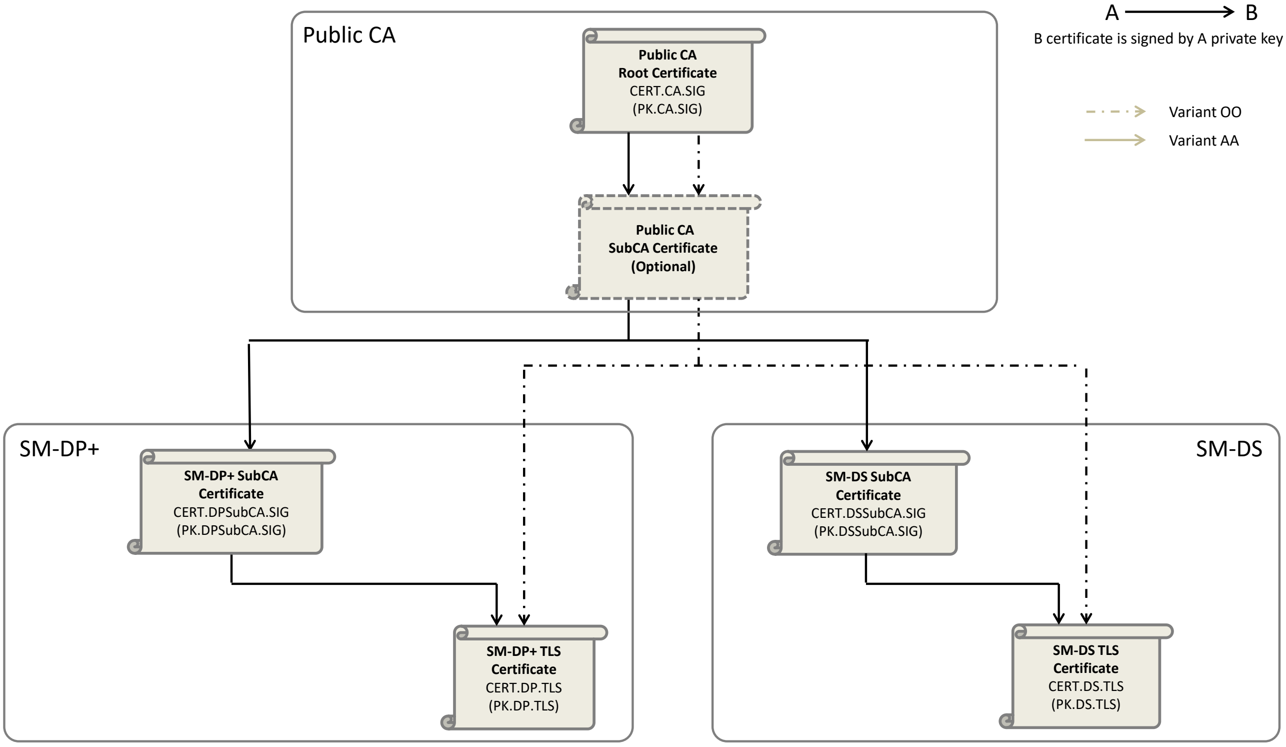


Figure 30c: RSP Server TLS Certificate Chains with Public CA

- Variant OO: the RSP Server TLS certificates chain to a Public CA Root Certificate, with zero, one or several Public CA SubCA Certificates in the chain.

- Variant AA: the RSP Server TLS certificates chain to a Public CA Root Certificate, through an intermediate SubCA certificate under the control of the RSP Servers, and zero, one or several Public CA SubCA Certificates in the chain.

At least one of the variants A or B or C SHALL be supported by the eSIM CA. Variant O MAY be supported. The SM-DS and SM-DP+ SHALL be able to request their certificate(s) following any of the variants supported by the eSIM CA or Public CA.

If the Variant A or AA is used, the RSP Server MAY choose not to manage revocation status of its leaf certificates. In that case the RSP Server SHALL adopt an appropriate renewal policy in order to mitigate the risk of a compromised certificate being used (i.e., limit their validity periods).

The RSP Servers MAY select different variants for certificate chains for TLS and CERT.X.SIG Certificates.

The Algorithm Identifiers of all certificates of a certificate chain SHALL point to the same curve.

The SM-DP+ has 2 Certificates for digital signature (CERT.DPauth.SIG and CERT.DPpb.SIG). The CERT.DPauth.SIG is used for authentication to the eUICC, and the CERT.DPpb.SIG is used for Profile binding.

These certificates are described in the next sections.

4.5.2.1 X.509 Certificate Profile

This section describes the X.509 certificate profile. Those certificates SHALL follow RFC 5280 [17], with the specific coding given in this section.

In particular:

- 'Issuer' and 'Subject' fields SHALL be limited to standard attributes defined in ITU-T X.520 [24] and RFC 4519 [28].
- Certificates SHALL contain all extensions defined in their respective profile, except if stated otherwise.
- Certificates SHALL NOT contain the `freshestCRL` extension (use of Delta CRL is not supported).
- The Subject Key Identifier SHOULD be computed using method 1 specified in section 4.2.1.2 in RFC 5280 [17] for all the certificates listed in section 4.5.2.1.0.

Entities SHALL perform certificate verification according to section 4.5.2.2.

> NOTE: Certificates are described using table representation for easiness, but conform to the ASN.1 format given in RFC 5280 [17].

4.5.2.1.0 Certificates description

4.5.2.1.0.0 Certificates common fields

The table below describes the common fields that all certificates defined in this section SHALL contain. A certificate MAY have additional fields or a different content for a field; in that case it will be indicated in its specific description.

Certificate common fields		
Field	**Value Description**	
tbsCertificate	Data to be signed	
	Field	**Value Description**
	version	Version SHALL be 3 (value is 2) as extensions are used in all certificates defined in this specification.

	serialNumber	Certificate serial number.
	signature	Contains the algorithm identifier used by the issuer to compute the value of the field 'signatureValue' (section 4.5.2.1.1). NOTE: The algorithm identifier value SHALL be the same as the one of the field 'signatureAlgorithm'.
	issuer	Distinguished Name of the entity that has signed the certificate. It SHALL match the 'subject' field of the issuing entity certificate.
	validity	Validity period of the certificate. The validity period allowed for each certificate is defined in SGP.14 [45].

	subject	Distinguished Name of the entity owning the certificate.
		Example of eSIM CA DN:
		cn = GSMA Class 3 Public Primary Certification Authority
		ou = GSMA Trust Network
		o = GSMA
		c = UK

	subjectPublicKeyInfo	Contains the algorithm identifier, parameters and public key value. Algorithm identifier and parameters SHALL be set according to section 4.5.2.1.1. subjectPublicKeyInfo.subjectPublicKey contains the public key value and SHALL be coded as defined in RFC 5480 [27].

	extensions	Extension for Subject Key Identifier. (RFC 5280 [17] section 4.2.1.2). This extension SHALL be set with:
		extnID = id-ce-subjectKeyIdentifier
		critical = false
		extnValue = <Identifier of the public key bound in the certificate>

signatureAlgorithm	Section 4.5.2.1.1	
signatureValue	Signature computed accordingly to one of the possible algorithm listed in section 4.5.2.1.1	

Table 9a: Certificates common fields

4.5.2.1.0.1 eSIM Certificate Issuer

Field	Value Description
issuer	This SHALL be identical to 'subject' field value.

Extension for Key usage (RFC 5280 [17] section 4.2.1.3)	extnID = id-ce-keyUsage
	critical = true
	extnValue = {
	keyCertSign (5),
	cRLSign(6) }

Extension for Certificate Policies (RFC 5280 [17] section 4.2.1.4)	extnID = id-ce-certificatePolicies
	critical = true
	extnValue = id-rspRole-ci (Annex H)
	To indicate the eSIM CA role.

Extension for Basic Constraints (RFC 5280 [17] section 4.2.1.9)	extnID = id-ce-basicConstraints
	critical = true
	extnValue = {
	cA = true }

Extension for subjectAltName (RFC 5280 [17] section 4.2.1.6)	extnID = id-ce-subjectAltName
	critical = false
	extnValue = {
	registeredID (8) = CI OID }

Table 10: CERT.CI.SIG

NOTE: The CERT.CI.SIG is a self-signed certificate, there is no need to include the Extension for Authority Key Identifier.

The table below describes the specific fields of a CERT.CISubCA.SIG in complement of the description given in section 4.5.2.1.0.0:

Field	Value Description
subject	Distinguished Name of the eSIM CA SubCA.

Extension for Authority Key Identifier (RFC 5280 [17] section 4.2.1.1)	extnID = id-ce-authorityKeyIdentifier
	critical = false
	extnValue = <Identifier of the public key to verify this certificate>

Extension for Key usage (RFC 5280 [17] section 4.2.1.3)	extnID = id-ce-keyUsage
	critical = true
	extnValue = {
	keyCertSign (5),
	cRLSign (6)}

Extension for Certificate Policies (RFC 5280 [17] section 4.2.1.4)	extnID = id-ce-certificatePolicies
	critical = true
	extnValue = id-rspRole-ciSubCa (Section 2.4a.1.0)
	To indicate that this is an eSIM CA SubCA Certificate.

Extension for subjectAltName (RFC 5280 [17] section 4.2.1.6)	It SHALL contain the same value as for CERT.CI.SIG of that same eSIM CA.

Extension for Basic Constraints (RFC 5280 [17] section 4.2.1.9)	extnID = id-ce-basicConstraints
	critical = true
	extnValue = {
	cA = true
	}
	To indicate that this certificate is a SubCA with no limit on the number of non-self-issued intermediate CA certificates that may follow.

Extension for CRL Distribution Points (RFC 5280 [17] section 4.2.1.13)	extnID = id-ce-cRLDistributionPoints
	critical = false
	extnValue = section 4.5.2.1.2

Table 10a: CERT.CISubCA.SIG

4.5.2.1.0.2 eUICC

The table below describes the specific fields of a CERT.EUICC.SIG in complement of the description given in section 4.5.2.1.0.0:

Field	Value Description
subject	Distinguished Name of the EUICC. It SHALL include, at least, 'organization' and 'serialNumber' attributes. Others attributes MAY be included for information. The 'organization' attribute SHALL have one of the values allowed in the nameConstraints extension of the EUM Certificate (CERT.EUM.SIG). See note 1. The 'serialNumber' attribute SHALL be the EID as a decimal PrintableString (see note 2). The EID SHALL start with one of the EINs allowed in the EUM Certificate (CERT.EUM.SIG). Example of an eUICC DN: o = ACME serialNumber = 89049032123451234512345678901235

Extension for Authority Key Identifier (RFC 5280 [17] section 4.2.1.1)	extnID = id-ce-authorityKeyIdentifier
	critical = false
	extnValue = <Identifier of the public key to verify this certificate>

Extension for Key usage (RFC 5280 [17] section 4.2.1.3)	extnID = id-ce-keyUsage
	critical = true
	extnValue = digitalSignature (0)

Extension for Certificate Policies (RFC 5280 [17] section 4.2.1.4)	extnID = id-ce-certificatePolicies
	critical = true
	extnValue = id-rspRole-euicc or id-rspRole-euicc-v2 (Section 2.4a.1.0)
	To indicate that this is an eUICC Certificate.
	NOTE: id-rspRole-euicc-v2 indicates a Certificate in a chain following Variant O (see figure 30)

NOTE 1: X.509 mandates that each Certificate in in a certification path obeys the nameConstraints extensions defined in preceding Certificates of the certification path. NOTE 2: former versions of this specification mandated a nameConstraints extension in the EUM Certificate that conflicts with the requirement that attribute 'serial number' of the Subject of the eUICC Certificate SHALL be the EID. Variant O maintains this conflict for backwards compatibility. See section 4.5.2.2 for a recommendation as to how the servers can work around this issue.	

Table 11: CERT.EUICC.SIG

4.5.2.1.0.3 EUM

The table below describes the specific fields of a CERT.EUM.SIG in complement of the description given in section 4.5.2.1.0.0:

Field	Value Description

subject	Distinguished Name of the EUM. It SHALL include at least 'organization' and 'commonName' attributes.
	Example of EUM DN:
	c = US
	l = New York
	o = ACME
	cn = ACME Public CA
	e = admin.pki@acme.com

Extension for Authority Key Identifier (RFC 5280 [17]): section 4.2.1.1)	extnID = id-ce-authorityKeyIdentifier
	critical = false
	extnValue = <Identifier of the public key to verify this certificate>

Extension for Key usage (RFC 5280 [17] section 4.2.1.3)	extnID = id-ce-keyUsage
	critical = true
	extnValue = {
	keyCertSign (5),
	cRLSign(6) --[Conditional] }
	cRLSign indicator SHALL be set if the EUM uses the additional EUM SubCA (see section 2.7).

Extension for Certificate Policies (RFC 5280 [17] section 4.2.1.4)	extnID = id-ce-certificatePolicies
	critical = true
	extnValue = id-rspRole-eum or id-rspRole-eum-v2 (Section 2.4a.1.0)
	To indicate that this is an EUM Certificate.
	NOTE: id-rspRole-eum-v2 indicates a Certificate in a chain following Variant O (see figure 30)

Extension for subjectAltName (RFC 5280 [17] section 4.2.1.6)	extnID = id-ce-subjectAltName
	critical = false
	extnValue = {
	registeredID (8) = EUM OID }

Extension for Basic Constraints (RFC 5280 [17] section 4.2.1.9)	extnID = id-ce-basicConstraints
	critical = true
	extnValue = {
	cA = true
	pathLenConstraint = 1 }
	To indicate that this certificate is a SubCA allowed to issue another level of EUM SubCA Certificate.
	pathLenConstraint SHALL be set to 0 when the Certificate Policies extension is set to id-rspRole-eum-v2.

Extension for CRL Distribution Points (RFC 5280 [17] section 4.2.1.13)	extnID = id-ce-cRLDistributionPoints
	critical = false
	extnValue = section 4.5.2.1.2

Extension for Name Constraints (see RFC 5280 [17] section 4.2.1.10)
(See Notes 1 and 2)
	extnID = id-ce-nameConstraints
	critical = true
	extnValue NameConstraints ::= {
	permittedSubtrees ::= {
	{
	base directoryName : rdnSequence : {
	{
	{
	type { <id-at-organizationName oid> }
	value : <organization name>
	}
	}
	}
	minimum 0
	}
	}
	}
	This restriction contains the organization name(s) that the EUM owning this certificate is allowed to set in the eUICC Certificates. This restriction applies on the subject name (containing the 'organization' attribute).
	The extension MAY contain several possible 'organization' values.
	Field 'minimum' has no meaning in this specification.

Extension for permitted EINs	extnID = id-rsp-extension-permitted-eins
	critical = false
	extnValue PermittedEins ::= {<ein1>, <ein2>,… }
	This extension contains the EINs that the EUM owning this Certificate is allowed to use in the EID of the eUICC.
	The EUM SHALL add this extension for Variants other than O.

NOTE 1: As per RFC 5280 [17], the name constraints applies to all subject names in subsequent Certificates in a certification path (i.e., EUM SubCA and EUICC certificates). NOTE 2: A former version of this specification included a restriction on the 'serialNumber' attribute in the nameConstraints extension to be one of the EUM's IINs. Variant O certificate chains maintain such restrictions for backwards compatibility. See section 4.5.2.2 for a recommendation as to how the servers can work around this issue.	

Table 12: CERT.EUM.SIG

The table below describes the specific fields of a CERT.EUMSubCA.SIG in complement of the description given in section 4.5.2.1.0.0:

Field	Value Description
subject	According to the name constraints set in CERT.EUM.SIG, the subject SHALL contain the 'organization' attribute with one of the value defined in the name constraints extension. Others attributes MAY be included for information. Example of EUM SubCA DN: o = ACME

Extension for Authority Key Identifier (RFC 5280 [17] section 4.2.1.1)	extnID = id-ce-authorityKeyIdentifier
	critical = false
	extnValue = <Identifier of the public key to verify this certificate>

Extension for Key usage (RFC 5280 [17] section 4.2.1.3)	extnID = id-ce-keyUsage
	critical = true
	extnValue = keyCertSign (5)

Extension for Certificate Policies (RFC 5280 [17] section 4.2.1.4)	extnID = id-ce-certificatePolicies
	critical = false
	extnValue = id-rspRole-eumSubCa (Annex H)
	To indicate that this is an EUM SubCA Certificate.

Extension for subjectAltName (RFC 5280 [17] section 4.2.1.6)	extnID = id-ce-subjectAltName
	critical = false
	extnValue = {
	registeredID (8) = EUM OID}
	The OID SHALL be the same as in CERT.EUM.SIG

Extension for Basic Constraints (RFC 5280 [17] section 4.2.1.9)	extnID = id-ce-basicConstraints
	critical = true
	extnValue = {
	cA = true,
	pathLenConstraint = 0 }
	To indicate that this certificate is a SubCA limited to issue only "leaf" certificate.

Extension for CRL Distribution Points (RFC 5280 [17] section 4.2.1.13)	extnID = id-ce-cRLDistributionPoints
	critical = false
	extnValue = see section 4.5.2.1.2.

Table 12a CERT.EUMSubCA.SIG

4.5.2.1.0.4 SM-DP+ SIG

The table below describes the specific fields of a CERT.DPSubCA.SIG in complement of the description given in section 4.5.2.1.0.0:

Field	Value Description

Extension for Authority Key Identifier (RFC 5280 [17] section 4.2.1.1)	extnID = id-ce-authorityKeyIdentifier
	critical = false
	extnValue = <Identifier of the public key to verify this certificate>

Extension for Key usage (RFC 5280 [17] section 4.2.1.3)	extnID = id-ce-keyUsage
	critical = true
	extnValue = {
	keyCertSign (5),
	cRLSign (6) -- [Conditional]}
	cRLSign indicator SHALL be set if the SM-DP+ SubCA manages certificate revocation (see section 2.7).

Extension for Certificate Policies (RFC 5280 [17] section 4.2.1.4)	extnID = id-ce-certificatePolicies
	critical = false
	extnValue = id-rspRole-dpSubCa (Section 2.4a.1.0)
	To indicate that this is an SM-DP+ SubCA Certificate.
	This extension MAY be absent or contain a different value when the Certificate is part of a chain following Variant AA (figure 30c).

Extension for subjectAltName (RFC 5280 [17] section 4.2.1.6)	extnID = id-ce-subjectAltName
	critical = false
	extnValue = {
	registeredID (8) = SM-DP+ OID}

Extension for Basic Constraints (RFC 5280 [17] section 4.2.1.9)	extnID = id-ce-basicConstraints
	critical = true
	extnValue = {
	cA = true,
	pathLenConstraint = 0 }
	To indicate that this certificate is a SubCA limited to issue only "leaf" certificate for the SM-DP+.

Extension for CRL Distribution Points (RFC 5280 [17] section 4.2.1.13)	extnID = id-ce-cRLDistributionPoints
	critical = false
	extnValue = section 4.5.2.1.2

Table 12b CERT.DPSubCA.SIG

The table below describes the specific fields of a CERT.DPauth.SIG/CERT.DPpb.SIG in complement of the description given in section 4.5.2.1.0.0:

Field	Value Description

subject	Distinguished Name of the SM-DP+. It SHALL include at least 'organization' and 'commonName' attributes.
	Example of an SM-DP+ DN for DPauth usage:
	c = US
	l =New York
	o = ACME
	cn = ACME DPauth
	e = dp@acme.com

Extension for Authority Key Identifier (RFC 5280 [17] section 4.2.1.1)	extnID = id-ce-authorityKeyIdentifier
	critical = false
	extnValue = <Identifier of the public key to verify this certificate>

Extension for Key usage (RFC 5280 [17] section 4.2.1.3)	extnID = id-ce-keyUsage
	critical = true
	extnValue = digitalSignature (0)

Extension for Certificate Policies (RFC 5280 [17] section 4.2.1.4)	extnID = id-ce-certificatePolicies
	critical = true
	extnValue = id-rspRole-dp-pb or id-rspRole-dp-pb-v2 (Section 2.4a.1.0) for CERT.DPpb.SIG,
	or
	extnValue = id-rspRole-dp-auth or id-rspRole-dp-auth-v2 (Section 2.4a.1.0) for CERT.DPauth.SIG,
	'extnValue' SHALL be one of the above OIDs to indicate the role of this SM-DP+ Certificate (authentication to the eUICC or Profile binding).
	NOTE: id-rspRole-dp-auth-v2 or id-rspRole-dp-pb-v2 indicates a Certificate in a chain following Variant O (see figure 30a).

Extension for subjectAltName (RFC 5280 [17] section 4.2.1.6)	extnID = id-ce-subjectAltName
	critical = false
	extnValue = {
	registeredID (8) = SM-DP+ OID}

Extension for CRL Distribution Points (RFC 5280 [17] section 4.2.1.13)	extnID = id-ce-cRLDistributionPoints
	critical = false
	extnValue = section 4.5.2.1.2
	This extension SHALL be absent for Certificates issued by SM-DP+ SubCA not managing revocation (see section 2.7).

Table 13: CERT.DPauth.SIG / CERT.DPpb.SIG

All the field values of the CERT.DPauth.SIG and CERT.DPpb.SIG SHALL be identical except for the following fields:

- subject
- serialNumber
- extension (extnValue=id-ce-certificatePolicies)
- subjectPublicKeyInfo
- signatureValue
- extension (extnID = id-ce-subjectKeyIdentifier)

4.5.2.1.0.5 SM-DP+ TLS

The table below describes the specific fields of a CERT.DP.TLS in complement of the description given in section 4.5.2.1.0.0:

Field	Value Description
subject	Refer to table defining CERT.DPauth.SIG / CERT.DPpb.SIG. Both CERT.DPauth.SIG, CERT.DPpb.SIG and CERT.DP.TLS SHALL have identical attributes except 'cn' that SHALL match one of the SM-DP+ hostname (FQDN) entries contained in the subjectAltName extension.

Extension for Authority Key Identifier (RFC 5280 [17] section 4.2.1.1)	extnID = id-ce-authorityKeyIdentifier
	critical = false
	extnValue = <Identifier of the public key to verify this certificate>

Extension for Key usage (RFC 5280 [17] section 4.2.1.3)	extnID = id-ce-keyUsage
	critical = true
	extnValue = digitalSignature (0)

Extension for Certificate Policies (RFC 5280 [17] section 4.2.1.4)	extnID = id-ce-certificatePolicies
	critical = false
	extnValue = id-rspRole-dp-tls or id-rspRole-dp-tls-v2 (Section 2.4a.1.0)
	To indicate that this is an SM-DP+ Certificate for TLS.
	This extension MAY be absent or contain a different value when the certificate is part of a Variant OO or AA (see figure 30c).
	NOTE: id-rspRole-dp-tls-v2 indicates a certificate in a chain following Variant O (see figure 30c).

Extension for Extended Key usage (RFC 5280 [17] section 4.2.1.12)	extnID = id-ce-extKeyUsage
	critical = true
	extnValue = ExtKeyUsageSyntax ::{
	{
	id-kp-serverAuth
	},
	{
	id-kp-clientAuth
	}
	}
	Certificate SHALL NOT contain any other extended key usage.

Extension for subjectAltName (RFC 5280 [17] section 4.2.1.6)	extnID = id-ce-subjectAltName
	critical = false
	extnValue = GeneralNames ::= {
	{
	dNSName '<SM-DP+ hostname (FQDN) value>'
	}
	{
	registeredID '<SM-DP+ OID value>'
	}
	}
	This extension SHALL contain at least one dNSName entry, and only one registeredID entry.
	A dNSName entry MAY contain the wildcard character * as allowed by RFC 2818 [67].

Extension for CRL Distribution Points (RFC 5280 [17] section 4.2.1.13)	extnID = id-ce-cRLDistributionPoints
	critical = false
	extnValue = section 4.5.2.1.2
	This extension SHALL be absent for Certificates issued by SM-DP+ SubCA not managing revocation (see section 2.7).

Table 14: CERT.DP.TLS

4.5.2.1.0.6 SM-DS SIG

The table below describes the specific fields of a CERT.DSSubCA.SIG in complement of the description given in section 4.5.2.1.0.0:

Field	Value Description

Extension for Authority Key Identifier (RFC 5280 [17] section 4.2.1.1)	extnID = id-ce-authorityKeyIdentifier
	critical = false
	extnValue = <Identifier of the public key to verify this certificate>

Extension for Key usage (RFC 5280 [17] section 4.2.1.3)	extnID = id-ce-keyUsage
	critical = true
	extnValue = {
	keyCertSign (5),
	cRLSign (6) -- [Conditional]}
	cRLSign indicator SHALL be set if the if the SM-DS SubCA manages certificate revocation (see section 2.7).

Extension for Certificate Policies (RFC 5280 [17] section 4.2.1.4)	extnID = id-ce-certificatePolicies
	critical = false
	extnValue = id-rspRole-dsSubCa (Section 2.4a.1.0)
	To indicate that this is an SM-DS SubCA Certificate.
	This extension MAY be absent or contain a different value when the Certificate is a chain following Variant AA (figure 30c).

Extension for subjectAltName (RFC 5280 [17] section 4.2.1.6)	extnID = id-ce-subjectAltName
	critical = false
	extnValue = {
	registeredID (8) = SM-DS OID}

Extension for Basic Constraints (RFC 5280 [17] section 4.2.1.9)	extnID = id-ce-basicConstraints
	critical = true
	extnValue = {
	cA = true,
	pathLenConstraint = 0 }
	To indicate that this certificate is a SubCA limited to issue only "leaf" certificate for the SM-DS.

Extension for CRL Distribution Points (RFC 5280 [17] section 4.2.1.13)	extnID = id-ce-cRLDistributionPoints
	critical = false
	extnValue = section 4.5.2.1.2

Table 14a CERT.DSSubCA.SIG

The table below describes the specific fields of a CERT.DSauth.SIG in complement of the description given in section 4.5.2.1.0.0:

Field	Value Description

subject	Distinguished Name of the SM-DS. It SHALL include at least 'organization' and 'commonName' attributes.
	Example of an SM-DS DN:
	c = US
	l = New York
	o = ACME
	cn = ACME SM-DS
	e = ds@acme.com

Extension for Authority Key Identifier (RFC 5280 [17] section 4.2.1.1)	extnID = id-ce-authorityKeyIdentifier
	critical = false
	extnValue = <Identifier of the public key to verify this certificate>
	To identify the PK.CI.SIG that has to be used to verify this certificate.

Extension for Key usage (RFC 5280 [17] section 4.2.1.3)	extnID = id-ce-keyUsage
	critical = true
	extnValue = digitalSignature (0)

Extension for Certificate Policies (RFC 5280 [17] section 4.2.1.4)	extnID = id-ce-certificatePolicies
	critical = true
	extnValue = id-rspRole-ds-auth or id-rspRole-ds-auth-v2 (Section 2.4a.1.0)
	To indicate that this is an SM-DS SIG Certificate for authentication to the eUICC.
	NOTE: id-rspRole-ds-auth-v2 indicates a Certificate in a chain following Variant O (see figure 30a).

Extension for subjectAltName (RFC 5280 [17] section 4.2.1.6)	extnID = id-ce-subjectAltName
	critical = false
	extnValue = {
	registeredID (8) = SM-DS OID}

Extension for CRL Distribution Points (RFC 5280 [17] section 4.2.1.13)	extnID = id-ce-cRLDistributionPoints
	critical = false
	extnValue = section 4.5.2.1.2
	This extension SHALL be absent for Certificates issued by SM-DS SubCA not managing revocation (see section 2.7).

Table 15: CERT.DSauth.SIG

4.5.2.1.0.7 SM-DS TLS

The table below describes the specific fields of a CERT.DS.TLS in complement of the description given in section 4.5.2.1.0.0:

Field	Value Description
subject	Refer to table defining CERT.DSauth.SIG. Both CERT.DSauth.SIG and CERT.DS.TLS SHALL have identical attributes except 'cn' that SHALL match one of the SM-DS hostname (FQDN) entries contained in the subjectAltName extension.

Extension for Authority Key Identifier (RFC 5280 [17] section 4.2.1.1)	extnID = id-ce-authorityKeyIdentifier
	critical = false
	extnValue = <Identifier of the public key to verify this certificate>

Extension for Key usage (RFC 5280 [17] section 4.2.1.3)	extnID = id-ce-keyUsage
	critical = true
	extnValue = digitalSignature (0)

Extension for Certificate Policies (RFC 5280 [17] section 4.2.1.4)	extnID = id-ce-certificatePolicies
	critical = false
	extnValue = id-rspRole-ds-tls or id-rspRole-ds-tls-v2 (Section 2.4a.1.0)
	To indicate that this an SM-DS TLS Certificate.
	This extension MAY be absent or contain a different value when the certificate is part of a Variant OO or AA (see figure 30c).
	NOTE: id-rspRole-ds-tls-v2 indicates a Certificate in a chain following Variant O (see figure 30c).

Extension for Extended Key usage (RFC 5280 [17] section 4.2.1.12)	extnID = id-ce-extKeyUsage
	critical = true
	extnValue = ExtKeyUsageSyntax ::{
	{
	id-kp-serverAuth
	},
	{
	id-kp-clientAuth
	}
	}
	Certificate SHALL NOT contain any other extended key usage.

Extension for subjectAltName (RFC 5280 [17] section 4.2.1.6)	extnID = id-ce-subjectAltName
	critical = false
	extnValue = GeneralNames ::= {
	{
	dNSName '<SM-DS hostname (FQDN) value>'
	},
	{
	registeredID '<SM-DS OID value>'
	}
	}
	This extension SHALL contain at least one dNSName entry, and only one registeredID entry.
	A dNSName entry MAY contain the wildcard character * as allowed by RFC 2818 [67].

Extension for CRL Distribution Points (RFC 5280 [17] section 4.2.1.13)	extnID = id-ce-cRLDistributionPoints
	critical = false
	extnValue = section 4.5.2.1.2
	This extension SHALL be absent for Certificates issued by SM-DS SubCA not managing revocation (see section 2.7).

Table 16: CERT.DS.TLS

4.5.2.1.1 Algorithm Identifiers and Parameters

This section provides the values to be set in 'AlgorithmIdentifier.algorithm' and 'AlgorithmIdentifier.parameters' fields of the certificate for each of the algorithms used in this specification.

For section 'subjectPublicKeyInfo' the following settings SHALL apply:

'AlgorithmIdentifier.algorithm' field SHALL be set to: "iso(1) member-body(2) us(840) ansi-X9-62(10045) keyType(2) ecPublicKey(1)" as defined in RFC 5480, or

"iso(1) standard(0) digital-signature-with-appendix(14888) part3(3) algorithm(0) sm2(14)" as defined in ISO 14888-3.

'AlgorithmIdentifier.parameters' field SHALL be set to:

- for BrainpoolP256r1: "iso(1) identified-organization(3) teletrust(36) algorithm(3) signatureAlgorithm(3) ecSign(2) ecStdCurvesAndGeneration(8) ellipticCurve(1) versionOne(1) brainpoolP256r1(7)" as defined in RFC 5639 [18]
- for NIST P-256: "iso(1) member-body(2) us(840) ansi-X9-62(10045) curves(3) prime(1) prime256v1(7)" as defined in RFC 5480 [27]
- For FRP256V1: "iso(1) member-body(2) fr(250) type-org(1) 223 101 256 1" as defined in ANSSI ECC [20]
- For SM2 public keys: 'AlgorithmIdentifier.parameters' field SHALL be omitted

For sections 'signature' and 'signatureAlgorithm' the following settings SHALL apply:

- 'AlgorithmIdentifier.algorithm' field SHALL be set as following for ECDSA with SHA256:
  - "iso(1) member-body(2) us(840) ansi-X9-62(10045) signatures(4) ecdsa-with-SHA2(3) ecdsa-with-SHA256(2)" as defined in RFC 5758 [25] and RFC 5759 [26].
  - 'AlgorithmIdentifier.parameters' field SHALL be omitted as defined in RFC 5758 [25] section 3.2.
- 'AlgorithmIdentifier.algorithm' field SHALL be set as following for SM2:
  - "iso(1) standard(0) digital-signature-with-appendix(14888) part3(3) algorithm(0) sm2(14)" as defined in ISO 14888-3.
  - 'AlgorithmIdentifier.parameters' field SHALL be omitted.

4.5.2.1.2 Extension CRL Distribution Point

A CRL Issuer (see section 2.7) SHALL include the `cRLDistributionPoints` extension in the Certificates they issue.

The `cRLDistributionPoints` extension in a certificate indicates the location(s) where a CRL can be retrieved, which lists this certificate upon its revocation.

The extension MAY contain several `DistributionPoint` entries. Nevertheless it is recommended to set only one entry in the Certificates part of a server certificate chain in order to limit their size, because of the eUICC constrained resources.

If an eSIM CA supports the issuance of version 2 Certificates, it SHALL also support the generation of the segmented CRL form as defined in version 2 of this specification.

A `DistributionPoint` entry SHALL only have the `distributionPoint` field set. The optional `reasons` field SHALL NOT be present; each revoked certificate SHALL have its own reason set. The `cRLIssuer` field SHALL NOT be present, because the CRL SHALL be issued by the Certificate Issuer.

The `distributionPoint` field MAY contain several general names, each describing a different mechanism to obtain the same CRL (the field `nameRelativeToCRLIssuer` is not used in this specification). But `distributionPoint` SHALL contain at least a general name of type URI with an HTTP scheme, indicating that the CRL can be retrieved as an HTTP resource.

4.5.2.2 Certificate Verification

Certificate verifiers SHALL only accept certificate chains defined in section 4.5.2.0a and 4.5.2.0b. They SHALL verify the Certificate according to RFC 5280 [17] with the exception given below.

The SM-DP+ and SM-DS SHALL verify that the EID in the eUICC Certificate CERT.EUICC.SIG is consistent with the EINs permitted in the EUM Certificate CERT.EUM.SIG (see section 4.5.2.0.2). For Variant O, the list of permitted EINs is given in the nameConstraints extension; for all other variants, it is given in the separate extension Permitted EINs.

For compatibility reasons, and as an exception to the verification requirements specified by RFC 5280 [17], the SM-DP+ and SM-DS SHALL NOT enforce the 'serialNumber' part of the nameConstraints extension when verifying the eUICC Certificate in Variant O.

In addition, verifiers of certificate chains for TLS establishment MAY verify restrictions to Certificate Profiles specified in section 4.5.2.1.0 for these Certificates. Restrictions to Certificate Profiles defined in section 4.5.2.1.0 for other Certificate chains SHALL be verified.

Certificate revocation status of each Certificate in the chain SHALL be verified:

- RSP Servers SHALL perform this verification as described in RFC 5280 [17], using the CRLs that they can retrieve as defined in section 4.6.4.
- The LPAd SHALL perform this verification for TLS Certificates using a revocation mechanism supported by the certificate issuers in the trust chain (e.g., CRL validation as described in RFC 5280 [17]).
- The eUICCs SHALL perform this verification as indicated in ES10b.AuthenticateServer, section 5.7.13.

NOTE: Verification of the revocation status of TLS Certificates by the LPAe is FFS.

If any of these verifications fail, the certificate SHALL be considered as invalid and the operation for which it was used, SHALL be rejected.

4.6 Certificate Revocation List

The CRL can be empty or contain one or several revoked certificates from among all the unexpired certificates in scope of the CRL.

The CRL SHALL follow RFC 5280 [17] with the specific coding and rules given in this section. The CRL SHALL have all the extensions described in table 17. The CRL MAY have additional extensions except `deltaCRLIndicator` and `freshestCRL` (Delta CRLs are not used).

A certificate listed in a CRL SHALL be considered as definitively revoked (i.e., the 'Hold' state is not considered).

NOTE: CRL is described using table representation for easiness, but conforms to the ASN.1 format given in RFC 5280 [17].

Field	Value Description	
tbsCertList	Data to be signed	
	Field	**Value Description**
	version	Version SHALL be 2.
	signature	Contains the algorithm identifier and parameters used by the issuer to compute the value of the field 'signatureValue' (section 4.5.2.1.1). The algorithm identifier value and parameters values SHALL be the same as the one of the field 'signatureAlgorithm'.
	issuer	Distinguished Name of the CRL Issuer. This DN SHALL be the same as the one used for issuing certificates (i.e., DN set in the `issuer` field of the Certificate).
	thisUpdate	Time stamp of the CRL.
	nextUpdate	Indicates the date by which the next CRL will be issued. The next CRL could be issued before the indicated date, but it will not be issued any later than the indicated date.
	revokedCertificates	Sequence of "revocation entry" as described in table 18.

	crlExtensions	Extension for Authority Key Identifier (RFC 5280 [17] section 5.2.1):
		extnID = id-ce-authorityKeyIdentifier
		critical = true
		extnValue = <Identifier of the public key to verify this CRL>

		Extension for CRL Number (RFC 5280 [17] section 5.2.3):
		extnID = id-ce-cRLNumber
		critical = false
		extnValue = CRL number, starting from 0.

		Extension for Issuing Distribution Point (RFC 5280 [17] section 5.2.5):
		extnID = id-ce-issuingDistributionPoint
		critical = true
		extnValue = {
		  distributionPoint [0] -- section 4.5.2.1.2
		}
		Fields indirectCRL and onlyContainsAttributeCerts SHALL be set to FALSE. onlySomeReasons SHALL be omitted.

signatureAlgorithm	Section 4.5.2.1.1	

signatureValue	Signature computed accordingly to one of the possible algorithm listed in signatureAlgorithm field

Table 17: CRL Description

Field	Value Description
userCertificate	Serial number of the revoked certificate
revocationDate	Date of revocation

crlEntryExtensions	Extension for Reason code(1) (RFC 5280 [17] section 5.3.1):
	extnID = id-ce-cRLReasons
	critical = false
	extnValue = a reason code

NOTE: A version 3 RSP entity SHALL ignore a Certificate Expiration Date extension, as defined in previous versions of this specification, which may be present in a CRL entry.	

Table 18: Revocation entry

4.6.1 CRL publication rules

Each CRL Issuer SHALL issue a new CRL 1) no later than the `nextUpdate` date indicated in the previous CRL with the same scope (even if no new revocation has occurred during the period), and 2) whenever at least one additional certificate is revoked.

NOTE: The publication periodicity is not defined in this document. This SHALL be defined in GSMA eUICC PKI Certificate Policy [45].

The SM-DP+, SM-DS or any relying party SHALL ensure it uses the up-to-date CRL(s) required to perform the on-going transaction.

The CRL and Certificates that are in scope of the CRL SHALL be signed with the same key; that allows to limit the amount of data to be passed to the eUICC during the Common Mutual Authentication procedure (see section 3.0.1). It implicitly means that the CRL and these Certificates have the same issuer.

In order to limit the maximum size of a CRL, the CRL Issuer SHOULD limit the scope of their CRL(s). This is beneficial to the overall performance of the system where the CRL has to be provided to the eUICC. However, this specification doesn't mandate any particular method for defining the CRL scope. E.g., an eSIM CA SubCA may define a different scope for each type of Certificate it issues: one scope for CERT.DSauth.SIG, one for CERT.DS.TLS, one for CERT.DPauth.SIG; a (Sub)CA may limit the number of Certificates per scope.

The CRL base list SHALL be complete (i.e., contain all previously revoked certificates, plus the newly revoked certificates). Delta CRLs as defined in RFC 5280 [17] SHALL NOT be used.

The CRL Issuers SHALL manage the `cRLNumber` extension according to RFC 5280 [17]. This number SHALL be incremented by one at each new CRL publication.

4.6.2 Void

4.6.3 eUICC Considerations

The eUICC faces a general issue regarding time management and CRL. It does not have a time reference internally and can only rely on time provided from an off-card entity (with the question on reliability of this information).

This section provides the rule for the eUICC to address this concern:

- The eUICC derives a lower time reference indicating "it must be later than" from the CRLs which are signed by the eSIM CA(s). These are considered reliable and are in addition subject to consistency checks by the SM-DP+ (mandatory) and the LPA (optional), see section 3.0.1.
- The eUICC validates the time information of all certificates and CRLs: As each of these provides a time window within which they are valid, and assuming all are neither expired nor not yet valid, "now" must be in the overlap of all these time windows. Plus at least a part of this overlap must be later than the lower time reference on the eUICC.

4.6.3.1 Lower time reference on the eUICC

The eUICC SHALL maintain a lower time reference (LTR), which is used for certificate checks. An initial value SHALL be set during production. A unique value SHALL be used across all eSIM CAs.

The CRL 'tbsCertList.thisUpdate' field contains a reference for a time that has already passed and is considered to be reliable if it is signed by an eSIM CA. Both CRLs signed by an eSIM CA RootCA and those signed by an eSIM CA SubCA SHALL be taken into account. Once the signature of such a CRL has been verified by the eUICC and 'tbsCertList.thisUpdate' is higher than LTR, LTR SHALL be updated to 'tbsCertList.thisUpdate'. This SHALL be done independently of the result of the time checks specified in the next section.

An eUICC supporting OS update SHOULD also provide a secure mechanism to reset LTR.

4.6.3.2 Time checks on the eUICC

The eUICC SHALL check the validity of the time information provided by all Certificates and CRLs of one transaction by performing the following calculations:

- TWL (time window low) is set to the highest value of the 'tbsCertList.thisUpdate' values from all CRLs, the 'tbsCertificate.validity.notBefore' values from all Certificates, and LTR.
- TWH (time window high) is set to the lowest value of the 'tbsCertList.nextUpdate' values from all CRLs, and the 'tbsCertificate.validity.notAfter' values from all Certificates.
- If TWH is higher or equal to TWL, the time check is successful. Otherwise the check fails, because at least one item is either expired or not yet valid, resulting in a negative time window.

4.6.4 Retrieving a CRL

A CRL can be retrieved by navigating the URI indicated in a distribution point entry of the `cRLDistributionPoints` extension of any Certificate in its scope. When using the HTTP scheme the CRL SHALL be returned in the HTTP response body as a DER encoded `CertificateList` data object which in turn is Base64 encoded, as defined in RFC 5280 [17] section 4.2.1.13. At least one distribution point entry SHALL contain an URI with the HTTP scheme, see section 4.5.2.1.2.

4.7 Confirmation Code

A Profile download order and/or Device Change of the Profile MAY be protected by a specific Confirmation Code. The Confirmation Code is provided by the Operator to the SM-DP+ and the End User during the Profile download initiation procedure (section 3.1.1) or the Device Change procedure (section 3.11.1). The means by which the Confirmation Code is provided to the End User is out of scope of this specification.

In case of the Profile Download order, during the Profile download and installation procedure (section 3.1.3), if the Profile download order is protected by a Confirmation Code, the SM-DP+ SHALL verify that the Confirmation Code provided by the End User matches the Confirmation Code provided by the Operator.

In case of the Device Change, during the Device Change procedure (section 3.11.1), if the Device Change is protected by a Confirmation Code, the SM-DP+ SHALL verify that the Confirmation Code provided by the LPAd of the old Device matches the Confirmation Code provided by the Service Provider.

In addition, the SM-DP+ SHALL protect against excessive incorrect entries of the Confirmation Code. The maximum number of incorrect Confirmation Code attempts allowed to a Profile download order or a Device Change of the Profile is defined by the Operator and communicated to the SM-DP+ by means out of scope of this specification.

Once the maximum number of incorrect Confirmation Code attempts is exceeded for a Profile download order, the Profile download order or the Device Change of the Profile SHALL be terminated and the SM-DP+ SHALL communicate the final status to the Operator. The Operator is free to request a new Profile download order corresponding to the same Profile, with the same or a different Confirmation Code.

4.8 Device Information Code

The Device Information Code (DEV-IC) is a set of Device and eUICC-related information fields including EID, SM-DS address, etc. The DEV-IC SHALL be coded as a concatenation of the strings listed in Table X1 using a URI format as defined in RFC 3986 [72]:

Name	Description	MOC
Scheme	SHALL be set to either "eid" or "EID".	M
Delimiter	SHALL be set to ":".	M
Path	EID of the eUICC, the numeric text representation SHALL comprise 32 digits, where each digit is represented by one character in the set [0123456789].	M

Name	Description	MOC
Delimiter	SHALL be present and set to "?" if any of the following query components is present.	C
Query	Additional Device-related information in the form of "key=value" pairs.	O

Table X1: Device Information Code fields

Each key/value pair is defined in Table Y1, and SHALL be concatenated by using "&" as a delimiter if there are more than one key/value pair in the DEV-IC. Any of keys can appear more than once in any order. Additional proprietary information MAY be included in the DEV-IC by using a key starting with "x-".

Key	Value	MOC
aeid	EID of an additional eUICC, the numeric text representation SHALL comprise 32 digits, where each digit is represented by one character in the set [0123456789].	O
tac	TAC of the Device, the numeric text representation SHALL comprise 8 digits, where each digit is represented by one character in the set [0123456789].	O
imei	IMEI of the Device, the numeric text representation SHALL comprise 15 digits, where each digit is represented by one character in the set [0123456789].	O
ds	FQDN (Fully Qualified Domain Name) of an Root SM-DS (e.g.,MYDS.COM) stored on the eUICC indicated in the Path field of the DEV-IC or stored within the Device.	O

Table Y1: Query component in DEV-IC

If the DEV-IC contains multiple EIDs, the EID value in the Path field SHALL be used for SM-DS Event Registration by using an SM-DS address(es) in the query component.

The DEV-IC SHOULD be represented in a case-sensitive text string restricted to Byte mode character set defined in table 6 of ISO/IEC 18004 [15] and the equivalent QR code. The QR code representation SHALL be encoded according to ISO/IEC 18004 [15].

Examples of the DEV-IC are as follows:

- EID:89001012012341234012345678901224
  (if only one EID is present)
- EID:89001012012341234012345678901224?aeid=89001567010203040506070809101152
  (if multiple EIDs are present)
- EID: 89001012012341234012345678901224?tac=35123451&tac=35123452
  (if an EID and two TACs are present)
- EID:89001012012341234012345678901224?imei=351234510000011&imei=351234520000029
  (if an EID and two IMEIs are present)
- EID:89001012012341234012345678901224?ds=MYDS-1.COM&ds=MYDS-2.COM
  (if an EID and SM-DS addresses are present)

- EID:89001012012341234012345678901224?aeid=89001567010203040506070809101152&ds=MYDS-1.COM&ds=MYDS-2.COM
  (if multiple EIDs and SM-DS addresses are present)

5 Functions

This section specifies the Functions associated with the Remote SIM Provisioning and Management of the eUICC for consumer Devices.

General rules for the function description:

- Wherever it says in the description of the steps to be taken within a function that the function provider "… SHALL return an error …", this implies that processing of the function ends at this point.
- A function provider is not mandated to perform the checks in the exact sequence of the description. This implies that different error codes may result from error situations that affect more than one verification step.

5.1 Overview of Functions per Interface

Provides the description of the interfaces and functions within the Remote SIM Provisioning and Management system involving the eUICC, including the following:

eUICC Interfaces

- ES6: The interface used by the Operator to manage the content of their Profile.
- ES8+: 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.
- ES10a: Used by the LPAd to get the configured addresses from the eUICC for Root SM-DS(s), and optionally the Default SM-DP+.
- ES10b: Used by the LPAd to transfer a Profile Package or RPM Package to the eUICC, and to perform Device Change or Profile Recovery.
- ES10c: Used by the LPAd for local End User management of Profiles installed on the eUICC (e.g., Enable, Disable, Delete).
- ES25: Used between the UIMe and the LUIe to transfer End User related interaction.

Server to Server Interfaces

- ES2+: Interface between the Operator and the SM-DP+ used by the Operator to order Profile Package preparation and to receive notifications, and by the SM-DP+ to get a confirmation for the Device Change of the Profile from the Operator.
- ES12: Interface between the SM-DP+ and an SM-DS (Alternative SM-DS or Root SM-DS) for the Event management.
- ES15: Interface between an Alternative SM-DS and a Root SM-DS for the Event management.

LPA to Server Interfaces

- ES9+: Used to provide a secure transport between the SM-DP+ and the LPA for the delivery of the Profile Package, the delivery of the RPM package and sending the notifications.
- ES11: Interface between the LPA and an SM-DS (Alternative SM-DS or Root SM-DS) for the Event retrieval, Event Checking and Push Service registration.

- EShri: Interface between the LPA and a server providing the High Resolution Icons for a Profile.

Profile Content Management Interfaces

- ES20: Interface between the PCMP and PCMAA for managing the contents of an Enabled Profile.
- ESaa: Interface between the PCMAA and the eUICC for managing the contents of an Enabled Profile.

These correspond to the Administrative Agent interfaces defined in GP SERAM [74].

Device Internal Interfaces

- ES21: Interface between a Device Application and the LPRd for the LPR session.

ESop (interface between the End User and the Operator), ESeum (Interface between the EUM and the eUICC) and ESeu (Interface between the End User and the LUI) are out of scope of this document.

The following table presents the normative list of all the functions that are defined in this section.

Request-Response Functions:

Interface	Functions	Function provider role	Feature support (*)
ES2+	DownloadOrder ConfirmOrder CancelOrder ReleaseProfile	SM-DP+	
	RpmOrder		#SupportedFromV3.0.0#
	HandleDeviceChangeRequest	Operator	#SupportedForDcV3.0.0#
ES6	UpdateMetadata	ISD-P	

ES8+	InitialiseSecureChannel ReplaceSessionKeys ConfigureISDP StoreMetadata LoadProfileElements	ISD-R/ISD-P	
ES9+	InitiateAuthentication GetBoundProfilePackage AuthenticateClient CancelSession	SM-DP+	
	ConfirmDeviceChange		#SupportedForDcV3.0.0#
ES10a	GetEuiccConfiguredData SetDefaultDpAddress	ISD-R (LPA Services)	
	VerifySmdsResponse		#SupportedFromV3.0.0#
ES10b	GetEUICCChallenge GetEUICCInfo PrepareDownload LoadBoundProfilePackage ListNotification RetrieveNotificationsList RemoveNotificationFromList AuthenticateServer CancelSession GetRAT	ISD-R (LPA Services)	
	LoadCRL		#SupportedOnlyBeforeV3.0.0#
	LoadRpmPackage		#SupportedForRpm#
	PrepareDeviceChange VerifyDeviceChange		#SupportedForDcV3.0.0#
ES10c	GetProfilesInfo EnableProfile DisableProfile DeleteProfile eUICCMemoryReset GetEID SetNickname	ISD-R (LPA Services)	
	LpaAlerting	LPAd	#SupportedFromV3.0.0#
ES11	InitiateAuthentication AuthenticateClient	SM-DS	
	CheckEvent		#SupportedForEventCheckingV3.0.0#
ES12	RegisterEvent DeleteEvent	SM-DS	

ES15	RegisterEvent DeleteEvent	SM-DS	
ES21	InitiateProfileContentManagement PcmProgressInformation	LPRd	#SupportedFromV3.0.0#
EShri	RetrieveIcon	HRI Server	#SupportedFromV3.0.0#

Table 19: Request-Response Functions

(*) If empty, the relevant function exists since version 2.

Notification Handler Functions:

Interface	Notification handler functions	Function handler/recipient
ES2+	HandleNotification	Operator
ES9+	HandleNotification	SM-DP+

Table 20: Notification Handler Functions

5.2 LPA to Server and Server to Server Function Commonalities

Each function represents an entry point that is provided by a Role (function provider) and that can be called by other Roles (function requester).

5.2.1 Common Data Types

The functions provided in this section deal with management of the eUICC and Profile, so that the common data defined in this section needs to be used in most of the functions.

Type name	Description	Type definition
Hexadecimal String	String of even length composed of characters between '0' to '9' and 'A' to 'F' or 'a' to 'f'.	
AID	The AID (Application Identifier) of an Executable Load File, an Executable Module, a Security Domain, or an Application.	Hexadecimal string representation of 5 to 16 bytes.

DATETIME	Any date and time used within any interface of this specification.	String format as specified by W3C: YYYY-MM-DDThh:mm:ssTZD Where: YYYY = four-digit year MM = two-digit month (01=Jan, etc.) DD = two-digit day of month (01-31) hh = two digits of hour (00 -23) mm = two digits of minute (00 - 59) ss = two digits of second (00 - 59) TZD = time zone designator (Z, +hh:mm or -hh:mm) Ex: 2001-12-17T09:30:47Z
EID	The EID type is for representing an eUICC-ID. An eUICC-ID is primarily used in the "Embedded UICC Remote Provisioning and Management System" to identify an eUICC. See SGP.02 [2] section 5.2.4.6 for EID description. This value SHALL NOT be modified during the lifetime of the eUICC.	String of 32 decimal characters.
FQDN	The FQDN type is for representing a Fully Qualified Domain Name (e.g., smdp.example.com).	String, as a list of domain labels concatenated using the full stop (dot, period) character as separator between labels. Labels are restricted to the Alphanumeric mode character set defined in table 5 of ISO/IEC 18004 [15] and the lower-case equivalents of the alphabetic characters within that set.
FQDN-Ext	The FQDN-extended type is either a FQDN or a designated alternative string as defined in this specification (e.g., '.unspecified').	String, as defined for the FQDN type, possibly starting with a full stop character (i.e., '.') .
ICCID	The ICCID type is for representing an ICCID (Integrated Circuit Card Identifier). The ICCID is primarily used to identify a Profile. ICCID is defined according to ITU-T recommendation E.118 [21]. However, the ICCID SHALL either consist of 19 or 20 digits.	String representation of 19 or 20 digits, where the 20th digit MAY optionally be the padding character F. Ex: 89470100001234567 84F A 19 digit ICCID with and without padding SHALL identify the same Profile. NOTE: In some markets, the ICCID may include hexadecimal digits and may omit the check digit.
OID	An Object Identifier.	String representation of an OID, i.e., of integers separated with dots (e.g.: '1.2', '3.4.5').

VERSION	The Version type is for indicating a version of any entity used within this specification. A version is defined by its major, minor and revision number.	String representation of three integers separated with dots (e.g.: '1.15.3').

Table 21: Common data types

5.2.2 Request-Response Function

As defined in SGP.02 [2].

5.2.3 Notification Handler Function

As defined in SGP.02 [2].

5.2.4 Functions Input Header

As defined in SGP.02 [2] subject to the following constraints:

- The field `Validity Period` SHALL NOT be present in Functions Input Headers.
- The field `Function Requester Identifier` SHALL contain the string representation of the function requester identity. For ES12/ES15, it MAY be the OID contained in the `subjectAltName` field of the certificate used by the function requester for its authentication, an OID in the sub-tree of this value, or any other value. It MAY also be the identity of an entity on behalf of which the function requester operates.
  This information MAY be used by the function provider for the purpose of authorisation, accounting and billing.

5.2.5 Functions Output Header

As defined in SGP.02 [2] subject to the following constraint:

- The fields `Processing Start`, `Processing End` and `Acceptable Validity Period` SHALL NOT be present in Functions Output Headers.
- The `Executed-WithWarning` and `Expired` values SHALL NOT be used in Function Execution Status field in Functions Output Headers.

5.2.6 Status Code

This specification relies on subject codes and reason codes as defined in SGP.02 [2]. In addition this specification defines the additional codes.

5.2.6.1 Subject codes

Hereunder are listed the subject codes used in this specification:

1. Generic (as defined in SGP.02 [2])

    1.1. Function Requester (as defined in SGP.02 [2])

    1.2. Function Provider (as defined in SGP.02 [2])

    1.3. Protocol (as defined in SGP.02 [2])

        1.3.1. Protocol Format (as defined in SGP.02 [2])

1.3.2. Protocol Version (as defined in SGP.02 [2])

1.6. Function (as defined in SGP.02 [2])

8. eUICC Remote Provisioning (as defined in SGP.02 [2])

8.1. eUICC (as defined in SGP.02 [2])

8.1.1. EID (as defined in SGP.02 [2])

8.1.2. EUM Certificate

8.1.3. eUICC Certificate

8.2. Profile (as defined in SGP.02 [2])

8.2.1. Profile ICCID (as defined in SGP.02 [2])

8.2.5. Profile Type (as defined in SGP.02 [2])

8.2.6. Matching ID

8.2.7. Confirmation Code

8.2.8. PPR

8.2.9. Profile Metadata

8.2.10. Bound Profile Package

8.2.11. Managing SM-DP+

8.2.12. Profile Owner

8.2.13. Enterprise

8.2.14. LPA Proxy

8.8. SM-DP+

8.8.1. SM-DP+ Address

8.8.2. Security configuration

8.8.3. Specification Version Number (SVN)

8.8.4. SM-DP+ Certificate

8.8.5. Download order

8.8.6. RPM Order

8.9. SM-DS

8.9.1. SM-DS Address

8.9.2. Security configuration

8.9.3. Specification Version Number (SVN)

8.9.4. SM-DS Certificate

8.9.5. Event Record

8.9.6. ECID

8.9.7. Push Service

8.10. RSP Operation

8.10.1. TransactionId

8.10.2. RPM Script

8.10.3. Device Change

8.10.4. Event Checking

8.10.6. Push Service Registration

8.11. eSIM CA

8.11.1. Public Key (PK)

8.12. LPA

5.2.6.2 Reason codes

Hereunder are listed the reason codes used in this specification:

1. Access Error (as defined in SGP.02 [2])

1.1. Unknown (Identification or Authentication) (as defined in SGP.02 [2])

1.2. Not Allowed (Authorisation) (as defined in SGP.02 [2])

2. Format Error (as defined in SGP.02 [2])

2.1. Invalid (as defined in SGP.02 [2])

2.2. Mandatory Element Missing (as defined in SGP.02 [2])

2.3. Conditional Element Missing (as defined in SGP.02 [2])

3. Conditions of Use Not Satisfied (as defined in SGP.02 [2])

3.1. Unsupported (as defined in SGP.02 [2])

3.3. Already in Use (Uniqueness) (as defined in SGP.02 [2])

3.7. Unavailable (as defined in SGP.02 [2])

3.8. Refused (as defined in SGP.02 [2])

3.9. Unknown (as defined in SGP.02 [2])

3.10. Invalid Association

3.11. Value has Changed

3.12. Invalid Match

4. Processing Error (as defined in SGP.02 [2])

4.2. Execution Error (as defined in SGP.02 [2])

4.3. Stopped on Warning (as defined in SGP.02 [2])

4.8. Insufficient Memory (as defined in SGP.02 [2])

4.10. Time to Live Expired

5. Transport Error (as defined in SGP.02 [2])

5.1. Inaccessible (as defined in SGP.02 [2])

6. Security Error (as defined in SGP.02 [2])

6.1. Verification Failed (as defined in SGP.02 [2])

6.3. Expired

6.4. Maximum number of attempts exceeded

5.2.6.3 Common Function Status Code

As defined in SGP.02 [2].

5.3 ES2+ (Operator -- SM-DP+)

The ES2+ interface is used by the Operator to order and manage RSP operations for specific eUICC(s).

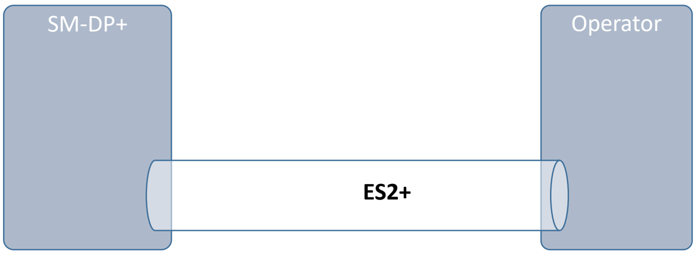


Figure 31: ES2+

The Operator communicates with the SM-DP+ through a secure connection. The level of security requested on this interface and the level of data encryption is defined in GSMA SAS SM specification [23].

The SM-DP+ SHOULD be able to support a framework of interchange and execution prioritisation for transactions over the ES2+ interface. For the purpose of assigning priority, there SHOULD be a categorisation of transactions passing over the ES2+ interface, the Operator SHOULD be able to preset the priority, and the SM-DP+ SHOULD handle the transactions according to this prioritisation. In particular, the SM-DP+ SHOULD ensure that transactions marked with the same priority are handled on a first in first out (FIFO) basis.

On an implementation that handles queues for Notifications, the SM-DP+ SHALL be able to prioritise as required by the Profile Owner the exchanges required on the ES2+ to manage the delivery of the Profiles regarding the Notifications issued on the ES2+ interface.

A Profile Owner SHALL be able to query about:

- The current state or state history of a Profile identified by ICCID, or a not empty MatchingID.
- The current state or state history of a one of more Profiles when queried by EID or empty MatchingID.

The possible Profile states are defined in section 3.1.6.

The following sections describe a standardised set of operations between the Operator and the SM-DP+. However, for a specific implementation, both parties MAY agree on exchanging only a subset of the standardised messages. The SM-DP+ MAY also perform additional functions, which are out of scope of this specification.

5.3.1 Function: DownloadOrder

Related Procedures: Download initiation

Function Provider Entity: SM-DP+

Description:

This function is used to instruct the SM-DP+ of a new Profile download request.

The EID is optional and MAY not be known at this stage. If the EID is known, the SM-DP+, with the Operator, MAY verify if the EID is compatible with the requested Profile Type (see also Annex F). If an SM-DS or Default SM-DP+ is to be used for the Profile download, then the EID SHOULD be present; if not present, the EID SHALL be provided later in "ES2+.ConfirmOrder".

Upon reception of this function call, the SM-DP+ SHALL:

- Reserve an ICCID in its inventory. If the ICCID was provided as input data, the reservation SHALL use this value. Otherwise, the reservation SHALL be done corresponding to the requested Profile Type with a value available in the SM-DP+'s inventory.
- Optionally, if not already done, the SM-DP+ performs the 'Profile generation' and 'Profile protection' steps, as described in section 2.5.3, for the Profile identified by its ICCID.

- If the EID is known, the ICCID is linked to this EID and the Profile state SHALL be set to "Linked". Otherwise, the Profile state SHALL be set to "Allocated".

The SM-DP+ MAY perform additional operations, which are out of scope of this specification.

This function SHALL return one of the following:

- A 'Function execution status' with 'Executed-Success' indicating that the ICCID has been reserved.
- A 'Function execution status' indicating 'Failed' with a status code as defined in table 24 or a specific status code as defined in the following table.

Additional Input Data:

Input data name	Description	Type	No.	MOC
eid	Identification of the targeted eUICC.	EID	1	O
iccid	Identification of the Profile to download and install in the eUICC.	ICCID	1	C(1)
profileType	Identification of the Profile Type to download and install in the eUICC.	String	1	C(1)
NOTE 1: At least one of iccid and profileType SHALL be present. If only profileType is provided, the SM-DP+ SHALL select a Profile of that Profile Type from its inventory.				

Table 22: DownloadOrder Additional Input Data

Additional Output Data:

Output data name	Description	Type	No.	MOC
iccid	Identification of the Profile to download and install in the eUICC. If ICCID was provided as an input data, the returned value SHALL be the same. If not provided as an input data the returned value SHALL be one of the values available in the SM-DP+ inventory and corresponding to the Profile Type.	ICCID	1	M

Table 23: DownloadOrder Additional Output Data

Specific status codes

Subject Code	Subject	Reason code	Reason	Description
8.2.1	Profile ICCID	3.9	Unknown	Indicates that the Profile, identified by this ICCID is unknown to the SM-DP+.
8.2.1	Profile ICCID	1.2	Not Allowed (Authorisation)	Indicates that the function caller is not allowed to perform this function on the target Profile.

8.2.1	Profile ICCID	3.3	Already in Use	Indicates that the Profile identified by the provided ICCID is not available.
8.2.5	Profile Type	3.9	Unknown	Indicates that the Profile Type identified by this Profile Type is unknown to the SM-DP+.
8.2.5	Profile Type	1.2	Not Allowed (Authorisation)	Indicates that the function caller is not allowed to perform this function on the Profile Type.
8.2.5	Profile Type	3.7	Unavailable	No more Profile available for the requested Profile Type.
8.2.5	Profile Type	3.8	Refused	Indicates that the Profile Type identified by this Profile Type is not aligned with the Profile Type of Profile identified by the ICCID.

Table 24: DownloadOrder Specific Status Codes

NOTE: If the Profile identified by the ICCID is already in state "Linked" or "Allocated" and this function would result in exactly this state when performed on an unallocated Profile, the function may return 'Executed-Success' and take no other action.
This allows graceful handling of resends in case a response on ES2+ gets lost.

5.3.2 Function: ConfirmOrder

Related Procedures: Download initiation

Function Provider Entity: SM-DP+

Description:

This function is used to confirm a previously requested download order.

If an SM-DS or Default SM-DP+ is to be used for the Profile download and the EID has not been provided within the DownloadOrder function, then the EID SHALL be present. If EID is not present, the SM-DP+ SHALL return a 'Function execution status' indicating 'Failed' with a status code "EID - Mandatory Element Missing".

If the EID is present in both the DownloadOrder and ConfirmOrder functions it SHALL be the same value. If EID is different, the SM-DP+ SHALL return a 'Function execution status' indicating 'Failed' with a status code "EID - Invalid Association".

On reception of this function call, the SM-DP+ SHALL:

- Confirm the allocation of an ICCID in its inventory.
- Generate a MatchingID (section 4.1.1) if it is not provided by the Operator.
- If the Operator has provided a non-zero-length MatchingID:
  - If its format is invalid, then the SM-DP+ SHOULD return a status code "Matching ID - Invalid".
  - If it conflicts with one already stored, then the SM-DP+ SHALL return a status code "Matching ID - Already in Use".
- Store the MatchingID.

- Store the EID if available.
- If the Confirmation Code is provided by the Operator, calculate the hash of the UTF-8-encoded representation of the Confirmation Code and store the hash value together with the MatchingID, where the hash value is SHA256(Confirmation Code).
- If a Root SM-DS address (rootSmdsAddress) is provided with a non-empty value:
  - If the Root SM-DS address begins with a full stop character (e.g., `'.unspecified')`, the SM-DP+ MAY determine the applicable Root SM-DS for this Profile in an implementation-dependent manner.
  - Verify that the MatchingID is not a zero-length value. Otherwise, the SM-DP+ SHALL return a status code "Matching ID - Invalid".
  - Store the Root SM-DS address with the Profile to be used later for Event Registration and Event Deletion.
  - If an Alternative SM-DS address (smdsAddress) is also provided with a non-empty value:
    - If the Alternative SM-DS address begins with a full stop character (e.g., `'.unspecified')`, the SM-DP+ MAY determine the applicable Alternative SM-DS for this Profile in an implementation-dependent manner.
    - Store the Alternative SM-DS address with the Profile to be used later for Event Registration and Deletion.
- If the releaseFlag is set to true, perform Event Registration to the Root SM-DS address stored with the Profile, cascaded through the Alternative SM-DS address stored with the Profile – if any, as defined in section 3.6.1. The MatchingID, SHALL be used as the EventID. If the releaseFlag is set to false, the Event Registration at this point in time is optional. If it is not done in this step, it will be done during the ReleaseProfile function.
- If a single SM-DS address (smdsAddress) is provided with a non-empty value (as in v2 of this specification):
  - Verify that the MatchingID is not a zero-length value. Otherwise, the SM-DP+ SHALL return a status code "Matching ID - Invalid".
  - The SM-DP+ MAY determine, in an implementation-dependent manner, whether this is the address of a Root SM-DS or an Alternative SM-DS, and – if it is an Alternative SM-DS – the applicable Root SM-DS for the cascaded Event Registration.
  - Store the SM-DS address with the Profile to be used later for Event Registration and Event Deletion.
  - If the releaseFlag is set to true, perform Event Registration to the SM-DS address stored with the Profile as defined in section 3.6.1, where the MatchingID SHALL be used as the EventID. Otherwise, the Event Registration at this point in time is optional. If it is not done in this step, it will be done during the ReleaseProfile function.

The SM-DP+ MAY perform additional operations.

This function SHALL return one of the following:

- A 'Function execution status' with 'Executed-Success' indicating that the ICCID has been reserved.

- A 'Function execution status' indicating 'Failed' with a status code as defined in section 5.2.6 or a specific status code as defined in the following table.

Additional Input Data:

Input data name	Description	Type	No.	MOC
iccid	Identification of the Profile to download and install in the eUICC	ICCID	1	M
eid	Identification of the targeted eUICC	EID	1	O
matchingId	The MatchingID as defined in section (0), when generated by the Operator	String	1	O
confirmationCode	A code used to authorise the usage of the MatchingID to confirm the download and installation of the Profile	String	1	O
smdsAddress	The SM-DS address to be used for Event Registration. If rootSmdsAddress is also present, smdsAddress SHALL be the address of the Alternative SM-DS used for cascaded Event Registration	FQDN-Ext	1	O
rootSmdsAddress	Root SM-DS address to be used for Event Registration	FQDN-Ext	1	O
releaseFlag	If 'true', the Profile SHALL be immediately released for Profile download and installation	Boolean	1	M

Table 25: ConfirmOrder Additional Input Data

Additional Output Data:

Output data name	Description	Type	No.	MOC
eid	Identification of the targeted eUICC. EID SHALL be returned if bound to this order.	EID	1	C
matchingId	The MatchingID as defined in section (0).	String	1	M
smdpAddress	The SM-DP+ address to be used for this specific download order. This SHALL be a valid fully-qualified domain name that can be resolved by a public DNS server.	FQDN	1	O

Table 26: ConfirmOrder Additional Output Data

Specific Status Codes

Subject Code	Subject	Reason code	Reason	Description
8.2.1	Profile ICCID	3.9	Unknown	Indicates that the Profile, identified by this ICCID is unknown to the SM-DP+.
8.2.1	Profile ICCID	1.2	Not Allowed (Authorisation)	Indicates that the function caller is not allowed to perform this function on the target Profile.

8.2.6	Matching ID	2.1	Invalid	Matching ID provided by the Operator is not valid.
8.2.6	Matching ID	3.3	Already in Use (Uniqueness)	Conflicting MatchingID value.
8.9	SM-DS	5.1	Inaccessible	Indicates that the smdsAddress is invalid or not reachable.
8.9	SM-DS	4.2	Execution Error	The cascade SM-DS registration has failed. SM-DS has raised an error.
8.1.1	EID	2.2	Mandatory Element Missing	Indicates that the EID is missing in the context of this order (SM-DS address provided or MatchingID value has zero-length).
8.1.1	EID	3.10	Invalid Association	Indicates that a different EID is already associated with this ICCID.

Table 27: ConfirmOrder Specific Status Codes

NOTE: If the Profile identified by the ICCID is already in state "Confirmed" and this function would result in exactly this state when performed on the Profile in a previous state, the function may return 'Executed-Success' and take no other action.
If the Profile identified by the ICCID is already in state "Released" or any subsequent state and this function would result in state "Released" when performed on the Profile in a previous state, the function may return 'Executed-Success' and take no other action.
This allows graceful handling of resends in case a response on ES2+ gets lost.

5.3.3 Function: CancelOrder

Related Procedures: Profile Download Initiation, RPM Initiation

Function Provider Entity: SM-DP+

Description:

This function is used to cancel a pending Profile or RPM download order request.

On reception of this function call, the SM-DP+ SHALL determine the context:

- If this function is called in the context of cancelling a Profile download order:
  - Confirm that the Profile identified by the provided `iccid` is allocated and not yet downloaded. Otherwise, the SM-DP+ SHALL return a status code "Profile ICCID - Already in Use".
  - If there is a MatchingID provided in the cancelOrder function, then check that the provided `matchingId` is the one associated with the ICCID. Otherwise, the SM-DP+ SHALL return a status code "MatchingID - Invalid Association".
  - If there is an EID already associated with the ICCID, then check that the provided `eid` is the one associated with the ICCID. Otherwise, the SM-DP+ SHALL return a status code "Profile ICCID - Invalid Association".

- If this function is called in the context of cancelling an RPM download order:
  - Verify that the RPM Package identified by the provided eid and matchingId is not yet downloaded. Otherwise, the SM-DP+ SHALL return a status code "RPM Package - Already in Use".
- Cancel the pending order:
  - If this function is called in the context of cancelling a Profile download order, return the ICCID to inventory or mark it as not available for future use, based on the provided final Profile status indicator.
- If the order was previously linked to an Event Registration, the SM-DP+ SHALL subsequently execute the Event deletion procedure for the SM-DS where it was registered.

The SM-DP+ MAY perform additional operations.

This function SHALL return one of the following:

- A functionExecutionStatus with Executed-Success indicating that the ICCID or RPM Package has been released from the MatchingID and the associated Profile or RPM Package will not be downloaded.
- A functionExecutionStatus indicating Failed with a status code as defined in section 5.2.6 or a specific status code as defined in the table here after.

Additional Input Data:

Input data name	Description	Type	No.	MOC
iccid	Identification of the Profile to be cancelled from previously requested download order.	ICCID	1	C
eid	eUICC identifier.	EID	1	C
matchingId	The MatchingID as generated in ConfirmOrder.	String	1	C
finalProfileStatusIndicator	Indicator to determine whether the Profile will be available for future use after a download order is cancelled, as described in Table 29.	String	1	C

Table 28: CancelOrder Additional Input Data

The iccid and finalProfileStatusIndicator SHALL be provided if and only if this function is called in the context of cancelling a Profile download order.

The eid SHALL be provided if an EID has been associated for the Profile download order to cancel.

The matchingId SHALL be provided if a MatchingID is known by the Operator for the Profile download order to cancel.

NOTE: If a response to an earlier function call was lost, then a MatchingID may have been allocated by the SM-DP+ but is not known by the Operator.

The `eid` and `matchingId` SHALL be provided in the context of cancelling an RPM download order.

Final Profile Status Indicator	Description
`Available`	Indicates that the download order for this Profile, identified by this ICCID will be cancelled; and the Profile is released back to the inventory and available for future use.
`Unavailable`	Indicates that the download order for this Profile, identified by this ICCID will be cancelled; and the Profile is not available for future use.

Table 29: Definition of Final Profile Status Indicator

Additional Output Data:

No additional output data.

Specific Status Codes

SubjectCode	Subject	Reason code	Reason	Description
8.2.1	Profile ICCID	3.9	Unknown	Indicates that the Profile, identified by this ICCID is unknown to the SM-DP+.
8.2.1	Profile ICCID	1.2	Not Allowed (Authorisation)	Indicates that the function caller is not allowed to perform this function on the target Profile.
8.2.1	Profile ICCID	2.2	Mandatory Element Missing	Indicates that the ICCID or the finalProfileStatusIndicator is missing in the context of cancelling a Profile download order.
8.2.1	Profile ICCID	3.3	Already in Use	The profile, identified by this ICCID, is already downloaded.
8.2.1	Profile ICCID	3.10	Invalid Association	Indicates that a different EID is associated with this ICCID.
8.2.6	Matching ID	3.10	Invalid Association	Indicates that a different MatchingID is associated with this ICCID or RPM order.
8.1.1	EID	1.2	Not Allowed (Authorisation)	Indicates that the function caller is not allowed to perform this function on the target EID.
8.1.1	EID	2.2	Mandatory Element Missing	Indicates that the EID is missing in the context of this order.
8.1.1	EID	3.9	Unknown	Indicates that the eUICC, identified by this EID is unknown to the SM-DP+.
8.1.1	EID	3.10	Invalid Association	Indicates that the RPM Package is associated with different EID.

8.2.6	MatchingID	2.2	Mandatory Element Missing	Indicates that the MatchingID is missing in the context of this order.
8.12	RPM Package	3.3	Already in Use	The RPM Package is already downloaded.

Table 30: CancelOrder Specific Status Codes

5.3.4 Function: ReleaseProfile

Related Procedures: Download initiation

Function Provider Entity: SM-DP+

Description:

This function is used to release the Profile in order to allow the End User to start the download and installation procedure after the Operator performs any relevant operation on its back-end (e.g., provisioning of HLR).

On reception of this function call, the SM-DP+ SHALL:

- Verify that the Profile identified by the provided ICCID has been processed with "ES2+.DownloadOrder" and "ES2+.ConfirmOrder", but not released yet. If this verification fails, the SM-DP+ SHALL return a status code "Profile ICCID - Unknown" or "Profile ICCID - Invalid transition"
- Set the Profile state as 'Released' to allow the download.
- If SM-DS address(es) were stored with the Profile and if the Event Registration was not already done, perform Event Registration to the SM-DS(s) as defined in section 3.6.1, where the MatchingID SHALL be used as the EventID. If the SM-DS if not reachable, the SM-DP+ SHALL return a status code "SM-DS - Inaccessible". If the SM-DS returns an execution error, the SM-DP+ SHALL return a status code "SM-DS - Execution Error".

The SM-DP+ MAY perform additional operations, which are out of scope of this specification.

This function SHALL return one of the following:

- A 'Function execution status' with 'Executed-Success' indicating that the Profile identified by the provided ICCID has been released and is ready to download.
- A 'Function execution status' indicating 'Failed' with a status code as defined in section 5.2.6 or a specific status code as defined in the table here after.

Additional Input Data:

Input data name	Description	Type	No.	MOC
`iccid`	Identification of the Profile to be released from previously requested download order.	ICCID	1	M

Table 31: ReleaseProfile Additional Input Data

Additional Output Data:

No additional output data.

Specific Status Codes

SubjectCode	Subject	Reason code	Reason	Description
8.2.1	Profile ICCID	3.9	Unknown	Indicates that the Profile, identified by this ICCID, is unknown to the SM-DP+.
8.2.1	Profile ICCID	3.5	Invalid transition	Indicates that the target Profile cannot be released.
8.9	SM-DS	5.1	Inaccessible	Indicates that the smdsAddress is invalid or not reachable.
8.9	SM-DS	4.2	Execution Error	The cascade SM-DS registration has failed. SM-DS has raised an error.

Table 1Table 32: ReleaseProfile Specific Status Codes

If a Profile has already been released this function returns 'Executed-Success' and take no other action.

5.3.5 Function: HandleNotification

Related Procedures: Profile Download and Installation, RPM Download and Execution, Local/Remote Profile Management

Notification Handler/Recipient: Operator

NOTE: Prior to version 3, this function was called HandleDownloadProgressInfo.

Description:

This function is used by the SM-DP+ to notify the Operator of the progress of a pending Profile download or RPM order. This function MAY be used at several points of the Profile Download and Installation or RPM Download and Execution procedure. In addition, it is used by the SM-DP+ to notify the Operator owning the Profile that a Profile Management Operation (install, enable, disable or delete) has successfully been performed on the eUICC.

Upon reception of an ES9+.HandleNotification, the SM-DP+ SHALL correlate it to a Profile download order, an RPM order or an installed Profile and notify the related Operator accordingly.

NOTE 1: The ICCID and EID are enough to identify the related Profile download order and to retrieve the information (e.g., notificationReceiverIdentifier and notificationIdentifier) to notify the Operator.

NOTE 2: The MatchingID and EID are enough to identify the related RPM order and to retrieve the information (e.g., notificationReceiverIdentifier and notificationIdentifier) to notify the Operator.

NOTE 3: In the case of other ES9+ Notifications, the ICCID and EID are enough to retrieve the information to notify the Operator.

What is performed by the Operator receiving this notification is out of scope of this specification.

Additional Input Data:

Input data name	Description	Type	No.	MOC
eid	Identifies the targeted eUICC. This information SHALL be set if available to the SM-DP+.	EID	1	C
iccid	Identifies the Profile to download to and install in the eUICC. This SHALL only be present if this function is called in the context of Profile Download and Installation and for Profile Management Operations.	ICCID	1	C
matchingId	Identifies the RPM order to download to the eUICC. This SHALL only be present if this function is called in the context of RPM Download and Execution.	String	1	C
notificationReceiverIdentifier	Echoes the Function Requester Identifier of the relevant Profile download order or RPM order. This SHALL be present if this function is called in the context of Profile Download and Installation or RPM Download and Execution.	String	1	C
notificationIdentifier	Echoes the Function Call Identifier of the relevant Profile download order or RPM order. This SHALL be present if this function is called in the context of Profile Download and Installation or RPM Download and Execution.	String	1	C
profileType	Identifies the Profile Type to download to and install in the eUICC. This SHALL only be present if this function is called in the context of Profile Download and Installation.	String	1	C
timestamp	Indicates the date/time when the operation has been performed or when the notification has been received by the SM-DP+.	DATETIME	1	M

	NOTE: In the latter case, this is not the time when the notification was generated.			
notificationEvent	Indicates the step reached within the Profile Download and Installation or RPM Download procedure or the Profile Management Operation that was executed. Defined events are (1): 1 -> Eligibility and attempt limit check 2 -> Confirmation Failure 3 -> BPP download or RPM download 4 -> BPP installation or RPM execution (3) 5 -> notificationInstall (2) 6 -> notificationLocalEnable (2) (4) 7 -> notificationLocalDisable (2) (4) 8 -> notificationLocalDelete (2) (4) 9 -> notificationRpmEnable (2) (4) 10 -> notificationRpmDisable (2) (4) 11 -> notificationRpmDelete (2) (4) 12 -> Device Change request 13 -> Device Change confirmation 14 -> Device Change confirmation failure 15 -> Profile preparation for Device Change	INTEGER	1	M
notificationEventStatus	Indicates the status after the execution of the notification event. The ExecutionStatus type is re-used to specify the result of processing of the operation related to the notification event (Executed-Success, Failed), and optionally to provide information on any encountered problem (status code, data/object that causes the status code, and message to provide textual and human readable explanation of the status code). This field SHALL be absent for certain operations, see NOTE 3.	ExecutionStatus	1	C
resultData	The finalResult data object as contained in the ProfileInstallationResult or LoadRpmPackageResult, when received from the eUICC.	Binary	1	C
NOTE 1: This specification reserves values from 16 to 99 for future use. The Operator and the SM-DP+, based on agreed behaviour, MAY define additional custom notification events. In that case, values >=100 SHALL be used. NOTE 2: As on ES9+ these events are provided in a BIT STRING starting with bit 0, they have to be remapped to the given integer values. NOTE 3: when notificationEvent indicates 'RPM execution', the notificationEventStatus SHALL be: • 'Executed-Success' if all RPM commands in the corresponding RPMPackage have been executed successfully.				

- Otherwise, 'Failed', with a Status Code as specified in Table 32c.

NOTE 4: notificationEventStatus SHALL NOT be sent for notificationEvent values 6 to 11.

Table 32a: HandleNotification Additional Input Data

The following table provides the mapping between the cancel session reason received within the ES9+.CancelSession and the status code that SHALL be set in the notificationEventStatus input data.

Cancel session reason	Status code
endUserRejection(0)	'8.8.5 Download Order - 3.8 Refused'
pprNotAllowed(3)	'8.2.8 PPR - 1.2 Not Allowed'
metadataMismatch(4)	'8.2.9 Profile Metadata - 3.11 Value has Changed'
loadBppExecutionError(5)	'8.2.10 Bound Profile Package – 4.2 Execution Error'
enterpriseProfilesNotSupported(17)	'8.2.13 Enterprise – 3.1 Unsupported'
enterpriseRulesNotAllowed(18)	'8.2.13 Enterprise – 1.2 Not Allowed'
enterpriseProfileNotAllowed(19)	'8.2.13 Enterprise – 3.8 Refused'
enterpriseOidMismatch(20)	'8.2.13 Enterprise – 3.12 Invalid Match'
enterpriseRulesError(21)	'8.2.13 Enterprise – 3.8 Refused'
enterpriseProfilesOnly(22)	'8.2.13 Enterprise – 3.8 Refused'
lprNotSupported(23)	'8.2.14. LPA Proxy – 3.1 Unsupported' or '8.8.6 RPM Order – 3.1 Unsupported'
lprNetworkDataNotAllowed(24)	'8.8.6 RPM Order – 3.8 Refused'
emptyProfileOrSpName(25)	'8.2.9 Profile Metadata – 3.8 Refused'
rpmDisabled(27)	'8.8.6 RPM Order – 3.8 Refused'
invalidRpmPackage(28)	'8.8.6 RPM Order – 2.1 Invalid'
loadRpmPackageError(29)	'8.8.6 RPM Order – 4.2 Execution Error'
undefinedReason(127)	'8 eUICC Remote Provisioning - 4.2. Execution Error'

Table 32b: Cancel session reason code mapping to Status code

NOTE: `postponed(1)`, `timeout(2)` and `sessionAborted(22)` cancel session reasons do not terminate the download order, and thus do not lead to a notification to the Operator.

The following table provides additional status codes that can be set in the notificationEventStatus input data by the SM-DP+:

Reason	Status code
Maximum number of attempts of Profile Download has been exceeded	'8.8.5 Download Order - 6.4 Maximum number of retries exceeded'

Maximum number of incorrect attempts of Confirmation Code verification has been exceeded	'8.2.7 Confirmation Code – 6.4 Maximum number of retries exceeded'
Execution of at least one RPM command within an RPM package has failed	'8.10.2. RPM Script - 4.2. Execution Error'

Table 32c: Additional Status Codes

5.3.6 Function: RpmOrder

Related Procedures: RPM Initiation

Function Provider Entity: SM-DP+

Description:

This function is used to instruct the SM-DP+ of a new RPM Package.

The rpmScript as defined in section 2.10.1 SHALL include one or more RPM Command(s) for the target EID.

On reception of this function call, the SM-DP+ SHALL:

- Verify that the `eid` is present. Otherwise, the SM-DP+ SHALL return a status code "EID - Mandatory Element Missing".
- Identify the eUICC by using the `eid`. If it cannot be identified, the SM-DP+ SHALL return a status code "EID - Unknown".
- Generate a MatchingID (section 4.1.1) if it is not provided by the Operator.
- If the Operator has provided the MatchingID:
    - If its format is invalid, then the SM-DP+ SHOULD return a status code "Matching ID - Invalid".
    - If it conflicts with one already stored, then the SM-DP+ SHALL return a status code "Matching ID - Already in Use".
- Store the MatchingID and the EID.
- Prepare an RPM Package as follows.
    - If the rpmScript includes RPM Command 'Enable Profile', 'Disable Profile', 'Delete Profile', 'List Profile Info' (with ICCID), 'Contact PCMP' or 'Update Metadata' coded as defined in section 2.10.1, the SM-DP+ SHALL for each of these commands:
        - Identify the Profile associated with the ICCID. If the ICCID is not provided, the SM-DP+ SHALL return a status code "Profile ICCID - Conditional Element Missing". If the Profile cannot be identified, the SM-DP+ SHALL return a status code "Profile ICCID - Unknown".
        - Verify that the function caller is the Profile Owner of the Profile. If it is not, the SM-DP+ SHALL return a status code "Profile ICCID - Unknown".
        - Verify that the Profile is installed in the target eUICC. If it is not, the SM-DP+ SHALL return a status code "EID - Invalid Association".

        - For RPM Command 'Update Metadata', validate the provided updateMetadataRequest field. If no Metadata object is present, the SM-DP+ SHALL return a status code "Profile Metadata - Conditional Element Missing". If it is invalid, the SM-DP+ SHALL return a status code "Profile Metadata - Invalid".

    - If the rpmScript includes RPM Command(s) 'List Profile Info' (with Profile Owner OID) coded as defined in section 2.10.1, for each of these commands, the SM-DP+ SHALL verify that the function caller correctly presented its Profile Owner OID in the RPM Command. If not, the SM-DP+ SHALL return a status code "Profile Owner - Invalid Association".

- Associate the RPM Package with the EID and MatchingID.

- If a Root SM-DS address is provided and optionally also an Alternative SM-DS address with non-empty value(s):

    - Verify that the MatchingID is not a zero length value. If the MatchingID is a zero length value, the SM-DP+ SHALL return a status code "Matching ID - Invalid".

    - Store the SM-DS address(es) with the RPM Package to be used later for Event Registration and Event Deletion.
        - If the Root SM-DS address begins with a full stop character (e.g., `'.unspecified')`, the SM-DP+ MAY determine the applicable Root SM-DS for this Profile in an implementation-dependent manner.
        - If the Alternative SM-DS address begins with a full stop character (e.g., `'.unspecified')`, the SM-DP+ MAY determine the applicable Alternative SM-DS for this Profile in an implementation-dependent manner.

    - Perform Event Registration as defined in section 3.6.1, where the MatchingID SHALL be used as the EventID. If the SM-DS is not reachable, the SM-DP+ SHALL return a status code "SM-DS - Inaccessible". If the Event Registration fails, the SM-DP+ SHALL return a status code "SM-DS - Execution Error".
        - If an Alternative SM-DS was specified, this SHALL be a cascaded registration as defined in section 3.6.1.2. Otherwise, it SHALL be a non-cascaded registration as defined in section 3.6.1.1.

The SM-DP+ MAY perform additional operations.

This function SHALL return one of the following:

- A 'Function execution status' with 'Executed-Success' indicating that the RPM Package has been created.

- A 'Function execution status' indicating 'Failed' with a status code as defined in section 5.2.6 or a specific status code as defined in the table here after.

Additional Input Data:

Input data name	Description	Type	No.	MOC
`eid`	Identification of the target eUICC.	EID	1	M
`rpmScript`	RpmPackage as defined in section 2.10.1.	Binary	1	M
`matchingId`	The MatchingID as defined in section 3.7.1, when generated by the Operator.	String	1	O

rootSmdsAddress	The Root SM-DS address to be to be used for the Event Registration.	FQDN-Ext	1	O
altSmdsAddress	The Alternative SM-DS address to be used for cascaded Event Registration.	FQDN-Ext	1	C(1)
NOTE 1: This field MAY be provided only if rootSmdsAddress is present.				

Table 32d: RpmOrder Additional Input Data

Additional Output Data:

Output data name	Description	Type	No.	MOC
matchingId	The MatchingID as defined in section 3.7.1	String	1	M

Table 32e: RpmOrder Additional Output Data

Specific status codes:

SubjectCode	Subject	Reasoncode	Reason	Description
8.1.1	EID	2.2	Mandatory Element Missing	Indicates that the EID is missing in the context of this order.
8.1.1	EID	3.9	Unknown	Indicates that the eUICC, identified by this EID is unknown to the SM-DP+.
8.1.1	EID	3.10	Invalid Association	Indicates that a different EID is already associated with this ICCID.
8.2.1	Profile ICCID	2.3	Conditional Element Missing	Indicates that the ICCID is missing in the context of the RPM Command.
8.2.1	Profile ICCID	3.9	Unknown	Indicates that the Profile, identified by this ICCID is unknown to the SM-DP+.
8.2.6	Matching ID	2.1	Invalid	Matching ID provided by the Operator is not valid.
8.2.6	Matching ID	3.3	Already in Use (Uniqueness)	Matching ID provided by the Operator is already in use at the SM-DP+.
8.2.9	Profile Metadata	2.1	Invalid	Indicates that the provided Profile Metadata is not valid.
8.2.9	Profile Metadata	2.3	Conditional Element Missing	Indicates that no Metadata object is provided in the

				context of the RPM Command.
8.2.12	Profile Owner	3.10	Invalid Association	The Operator provided incorrect Profile Owner OID.
8.9	SM-DS	4.2	Execution Error	The cascade SM-DS registration has failed. SM-DS has raised an error.
8.9	SM-DS	5.1	Inaccessible	Indicates that the smdsAddress is invalid or not reachable.
8.10.2	RPM Script	2.2	Mandatory Element Missing	Indicates that the RPM Script is missing in the context of this order.

Table 32f: RpmOrder Specific Status Codes

NOTE: If an RPM order with exactly the same parameters in the input data already exists, the function may return 'Executed-Success' and take no other action. This allows graceful handling of resends in case a response on ES2+ gets lost.

5.3.7 Function: HandleDeviceChangeRequest

Related Procedures: Device Change

Function Provider Entity: Service Provider

Description:

The SM-DP+ supporting Device Change SHALL be able to call ES2+.HandleDeviceChangeRequest function as per Service Provider's configuration.

This function requests for Service Provider's confirmation on the Device Change of a Profile. If it was requested as per Service Provider's configuration, the SM-DP+ SHALL call this function on the reception of Device Change request for the Profile.

On reception of this function call, the Service Provider SHALL:

- Identify the Profile by ICCID. If the Profile cannot be identified, the Service Provider SHALL return an error status "ICCID – Unknown".
- Verify that the identified Profile is eligible for the Device Change. If not, the Service Provider SHALL return an error status "Device Change – Not Allowed".
- Set the `isNewProfileRequired` to true if a new Profile is required for the Device Change.
- If the End User requires to enter a Confirmation Code to proceed the Device Change procedure: generate a Confirmation Code.

Additionally, the Service Provider MAY:

- Generate Service Provider Message for Device Change to be confirmed by the End User.

The Service Provider MAY perform additional operations, which are out of scope of this specification.

This function SHALL return one of the following:

- A 'Function execution status' with 'Executed-Success' indicating that the Service Provider has confirmed the Device Change of the Profile.
- A 'Function execution status' indicating 'Failed' with a status code as defined in section 5.2.6, or a specific status code as defined in the following table.

Additional Input Data:

Input data name	Description	Type	No.	MOC
iccid	Identification of the Profile installed in the eUICC of the old eUICC.	ICCID	1	M
eid	Identification of the target eUICC in the new Device	EID	1	O
tac	TAC value of the new Device.	String	1	O

Table 32g: HandleDeviceChangeRequest Additional Input Data

Additional Output Data:

Output data name	Description	Type	No.	MOC
isNewProfileRequired	Indicates if the new Profile is required for the Device Change.	Boolean	1	M
serviceProviderMessageForDc	Service Provider specific message for Device Change to be confirmed by the End User.	String	1	O
cc	A Confirmation Code to be verified for the Device Change of the Profile.	String	1	O

Table 32h: HandleDeviceChangeRequest Additional Output Data

Specific status codes

Subject Code	Subject	Reason code	Reason	Description
8.2.1	Profile ICCID	3.9	Unknown	Indicates that the installed Profile, identified by this ICCID, is unknown to the Service Provider.
8.10.3	Device Change	1.2	Not Allowed	Indicates that the installed Profile is not eligible for Device Change.

Table 32i: HandleDeviceChangeRequest Specific Status Codes

5.4 ES6 (Operator -- eUICC)

This interface is present between the Operator and their Enabled Profile in eUICC. It allows the Operator to make modifications on their Profile in the eUICC using legacy OTA mechanisms.

The ES6 functions are addressed to the eUICC through a secure channel, as defined in ETSI TS 102 225 [38] and ETSI TS 102 226 [39], established between the Operator and the MNO-SD of the Enabled Profile. This interface is the same as the one used with UICCs.

The initial OTA Key sets are part of the Profile and are loaded by the SM-DP+ during the "Profile Download and Installation" (section 3.1.3), or loaded by the EUM before eUICC issuance.

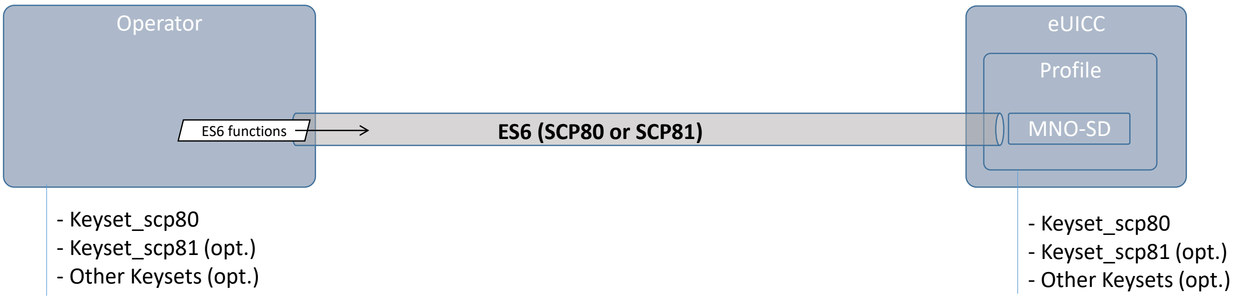


Figure 32: ES6

5.4.1 Function: UpdateMetadata

Related Procedures: Metadata Update

Function Provider Entity: ISD-P

Description:

This function allows updating the following Profile Metadata of the target Profile:

- Service Provider name
- Profile Name
- Icon type and Icon
- Profile Policy Rule
- Notification Configuration Info
- Service Specific Data stored in eUICC
- RPM Configuration
- Address of the HRI Server
- LPA Proxy Configuration
- Enterprise Configuration
- Device Change Configuration

All Metadata elements that can be updated by this function can also be deleted from the Metadata except Service Provider name and Profile Name.

As this function is provided by the ISD-P, the STORE DATA command message defined hereunder has to be preceded by an INSTALL [for personalisation] as defined in SGP.02 [2] section 4.1.2.1:

- The reserved AID value for Profile’s ISD-P (Annex D) SHALL be used to indicate that the Security Domain target by the INSTALL [for personalisation] command is the ISD-P of the Profile containing the MNO-SD.
- According to GlobalPlatform Card Specification [8], INSTALL [for personalisation] command can only be used on applications associated with a Security Domain. As an exception to this rule, the eUICC SHALL allow the MNO-SD to receive this command sequence with data destined to the ISD-P.

If supported by the Device, after a successful update of Metadata objects, the eUICC SHALL alert the LPA as specified in section 5.7.23.

Command Message

This function uses the command message STORE DATA as defined in GlobalPlatform Card Specification [8] with the specific coding defined in this section.

Code	Value	Meaning
CLA	'80'	GPCS [8] section 11.11
INS	'E2'	STORE DATA
P1	'10' or '90'	See below
P2	xx	Block number
Lc	xx	Length of data field
Data	'xx xx…'	See below

Table 33: UpdateMetadata Command Message

Parameter P1

The P1 SHALL be coded as follows:**Error! Reference source not found.**

b8	b7	b6	b5	b4	b3	b2	b1	Meaning
0/1	-	-	-	-	-	-	-	More blocks / Last block
-	0	0	-	-	-	-	-	No general encryption information or non-encrypted data
-	-	-	1	0	-	-	-	BER-TLV format of the command data field
-	-	-	-	-	-	-	0	Case 3 command as defined in GlobalPlatform Amd A [9]
-	-	-	-	-	X	X	-	RFU

Table 34: UpdateMetadata P1

If the provided Profile Metadata values do not fit in a single STORE DATA command, the Operator SHALL split them into several STORE DATA commands. A transfer of an intermediate command SHALL be done by indicating "More blocks". The last or only command SHALL be transferred indicating "Last block".

The command can also be sent via RPM. In this case, the eUICC SHALL perform the following before processing the command data:

- Verify that the Profile identified by the ICCID exists. Otherwise, the eUICC SHALL return an error code `commandError`.
- Verify that the SM-DP+ that sent the RPM Command is included in the Managing SM-DP+ List of the target Profile and is authorised to update all the given Metadata objects. Otherwise, the eUICC SHALL return an error code `commandError`.
- If the Profile Metadata of the target Profile specifies an allowed eSIM CA RootCA public key identifier for the Managing SM-DP+: verify that the Subject Key Identifier of the eSIM CA RootCA corresponding to CERT.DPauth.SIG matches that value. Otherwise, the eUICC SHALL return an error code `commandError`.

Data Field

The data field SHALL be coded as follows:

```
-- ASN1START
UpdateMetadataRequest ::= [42] SEQUENCE {  -- Tag 'BF2A'
   serviceProviderName [17] UTF8String (SIZE(0..32)) OPTIONAL, -- Tag '91'
   profileName [18] UTF8String (SIZE(0..64)) OPTIONAL, -- Tag '92'
   iconType [19] IconType OPTIONAL, -- Tag '93'
   icon [20] OCTET STRING (SIZE(0..1024)) OPTIONAL, -- Tag '94'
   profilePolicyRules [25] PprIds OPTIONAL, -- Tag '99'
   serviceSpecificDataStoredInEuicc [34] VendorSpecificExtension OPTIONAL, --
#SupportedFromV2.4.0# Tag 'BF22'
   notificationConfigurationInfo [22] SEQUENCE OF
NotificationConfigurationInformation OPTIONAL, -- #SupportedFromV3.0.0# Tag 'B6'
   tagsForDeletion [APPLICATION 28] OCTET STRING OPTIONAL, -- for tagList
#SupportedFromV3.0.0# tag '5C'
   rpmConfiguration [26] RpmConfiguration OPTIONAL, -- #SupportedForRpmV3.0.0# Tag
'BA'
   hriServerAddress [27] UTF8String OPTIONAL, -- #SupportedFromV3.0.0# Tag '9B'
   lprConfiguration [28] LprConfiguration OPTIONAL, -- #SupportedForLpaProxyV3.0.0#
Tag 'BC'
   enterpriseConfiguration [29] EnterpriseConfiguration OPTIONAL, --
#SupportedForEnterpriseV3.0.0# Tag 'BD'
   deviceChangeConfiguration [32] DeviceChangeConfiguration OPTIONAL --
#SupportedForDcV3.0.0# Tag 'BF20'
}
-- ASN1STOP
```

Even though all objects directly contained in the `UpdateMetadataRequest` are optional, at least one SHALL be present. An update always applies to the whole object.

Unless defined otherwise below, the following general processing rules SHALL apply for all objects except `tagsForDeletion`:

- If an object is present, the eUICC SHALL update the Profile Metadata with the provided value. It the object was not present in the Metadata before, it SHALL be added to the Metadata.
- If an object is absent, the eUICC SHALL NOT update the corresponding Profile Metadata object.

- If there is an error while processing the UpdateMetadata, the Profile Metadata SHALL remain unchanged.

If present, tagsForDeletion SHALL contain a concatenated list of tags of objects to be deleted from the Metadata. An object that is to be created or updated by the function SHALL NOT also appear in the list for deletion. The eUICC SHALL delete objects as follows:

- If Service Provider name or Profile Name or any non-updateable objects are listed, the eUICC SHALL return an error code deleteNotAllowed. These objects cannot be deleted.
- If an object indicated by a tag is not present in the Metadata, the eUICC SHALL silently ignore the tag.
- For all other tags, the eUICC SHALL delete the object.

NOTE: Version 2 defined a different mechanism for the deletion of Metadata objects. An OTA server that wants to delete a Metadata object or update a Metadata object with a zero-length value should take these differences into account. An SM-DP+ can know from EuiccRspCapability.updateMetadataV3Support whether the new mechanism is supported on the eUICC and can forward this information to the OTA server.

An OTA server SHALL add or remove iconType and icon together. The eUICC is not mandated to check for both being present or both being absent as a result of a Metadata update.

For profilePolicyRules:

- This version of the specification only defines unsetting PPRs. For this operation, the pprUpdateControl bit SHALL be set to zero. In case of a value of one, the eUICC SHALL return an error code pprUpdateInvalidSetting.
- If pprUpdateControl is set to zero, the following SHALL apply: Each PPR bit of the Profile SHALL be logically ANDed with the corresponding bit of the UpdateMetadataRequest.

For notificationConfigurationInfo: Version 2 did not include this object in the list of Metadata objects that can be updated via ES6. An SM-DP+ can know from EuiccRspCapability.updateNotifConfigInfoSupport whether updating of this object is supported on the eUICC and can forward this information to the OTA server.

hriServerAddress being present in the command, even if it contains the same value as the one already stored in the Metadata on the eUICC, SHALL be taken by the LPA as indication that the icon on the server has changed.

The following processing SHALL apply if rpmConfiguration is contained in the command:

- If the object in the command contains a zero-length value, the command SHALL be rejected with error code `invalidRpmConfiguration`.
- If an RPM Configuration already exists in the target Profile and if the `profileOwnerOid` in the command does not match the Profile Owner OID in the RPM Configuration of the target Profile, the command SHALL be rejected with error code `invalidRpmConfiguration`.

The following processing SHALL apply if `enterpriseConfiguration` is contained in the command:

- If the target Profile is not an Enterprise Profile or the object in the command contains a zero-length value, the command SHALL be rejected.
- If the `enterpriseOid` in the command is different from the value stored in the Metadata of the target Profile, the command SHALL be rejected.
- If it contains `enterpriseRules` and the Metadata of the target Profile does not already contain Enterprise Rules, the command SHALL be rejected.
- If it contains `enterpriseRules` with the `referenceEnterpriseRule` bit being set, the eUICC SHALL verify the following:
  - If the `enterpriseRules` indicate `onlyEnterpriseProfilesCanBeInstalled` and `numberOfNonEnterpriseProfiles` contains a value different from 0, the command SHALL be rejected.
  - If the `enterpriseRules` indicate "`priorityEnterpriseProfile`" and the Target Profile is Disabled, the command SHALL be rejected.
  - If more non-Enterprise Profiles are enabled than allowed by `numberOfNonEnterpriseProfiles`, the command SHALL be rejected.
- For all errors listed above, the eUICC SHALL return an error code `enterpriseConfigurationNotAllowed`.
- If the previous steps did not result in the rejection of the command, the `enterpriseConfiguration` SHALL be updated.
- If the `enterpriseConfiguration` contains `enterpriseRules` with the `referenceEnterpriseRule` bit being set, the eUICC SHALL unset the `referenceEnterpriseRule` bit of the Enterprise Profile for which it is currently set, if any.

<u>Response data if the command is sent via RPM</u>

```
-- ASN1START
UpdateMetadataResponse ::= [42] INTEGER { -- #SupportedForRpmV3.0.0# Tag 'BF2A'
   ok (0),
   enterpriseConfigurationNotAllowed (6), -- #SupportedForEnterpriseV3.0.0#
   commandError (7),
   pprUpdateInvalidSetting (12),
   invalidRpmConfiguration (14),
   deleteNotAllowed (15),
   undefinedError(127)
}
-- ASN1STOP
```

Response Message for ES6

Data Field

The data field of the response message SHALL NOT be present.

Processing State Returned in the Response Message

See GlobalPlatform Card Specification [8] section 11.11.3.2.

The following additional status bytes are defined:

- '69 85': `enterpriseConfigurationNotAllowed`
- '6A 80': `deleteNotAllowed`
- '6A 81': `pprUpdateInvalidSetting` or `invalidRpmConfiguration`

5.5 ES8+ (SM-DP+ -- eUICC)

The ES8+ is an interface defined between the Profile Package Binding function of the SM-DP+ and the eUICC. This interface is intended to be tunnelled over the ES9+ and ES10b interfaces.

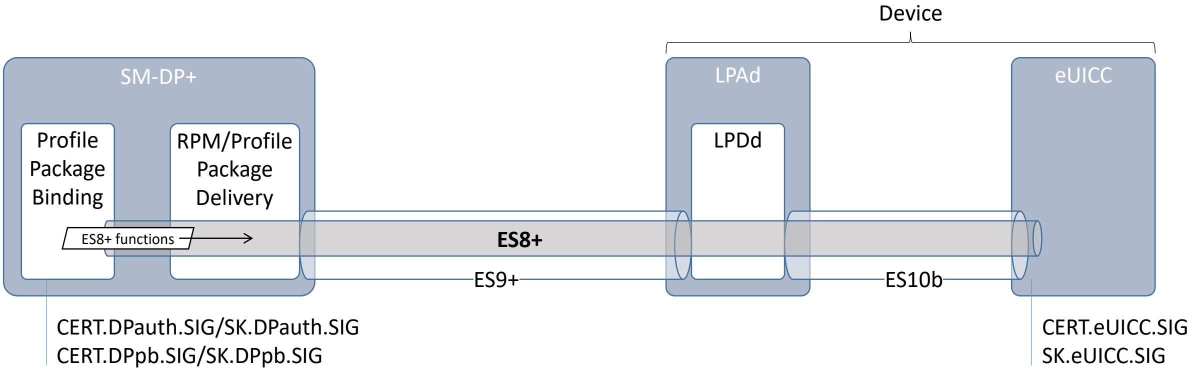


Figure 33: ES8+

The ES8+ functions are addressed to the eUICC through a secure channel established between the Profile Package Binding function of the SM-DP+ and the eUICC.

The secure channel is established by:

- Mutual authentication of the eUICC and the SM-DP+ using SK.DPauth.SIG / CERT.DPauth.SIG and SK.EUICC.SIG/CERT.EUICC.SIG.
- Session keys agreement based on exchanged one-time public keys of both parties during mutual authentication.

The SM-DP+ authenticates the eUICC by:

- Verifying the CERT.EUICC.SIG Certificate chain.

- Verifying the signature of the eUICC computed over an SM-DP+ challenge with PK.EUICC.SIG extracted from the verified CERT.EUICC.SIG.

The eUICC authenticates the SM-DP+ by:

- Verifying the CERT.DPauth.SIG Certificate chain.
- Verifying the signature of the SM-DP+ with the PK.DPauth.SIG extracted from the verified CERT.DPauth.SIG.

The data exchanged after channel establishment are secured using the BSP as defined in section 2.6.4. The eUICC SHALL support the BSP with:

- The symmetric algorithm related to the selected signature algorithm, see section 2.6.5.
- Use of C-MAC and C-DECRYPTION.

As a result the SM-DP+ and eUICC are mutually authenticated, all data sent from the Profile Package Binding function of the SM-DP+ to the eUICC are MACed and encrypted, except the 'StoreMetadata' command which is only MACed.

Response data generated by the eUICC when processing the BPP received on ES8+ is returned protected by a signature generated by the eUICC.

5.5.1 Function: InitialiseSecureChannel

Related Procedures: Profile Download and Installation

Function Provider Entity: ISD-R

Description:

This function is used by the SM-DP+ to initialise the secure channel for a Profile Download and Installation with the target eUICC. The function carries the identifier of the remote operation type to be performed by the eUICC (e.g., installation of a new Bound Profile Package) and the necessary material for key agreement with Perfect Forward Secrecy (PFS), allowing a secure end-to-end communication between the SM-DP+ and the eUICC:

- Transaction ID
- Description of the keys to generate
- One-time public key for key agreement generated by SM-DP+ (otPK.DP.KA)
- Signature upon material (including the previously generated otPK.EUICC.KA, also acting as an eUICC challenge) to ensure its integrity and authenticity.

The level of security is implicitly deduced from the remote operation type to execute.

The reception of the InitialiseSecureChannel function SHALL be rejected if a secure channel session is already ongoing.

On reception of this command the eUICC SHALL:

- Verify the SM-DP+ signature using the PK.DPpb.SIG; if the signature is invalid the command SHALL be rejected, an `invalidSignature` error SHALL be returned in the Profile Installation Result, Profile installation SHALL be aborted, and any contextual data associated to its Profile installation (like the SM-DP+ Certificate) SHALL be discarded.

- Verify that the requested Remote operation type is one of the defined types. Otherwise an `unsupportedRemoteOperationType` error SHALL be returned in the Profile Installation Result.
- Verify that the received transaction ID matches the transaction ID of the on-going RSP session (section 5.7.5 "ES10b.PrepareDownload" function). Otherwise an `invalidTransactionId` error SHALL be returned in the Profile Installation Result.
- Verify that Control Reference Template describing the keys to generate matches the values defined here under (Command message part). Otherwise an `unsupportedCrtValues` error SHALL be returned in the Profile Installation Result.
- Generate the session keys (S-ENC and S-MAC) and the initial MAC chaining value from received otPK.DP.KA and previously generated otSK.EUICC.KA, using the key agreement algorithm determined according to section 2.6.5.

Command Data

The command data for this function is encoded in the ASN.1 data object as described below.

```
-- ASN1START
--Definition of data objects for InitialiseSecureChannel Request
InitialiseSecureChannelRequest ::= [35] SEQUENCE { -- Tag 'BF23'
   remoteOpId RemoteOpId, -- Remote Operation Type Identifier (value SHALL be set
to installBoundProfilePackage)
   transactionId [0] TransactionId, -- The TransactionID generated by the SM-DP+
   controlRefTemplate[6] IMPLICIT ControlRefTemplate, -- Control Reference Template
(Key Agreement). Current specification considers a subset of CRT specified in
GlobalPlatform Card Specification Amendment F [13] section 6.5.2.3 for the Mutual
Authentication Data Field
   smdpOtpk [APPLICATION 73] OCTET STRING, ---otPK.DP.KA in accordance with
GlobalPlatform Card Specification Amendment F [13] section 6.5.2.3 for ePK.OCE.KA,
tag '5F49'
   smdpSign [APPLICATION 55] OCTET STRING -- SM-DP's signature, tag '5F37'
}

ControlRefTemplate ::= SEQUENCE {
   keyType[0] Octet1, -- Key type according to GlobalPlatform Card Specification
[8] Table 11-16, Tag '80'
   keyLen[1] Octet1, --Key length in number of bytes. Tag '81'
   hostId[4] OctetTo16 -- Host ID value , Tag '84'
}
-- ASN1STOP
```

NOTE: The tag '90' for 'SCP identifiers and parameters' is not used. This specification only uses one SCP type derived from SCP11a defined in GlobalPlatform Card Specification Amendment F [13]. The tag '95' for 'Key Usage Qualifier' is also not used. This is determined by the 'Remote operation type identifier' (see hereunder).

The eUICC SHALL verify the values provided for key type and key length match the expected symmetric encryption algorithm according to section 2.6.5:

- When AES-128 is selected by the SM-DP+, `keyType` SHALL contain value '88' and `keyLen` SHALL contain '10'.

- When SM4 is selected by the SM-DP+, `keyType` SHALL contain value '89' and `keyLen` SHALL contain '10'.

NOTE: Key type values are assigned in the GlobalPlatform Card Specification [8].

SM-DP+ signature (smdpSign) is computed as described in section 2.6.9, using the SM-DP+ private key SK.DPpb.SIG across the following concatenated data objects:

- remoteOpId
- transactionId
- controlRefTemplate
- smdpOtpk
- euiccOtpk, as provided earlier in the prepareDownloadResponse data object received in the "ES9+.GetBoundProfilePackage" function.

As the signature includes the otPK.EUICC.KA, the eUICC can authenticate the SM-DP+.

When `remoteOpId` is `installBoundProfilePackage` , the implicit Key Usage Qualifier SHALL be set to MAC and ENCRYPTION.

5.5.2 Function: ConfigureISDP

Related Procedures: Profile Download and Installation

Function Provider Entity: ISD-R

Description:

This function is used by the SM-DP+ to provide data to the eUICC for configuring the ISD-P. For this version of the specification, this data element only contains the optional SM-DP+ proprietary data.

NOTE: Information like the amount of assigned memory MAY be added in future versions.

On reception of this command the eUICC SHALL:

- Create the ISD-P for the Profile and assign an AID value from the range reserved for ISD-Ps in SGP.02 [2].
- If the length of the SM-DP+ proprietary data exceeds the maximum size, terminate with error 'incorrectInputValues'.
- Store the SM-DP+ proprietary data in the ISD-P.

Command data

The command data for this function is encoded in the ASN.1 data object below.

```
-- ASN1START
--Definition of data objects for ConfigureISDPRequest
ConfigureISDPRequest ::= [36] SEQUENCE { -- Tag 'BF24'
   dpProprietaryData [24] DpProprietaryData OPTIONAL -- Tag 'B8'
}
```

```
DpProprietaryData ::= SEQUENCE { -- maximum size including tag and length field: 
128 bytes
   dpOid OBJECT IDENTIFIER -- OID in the tree of the SM-DP+ that created the 
Profile
   -- additional data objects defined by the SM-DP+ MAY follow
}
-- ASN1STOP
```

5.5.3 Function: StoreMetadata

Related Procedures: Profile Download and Installation

Function Provider Entity: ISD-R

Description:

This function is used by the SM-DP+ to provide Profile Metadata of the Profile to the eUICC.

On reception of this command the eUICC SHALL verify the following:

- The Profile Class is supported. Otherwise, the reported error SHALL be `unsupportedProfileClass`.
- The ICCID is different than that of all other installed profiles. Otherwise, the reported error SHALL be `installFailedDueToIccidAlreadyExistsOnEuicc`.
- If PPRs are provided in the Profile Metadata: the Profile Owner data object is present and the PPRs are allowed for the Profile Owner. This verification SHALL be done as described section 2.9.3.1. Otherwise, the reported error SHALL be `pprNotAllowed`.
- If `enterpriseConfiguration` is provided in the Profile Metadata:
    - That it supports Enterprise Profiles. Otherwise, the reported error SHALL be `enterpriseProfilesNotSupported`.
    - If it contains `enterpriseRules`: that the Device is an Enterprise-Capable Device. Otherwise, the reported error SHALL be `enterpriseRulesNotAllowed`.
    - That none of the installed Profiles has PPR1 set. Otherwise, the reported error SHALL be `enterpriseProfileNotAllowed`.
    - If an Enterprise Profile is already installed on the eUICC: that the `enterpriseOid` in the command is identical to the value of an Enterprise Profile already installed on the eUICC. Otherwise, the reported error SHALL be `enterpriseOidMismatch`.
    - If it contains `enterpriseRules`: that the `referenceEnterpriseRule` bit is not set. Otherwise, the reported error SHALL be `enterpriseRulesError`.
- If there is a Profile with a Reference Enterprise Rule installed on the eUICC and this rule prohibits the installation of non-Enterprise Profiles: that the Profile to be installed is an Enterprise Profile. Otherwise, the reported error SHALL be `enterpriseProfilesOnly`.
- If `lprConfiguration` is provided in the Profile Metadata: that it and the Device both support the LPA Proxy. Otherwise, the reported error SHALL be `lprNotSupported.`

- If an unknown TLV is encountered, the eUICC SHALL report an error `unknownTlvInMetadata`.

If any verification fails, the eUICC SHALL report the indicated error and stop the profile installation procedure. Otherwise, store the data elements for future use.

Command data

The command data for this function is identified by the data structure defined hereunder.

```
-- ASN1START
StoreMetadataRequest ::= [37] SEQUENCE { -- Tag 'BF25'
   iccid Iccid,
   serviceProviderName [17] UTF8String (SIZE(0..32)), -- Tag '91'
   profileName [18] UTF8String (SIZE(0..64)), -- Tag '92' (corresponds to 'Short
Description' defined in SGP.21 [2])
   iconType [19] IconType OPTIONAL, -- Tag '93' (JPG or PNG)
   icon [20] OCTET STRING (SIZE(0..1024)) OPTIONAL, -- Tag '94' (Data of the icon.
Size 64 x 64 pixel. This field SHALL only be present if iconType is present)
   profileClass [21] ProfileClass DEFAULT operational, -- Tag '95'
   notificationConfigurationInfo [22] SEQUENCE OF
NotificationConfigurationInformation OPTIONAL,
   profileOwner [23] OperatorId OPTIONAL, -- Tag 'B7'
   profilePolicyRules [25] PprIds OPTIONAL, -- Tag '99'
   serviceSpecificDataStoredInEuicc [34] VendorSpecificExtension OPTIONAL, --
#SupportedFromV2.4.0# Tag 'BF22'
   serviceSpecificDataNotStoredInEuicc [35] VendorSpecificExtension OPTIONAL, --
#SupportedFromV2.4.0# Tag 'BF23'
   rpmConfiguration [26] RpmConfiguration OPTIONAL, -- #SupportedForRpmV3.0.0# Tag
'BA'
   hriServerAddress [27] UTF8String OPTIONAL, -- #SupportedFromV3.0.0# Tag '9B'
   serviceProviderMessage [30] LocalisedTextMessage OPTIONAL, --
#SupportedFromV3.0.0# Tag 'BE'
   lprConfiguration [28] LprConfiguration OPTIONAL, -- #SupportedForLpaProxyV3.0.0#
Tag 'BC'
   enterpriseConfiguration [29] EnterpriseConfiguration OPTIONAL, --
#SupportedForEnterpriseV3.0.0# Tag 'BD'
   serviceDescription [31] ServiceDescription OPTIONAL, -- #SupportedFromV3.0.0#
Tag '9F1F'
   deviceChangeConfiguration [32] DeviceChangeConfiguration OPTIONAL, --
#SupportedForDcV3.0.0# Tag 'BF20'
   estimatedProfileSize [33] INTEGER OPTIONAL -- #SupportedFromV3.0.0# Tag '9F21'
}

NotificationEvent ::= BIT STRING {
   notificationInstall(0),
   notificationLocalEnable(1),
   notificationLocalDisable(2),
   notificationLocalDelete(3),
   notificationRpmEnable(4), -- #SupportedForRpmV3.0.0#
   notificationRpmDisable(5), -- #SupportedForRpmV3.0.0#
   notificationRpmDelete(6), -- #SupportedForRpmV3.0.0#
   loadRpmPackageResult(7) -- #SupportedForRpmV3.0.0#
}

NotificationConfigurationInformation ::= SEQUENCE {
   profileManagementOperation NotificationEvent,
   notificationAddress UTF8String -- FQDN to forward the notification
}

ServiceDescription ::= BIT STRING { -- 1: service is on, 0: service is off
#SupportedFromV3.0.0#
   voice (0), -- Operator-provided voice service
   data (1) -- Operator-provided data service
}
```

```
-- ASN1STOP
```

Unless specified otherwise below, the eUICC SHALL store a data object which is present in the command.

Each bit indicating a specific event MAY appear several times in the sequence of `notificationConfigurationInfo` data object. In that case, it specifies several recipient addresses for the same notification event. The `loadRpmPackageResult` has no meaning when provided in the `notificationConfigurationInfo`.

The data object `profileOwner` SHALL be present if the `profilePolicyRules` data object is present. In this instance the `mccMnc` field SHALL NOT specify any wildcard ('E') digits.

The SM-DP+ SHALL NOT specify an empty string in the data objects `profileName` and `serviceProviderName`.

The data object `profilePolicyRules` SHALL NOT be present for a Profile that has no PPR set. Otherwise, the `profilePolicyRules` SHALL identify all the PPRs set in the Profile. If the `profilePolicyRules` data object is not present, all PPR bits of the Profile SHALL be considered zero. The `PprIds` type is defined in section 2.4a.1.1.

The data object `lprConfiguration` SHALL be present only if the eUICC supports the LPA Proxy.

The eUICC SHALL NOT store the data object `serviceSpecificDataNotStoredInEuicc` if present.

The SM-DP+ SHALL not include the data objects `serviceSpecificDataStoredInEuicc` and/or `serviceSpecificDataNotStoredInEuicc` unless the eUICC indicated `serviceSpecificDataSupport`.

The information defined in `serviceSpecificDataStoredInEuicc` and `serviceSpecificDataNotStoredInEuicc` SHALL neither impact the functionalities and Profile Management Operations defined in this specification that are not vendor specific, nor the interoperability of the solution defined in this specification (incl. Devices, Profiles, and SM-DP+s).

The SM-DP+ SHALL include the data object `rpmConfiguration` only if the eUICC indicated `rpmSupport`.

The SM-DP+ SHALL include the data object `hriServerAddress` only if the eUICC indicated `hriServerAddressSupport`.

The SM-DP+ SHALL include the data object `serviceProviderMessage` only if the eUICC indicated `serviceProviderMessageSupport`. The eUICC SHALL NOT store this data object if present.

The SM-DP+ SHALL include the data object `lprConfiguration` only if the eUICC indicated `lpaProxySupport`.

The SM-DP+ SHALL include the data object `enterpriseConfiguration` if and only if the Profile is an Enterprise Profile, and only if the eUICC indicated `enterpriseProfilesSupport`.

The SM-DP+ SHALL include the data object `serviceDescription` only if the eUICC indicated `serviceDescriptionSupport`.

The SM-DP+ SHALL include the data object `deviceChangeConfiguration` if and only if the Profile supports Device Change, and only if the eUICC indicated `deviceChangeSupport`.

The `estimatedProfileSize` data object includes an estimated size of the installed Profile in the non-volatile memory, expressed in bytes. The SM-DP+ SHALL include this data object only if the eUICC indicated `estimatedProfileSizeIndicationSupport`. If present, it SHALL NOT be stored in the eUICC.

5.5.4 Function: ReplaceSessionKeys

Related Procedures: Profile Download and Installation

Function Provider Entity: ISD-R

Description:

This function is used to replace the BSP session keys (S-ENC and S-MAC) during the loading of a Bound Profile Package by a new set of session keys (typically the PPK-ENC and PPK-CMAC (section 2.5). Note that both keys are replaced; this function doesn't allow replacement of only one of the session keys.

On reception of this function the eUICC SHALL:

- Verify that the new keys are of same length as the old keys. Otherwise the eUICC SHALL return an error, and the loading of the BPP SHALL be aborted.
- Replace the current session keys with the new set of keys.

Once the function is successfully executed, the eUICC SHALL use this new set of keys for decryption and MAC verification of subsequent BSP payload blocks of data. The key type of the new set of keys is the same as the session keys they replace.

Command data

The command message for this function is encoded in the ASN.1 data object below.

```
-- ASN1START
-- Definition of request message for command ReplaceSessionKeys
ReplaceSessionKeysRequest ::= [38] SEQUENCE { -- tag 'BF26'
-- The new initial MAC chaining value
   initialMacChainingValue OCTET STRING,
-- New session key value for encryption/decryption (PPK-ENC)
   ppkEnc OCTET STRING,
-- New session key value of the session key C-MAC computation/verification (PPK-
MAC)
   ppkCmac OCTET STRING
}
-- ASN1STOP
```

5.5.5 Function: LoadProfileElements

Related Procedures: Profile Download and Installation

Function Provider Entity: ISD-R

Description:

This function is used by the SM-DP+ to provide the Profile Elements defined by eUICC Profile Package specification [5] to the eUICC.

Command messages, response messages and the processing on the eUICC are defined in eUICC Profile Package specification [5].

The eUICC SHALL ignore the ICCID value provided in the 'ProfileHeader' PE.

The eUICC SHALL verify that the following values provided in the Profile Metadata via "ES8+.StoreMetadata" are reflected in the content of EFs of the Profile:

- The ICCID provided in the Profile Metadata is identical to the value of EF_{ICCID}.
- If `profileOwner` is provided in the Profile Metadata:
  - The `mccMnc` value provided in the Profile Metadata SHALL match the MCC and MNC values in EF_{IMSI}.
  - If `gid1` or `gid2` is provided in the Profile Metadata: The corresponding EF_{GID1} or EF_{GID2} SHALL be present and contain the same value as provided in the data object and the related service in EF_{UST} SHALL indicate "available".
  - If `gid1` or `gid2` is not provided in the Profile Metadata: The corresponding service in EF_{UST} for EF_{GID1} or EF_{GID2} SHALL indicate "not available".

Any failure SHALL be indicated by an `installFailedDueToDataMismatch` error.

If the Profile is a Test Profile, the eUICC SHALL check if the key(s) for network authentication follow the requirements defined in section 2.4.5.3.

On any error during the processing of a Profile Element, the Profile installation SHALL stop and the ISD-P and all the related Profile Components SHALL be deleted.

If the Profile is successfully installed, the eUICC SHALL first generate the Profile Installation Result and then as many Notifications as configured in its metadata (`notificationConfigurationInfo`) in the format of `OtherSignedNotification.`

Otherwise, the eUICC SHALL only generate the Profile Installation Result with an error indication.

5.6 ES9+ (LPA -- SM-DP+)

ES9+ is the interface between:

- The LPA entity (more specifically the LPD endpoint)and,

- the SM-DP+ (more specifically the Profile Package Delivery endpoint, which is in charge to deliver the input data from the LPA to the Profile Package Binding function, and deliver the output data from the Profile Package Binding function to the LPA).

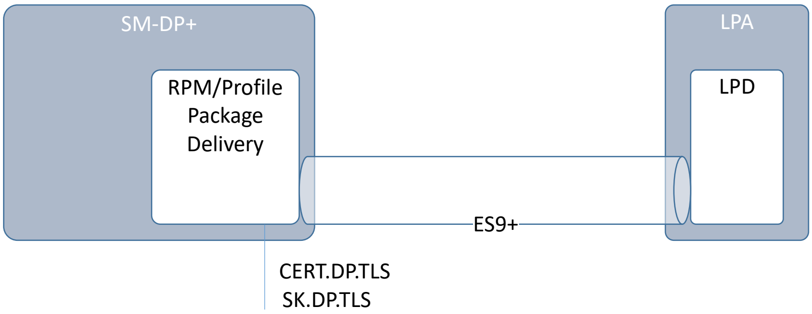


Figure 34: ES9+

The LPA SHALL communicates with the SM-DP+ secured by HTTPS in server authentication mode as described in section 2.6.6.

The format of the TLS Certificates (CERT.DP.TLS) used for TLS connections is described in section 4.5.2.1.

During TLS establishment, the LPA SHALL verify the received CERT.DP.TLS according to section 4.5.2.2. If any of these verifications fail, the TLS connection SHALL be rejected, and the on-going procedure SHALL fail.

5.6.1 Function: InitiateAuthentication

Related Procedures: Profile Download and Installation

Function Provider Entity: SM-DP+

Description:

This function requests the SM-DP+ authentication. This is following the "GetEUICCChallenge" between the eUICC and the LPAd, where the LPAd retrieves material from the eUICC to be provided to the SM-DP+.

On reception of this function call, the SM-DP+ SHALL:

- Verify that the received address matches its own SM-DP+ address, where the comparison SHALL be case-insensitive. Otherwise, the SM-DP+ SHALL return a status code "SM-DP+ Address - Refused".
- If `euiccCiPKIdListForSigningV3` is present, verify it supports one of these eSIM CA RootCA Public Keys against which eUICC signatures can be verified, and select the eSIM CA RootCA Public Key. If the SM-DP+ does not have any other priorities defined, the SM-DP+ SHALL follow the priority order given by the eUICC in `euiccCIPKIdListForSigningV3`. If an eSIM CA public key is selected, then:
  - the public key identifier SHALL be returned in `sessionContext.euiccCiPKIdToBeUsedV3`

    - and in addition: if LPA indicates `certChainV3Support`, `euiccCiPKIdToBeUsed` SHALL be omitted, otherwise it SHALL be present with a zero-length value.
- If no eSIM CA RootCA Public Key has been selected from `euiccCiPKIdListForSigningV3`, verify it supports one of the keys indicated by `euiccCiPKIdListForSigning` (again using its own defined priority or priority from the list). If not, the SM-DP+ SHALL return a status code (Security configuration - Unsupported). The key identifier SHALL be returned in `euiccCiPKIdToBeUsed`, and `sessionContext.euiccCiPKIdToBeUsedV3` SHALL be omitted.

NOTE: A version 2 eUICC only sends `euiccCiPKIdListForSigning`, therefore the SM-DP+ can only select an eSIM CA RootCA Public Key Identifier among this list. A version 3 eUICC can send various combinations of `euiccCiPKIdListForSigning` and `euiccCiPKIdListForSigningV3` (see section 5.7.8). If the SM-DP+ selects an eSIM CA RootCA Public Key identifier in `euiccCiPKIdListForSigning`, the SM-DP+ has to support the verification of the certificate chain Variant O.

- Determine the set of CERT.DPauth.SIG that satisfy the following criteria:
    - Part of a certificate chain ending at one of the eSIM CA RootCA Certificate, whose Public Keys is supported by the eUICC (indicated by `euiccCiPKIdListForVerification`).
    - Using a certificate chain that the eUICC and the LPA both support:
        - If the eUICC indicates `certChainV3VerificationSupport` and the LPA indicates `certChainV3Support`, then it SHALL belong to a chain following one of the Variants O, A, B or C (see section 4.5.2.0b).
        - Otherwise, it SHALL belong to a chain following Variant O.
- Depending on the number of Certificates in the set, do the following:
    - If there is one, selects this CERT.DPauth.SIG.
    - If there are more than one, selects the CERT.DPauth.SIG preferably according to the priority provided by the eUICC for the eSIM CA RootCA Public Keys.
    - If there is none, and the LPA indicated `euiccCiUpdateSupport`, it SHOULD selects its preferred CERT.DPauth.SIG.
    - Otherwise, the SM-DP+ SHALL return a status code "SM-DP+ Certificate - Unavailable".
- If both eUICC and LPA indicate `crlStaplingV3Support`, verify it can provide an up-to-date CRL corresponding to each Certificate in the chain that has a cRLDistributionPoints extension set. The SM-DP+ can retrieve each of them from the Distribution Point(s) indicated in the corresponding Certificate (see section 4.6.4), or from a local repository if the CRL copy is still not outdated. If the SM-DP+ cannot provide the necessary CRL(s), it SHALL return a status code (SM-DP+ Certificate - Unavailable).
- Generate a TransactionID which is used to identify the ongoing RSP session. The TransactionID SHALL be unique within the scope and lifetime of each SM-DP+.

NOTE: TransactionIDs not being reused protects against attacks which replay CancelSession messages.

- Generate a serverChallenge for eUICC authentication attached to the ongoing RSP session.
- Generate a serverSigned1 data object as expected by the eUICC and described in section 5.7.13 "ES10b.AuthenticateServer". If and only if both eUICC and LPA indicate `crlStaplingV3Support`, the SM-DP+ SHALL indicate `crlStaplingV3Used` in `sessionContext`.
- Generate a signature (serverSignature1) as described in section 5.7.13 "ES10b.AuthenticateServer" using the SK related to the selected CERT.DPauth.SIG.

The SM-DP+ MAY perform additional operations, which are out of scope of this specification.

This function SHALL return one of the following:

- A 'Function execution status' with 'Executed-Success', indicating that the RSP session has been successfully initiated at the SM-DP+.
- A 'Function execution status' indicating 'Failed' with a status code as defined in section 5.2.6 or a specific status code as defined in the following table.

Additional Input Data:

Input data name	Description	Type	No.	MOC
`euiccChallenge`	euiccChallenge generated by the eUICC. LPDd can retrieve the euiccChallenge from the eUICC by calling "ES10b.GetEUICCChallenge" (section 5.7.7).	Binary[16]	1	M
`euiccInfo1`	euiccInfo1 of the eUICC. LPDd can retrieve the euiccInfo1 by calling "ES10b.GetEUICCInfo" (section 5.7.8).	Binary(1)	1	M
`smdpAddress`	The SM-DP+ Address as known and provided by the LPA.	FQDN	1	M
`lpaRspCapability`	#SupportedFromV3.0.0# The RSP capability supported by the LPA.	Binary(2)	1	C(3)
NOTE 1: euiccInfo1 SHALL be provided as an encoded EuiccInfo1 data object. NOTE 2: `lpaRspCapability` SHALL be provided as an encoded data object including the tag defined for it in the DeviceInfo data object as defined in section 4.2. NOTE 3: `lpaRspCapability` SHALL be provided by a version 3 or higher compliant LPA.				

Table 35: InitiateAuthentication Additional Input Data

Additional Output Data:

Output data name	Description	Type	No.	MOC
`transactionId`	Transaction ID as generated by the SM-DP+ (section 3.0.1).	Binary[1-16]	1	M

serverSigned1	The data object signed by the SM-DP+.	Binary(1) (3)	1	M
serverSignature1	SM-DP+ signature.	Binary(1)	1	M
euiccCiPKIdToBeUsed	If present and not empty, indicates the Variant O certificate chain related to the Private Key that the eUICC SHALL use for signing.	Binary(1) (6)	1	C(4)
serverCertificate	SM-DP+ Certificate (CERT.DPauth.SIG).	Binary(1)	1	M
otherCertsInChain	#SupportedFromV3.0.0# The remaining part of the CERT.DPauth.SIG Certificate chain (If any). It contains a CERT.DPSubCA.SIG, a CERT.CISubCA.SIG, or both.	Binary(1)	1	C(2)
crlList	#SupportedFromV3.0.0# The list of CRLs needed to verify the revocation status of each Certificate that contains a cRLDistributionPoints extension in the returned Certificate chain.	Binary(1)	1	C(5)

NOTE 1: serverSigned1, serverSignature1, euiccCiPKIdToBeUsed, serverCertificate, otherCertsInChain and crlList are data objects defined in section 5.7.13 (function "ES10b.AuthenticateServer"). They SHALL be returned as encoded data objects including the tags defined for them in the AuthenticateServerRequest data object.

NOTE 2: otherCertsInChain SHALL be omitted if the CERT.DPauth.SIG is directly signed by an eSIM RootCA. If present, this list SHALL contain one or two Certificate(s) according to the Certificate chain variant used by the server (section 4.5.2.0b). This may be extended in future version of this specification.

NOTE 3: sessionContext and serverRspCapability SHALL be omitted in serverSigned1 if the euiccRspCapability was not present in the input data.

NOTE 4: if the eSIM CA RootCA Public Key Identifier has been selected from euiccCiPKIdListForSigningV3, euiccCiPKIdToBeUsed SHALL either be omitted or have a zero-length value as described above.

NOTE 5: crlList SHALL only be present if both eUICC and LPA indicated crlStaplingV3Support.

NOTE 6: euiccCiPKIdToBeUsed is not protected by a signature. However, the SM-DP+ assures its integrity by verifying that the eUICC Certificate in the subsequent call to ES9+.AuthenticateClient chains to the root public key identified by this value.

Table 36: InitiateAuthentication Additional Output Data

<u>Specific Status Codes</u>

Subjectcode	Subject	Reasoncode	Reason	Description
8.8.1	SM-DP+ Address	3.8	Refused	Invalid SM-DP+ Address.
8.8.2	Security configuration	3.1	Unsupported	None of the proposed Public Key Identifiers is supported by the SM-DP+.
8.8.4	SM-DP+ Certificate	3.7	Unavailable	The SM-DP+ has no CERT.DPauth.SIG which chains to one of the eSIM CA RootCA Certificate with a Public Key supported by the eUICC.

Table 37: InitiateAuthentication Specific Status codes

5.6.2 Function: GetBoundProfilePackage

Related Procedures: Profile Download and Installation

Function Provider Entity: SM-DP+

Description:

This function SHALL be called to request the delivery and the binding of a Profile Package for the eUICC.

This function is correlated to a previous normal execution of an "ES9+.AuthenticateClient" function through a TransactionID delivered by the SM-DP+.

On reception of this function call, the SM-DP+ SHALL:

- Verify that the received transactionId is known and relates to an ongoing RSP session. If not, the SM-DP+ SHALL return a status code "TransactionId - Unknown".
- Verify the `euiccSignature2` computed over `euiccSigned2` and `smdpSignature2` using the PK.EUICC.SIG attached to the ongoing RSP session. If the signature is invalid, the SM-DP+ SHALL return a status code "eUICC - Verification failed".
- Check if this order requires a Confirmation Code verification; if yes, the SM-DP+ SHALL verify that the received Hashed Confirmation Code matches the value known by the SM-DP+. If the Confirmation Code is not received, the SM-DP+ SHALL return a status code "Confirmation Code - Mandatory Element Missing". If the values do not match, the SM-DP+ SHALL increment the number of incorrect Confirmation Code attempts. If the maximum number of incorrect attempts for Confirmation Code verification is not exceeded the SM-DP+ SHALL return a status code "Confirmation Code - Refused". If it is exceeded, the corresponding Profile download order SHALL be terminated and the SM-DP+ SHALL return a status code "Confirmation Code - Maximum number of attempts exceeded".

If the Bound Profile Package has been previously generated for this eUICC, the SM-DP+ SHALL check if the otPK.EUICC.KA provided by the eUICC is the same as the one used to generate this BPP. If so, the BPP can be re-used: only the signature for InitialiseSecureChannel needs to be recalculated.

If the Bound Profile Package has been previously generated for this eUICC, but the otPK.EUICC.KA provided by the eUICC is different than the one previously used to generate this BPP, the SM-DP+ SHALL re-bind the PPP as described below.

To bind the PPP, the SM-DP+ SHALL:

- Attach the otPK.EUICC.KA to the ongoing RSP session.
- Link the Profile to the EID of CERT.EUICC.SIG if the Profile is not already linked.
- Generate a one-time KA key pair (otPK.DP.KA, otSK.DP.KA) for key agreement using the parameters indicated by the `subjectPublicKeyInfo.algorithmIdentifier.parameters` field of CERT.DPpb.SIG.
- Generate the session keys (S-ENC and S-MAC) and the initial MAC chaining value using the CRT, otPK.EUICC.KA and otSK.DP.KA.
- Generate the Profile Metadata of the Profile.
- Generate the Bound Profile Package as described in (section 2.5.4), optionally including the Profile Protection Keys (PPK).
- Erase otSK.DP.KA immediately once BPP is generated.

The SM-DP+ MAY perform additional operations, which are out of scope of this specification.

This function SHALL return one of the following:

- A 'Function execution status' with 'Executed-Success' indicating that the BoundProfilePackage has been successfully built and is part of the output data.
- A 'Function execution status' indicating 'Failed' with a status code as defined in section 5.2.6 or a specific status code as defined in the following table.

Additional Input Data:

Input data name	Description	Type	No.	MOC
`transactionId`	Transaction ID as generated by the SM-DP+ (section 3.0.1).	Binary[1-16]	1	M
`prepareDownloadResponse`	PrepareDownloadResponse data object defined in section 5.7.5.	Binary[1]	1	M
NOTE 1: prepareDownloadResponse SHALL be provided as an encoded PrepareDownloadResponse data object				

Table 38: GetBoundProfilePackage Additional Input Data

Additional Output Data:

Output data name	Description	Type	No.	MOC
`transactionId`	Transaction ID as generated by the SM-DP+ (section 3.0.1).	Binary[1-16]	1	M

`boundProfilePackage`	Bound Profile Package data object to be transferred to the eUICC using "ES10b.LoadBoundProfilePackage" (section 5.7.6).	Binary(1)	1	M
NOTE 1: boundProfilePackage SHALL be provided as an encoded BoundProfilePackage data object				

Table 39: GetBoundProfilePackage Additional Output Data

Specific Status Codes

Subjectcode	Subject	Reasoncode	Reason	Description
8.1	eUICC	6.1	Verification Failed	eUICC signature is invalid.
8.2	Profile	3.7	Unavailable	BPP is not available for a new binding.
8.10.1	TransactionId	3.9	Unknown	The RSP session identified by the TransactionID is unknown.
8.2.7	Confirmation Code	2.2	Mandatory Element Missing	Confirmation Code is missing.
8.2.7	Confirmation Code	3.8	Refused	Confirmation Code is refused.
8.2.7	Confirmation Code	6.4	Maximum number of attempts exceeded	The maximum number of incorrect attempts for the Confirmation Code has been exceeded.

Table 40: GetBoundProfilePackage Specific status codes

5.6.3 Function: AuthenticateClient

Related Procedures: Common Mutual Authentication

Function Provider Entity: SM-DP+

Description:

This function SHALL be called by the LPA to request the authentication of the eUICC by the SM-DP+.

This function is correlated to a previous normal execution of an "ES9+.InitiateAuthentication" function through a Transaction ID delivered by the SM-DP+.

On reception of this function call, the SM-DP+ SHALL:

- Verify the validity of the eUICC certificate chain, as defined in section 4.5.2.2. If the eUICC Certificate (or any of the certificates in the chain) is invalid or expired, the SM-DP+ SHALL return a status code "eUICC Certificate - Verification failed" or "eUICC Certificate - Expired" respectively. If any of the certificates is missing in the chain, the SM-DP+ SHALL return a status code "eUICC Certificate - Verification failed".
- Verify that the Root Certificate of the eUICC certificate chain corresponds to the `euiccCiPKIdToBeUsed` or `euiccCiPKIdToBeUsedV3` that the SM-DP+ selected

when executing the "ES9+.InitiateAuthentication" function. If it doesn't correspond, or if the chain variant doesn't match, the SM-DP+ SHALL return a status code "CI Public Key - Unknown".

- Verify the eUICC signature (euiccSignature1) using the PK.EUICC.SIG as described in section 5.7.13 "ES10b.AuthenticateServer". Otherwise, the SM-DP+ SHALL return a status code "eUICC - Verification failed".
- Verify that the transactionId is known and relates to an ongoing RSP session. Otherwise, the SM-DP+ SHALL return a status code "TransactionId - Unknown".
- Verify that the serverChallenge attached to the ongoing RSP session matches the serverChallenge returned by the eUICC. Otherwise, the SM-DP+ SHALL return a status code "eUICC - Verification failed".
- If the SM-DP+ determines that the related Profile download order or RPM order has expired before being downloaded, the SM-DP+ SHALL return a status code "Download order - Time to Live Expired".
- If the maximum number of attempts for Profile or RPM download has been exceeded, the corresponding Profile download order or RPM order SHALL be terminated, and the SM-DP+ SHALL return a status code "Download order - Maximum number of attempts exceeded".
- If `euiccRspCapability.euiccRspCapInInfo1` is set to '1', verify that `euiccRspCapability` was present in `euiccInfo1` as received in ES9+.InitiateAuthentication and that `euiccRspCapability` in `euiccInfo2` matches the value received in `euiccInfo1`. Otherwise, the SM-DP+ SHALL return a status code "eUICC - Value has Changed".
- Verify that `lpaRspCapability` in `deviceInfo` matches the value received in ES9+.InitiateAuthentication. Otherwise, the SM-DP+ SHALL return a status code "LPA - Value has Changed".

Additionally, if the expired/terminated Profile download order or RPM order was accompanied by an Event Registration, the SM-DP+ SHOULD send "ES12.DeleteEvent" to the corresponding SM-DS to delete the relevant Event Record.

In addition, the SM-DP+ SHALL perform additional verification depending on the use case where this function is involved and the received `ctxParams1`.

The SM-DP+ SHALL identify the type of operation requested by the LPA checking the content of `ctxParams1`.

- If it contains `ctxParamsForCommonAuthentication`, the SM-DP+ SHALL refer to the value of the `operationType`:
  - If `operationType` is not present, the SM-DP+ SHALL regard this as a Profile download request from the LPA that conforms to version 2 of this specification.
  - If `profileDownload` bit is set, the SM-DP+ SHALL regard this as a Profile download request.
  - If `rpm` bit is set, the SM-DP+ SHALL regard this as an RPM download request.

    - If all bits in the `operationType` are set, or more than one Profile download orders and/or RPM orders are pending, the SM-DP+ SHALL prioritize the Profile downloads and/or RPMs as per Operator's request.
- If it contains `ctxParamsForDeviceChange`, the SM-DP+ SHALL regard this as a Device Change request.
- If it contains `ctxParamsForProfileRecovery`, the SM-DP+ SHALL regard this as a Profile Recovery request.
- Otherwise, the SM-DP+ SHALL regard this as an error and SHALL return a status code "Function – Unsupported".

NOTE 1: In the context of a Device Change, a Profile download request from the new Device MAY additionally contain a `deleteNotificationForDc`. The presence of this field does not indicate a separate operation.

NOTE 2: If the `operationType` is set to `profileDownload` only, the data object will be absent in the DER encoding, which makes the function call backwards compatible to how it is defined in version 2 of this specification.

<u>Beginning of Profile Download operation</u>

The SM-DP+ SHALL:

- Verify there is a pending Profile download order for the incoming eUICC. For that, the SM-DP+ SHALL:
    - If there is a pending Profile download order associated with this EID:
        - If `ctxParamsForCommonAuthentication` contains a `MatchingId` then verify that it matches the `MatchingID` for this pending Profile download order. Otherwise:
            - The SM-DP+ SHALL return a status code "MatchingID - Refused"
            - If `ctxParamsForCommonAuthentication` contains a `matchingIdSource` data object set to an SM-DS OID, the SM-DP+ SHOULD delete the Event Record from the SM-DS identified by that OID (see Note 4).
        - If the `MatchingID` is missing in `ctxParamsForCommonAuthentication`, any pending Profile download order associated to this EID MAY be selected.
    - If there is no pending Profile download order associated with this EID:
        - If `ctxParamsForCommonAuthentication` contains a `MatchingId`, then:
            - verify that there is a pending Profile download order associated with this MatchingID, and the pending Profile download order is not associated with an EID. Otherwise:
                - The SM-DP+ SHALL return a status code "MatchingID - Refused".
                - If `ctxParamsForCommonAuthentication` contains a `matchingIdSource` data object set to an SM-DS OID, the SM-

DP+ SHOULD delete the Event Record from the SM-DS identified by that OID (see Note 4).

    - If the `MatchingID` is missing in `ctxParamsForCommonAuthentication`, this SHALL be considered as verification failure and a status code "EID - Refused" SHALL be returned.

- In case of a Profile download related to a Device Change that requires the deletion of the installed Profile, verify that either the Delete Notification of the installed Profile has been received or the previous Device Change response included encrypted Device Change data. If so, the SM-DP+ SHALL ignore the Delete Notification for Device Change received in this function call (if any). Otherwise, the SM-DP+ SHALL:
    - If the SM-DP+ supports the processing of the Delete Notification for Device Change:
        - Retrieve the Delete Notification for Device Change contained in the `deleteNotificationForDc`. If it is not received in this function call, the SM-DP+ SHALL return an errors status "Profile – Not Allowed".
        - Retrieve CERT.EUICC.SIG, CERT.EUM.SIG, CERT.EUMSubCA.SIG (if it exists), and the ICCID attached to the pending Profile download order. If any retrieval fails, the SM-DP+ SHALL return an error status "Profile – Not Allowed".
        - Restore the Delete Notification by combining CERT.EUICC.SIG, CERT.EUM.SIG, and CERT.EUMSubCA.SIG (if it exists) with the Delete Notification for Device Change retrieved above.
        - Validate the restored Delete Notification by verifying the eUICC signature and checking if the ICCID contained in the Delete Notification matches to the ICCID attached to the pending Profile download order. If either fails, the SM-DP+ SHALL return an error status "Profile – Not Allowed"

          NOTE: The Recipient Address of the restored Delete Notification may not be the FQDN of the SM-DP+ processing this function call. In such a case, it is out of scope of this document how the SM-DP+ interacts with the Notification receiver.
    - Otherwise, return an error status "Profile – Not Allowed".
- Identify the Profile corresponding to the pending Profile download order.
- Verify that the identified Profile has been released (Profile state is Released, see section 3.1.6). Otherwise, the SM-DP+ SHALL return a status code "Profile - Not allowed".
- Increment the number of attempts for the Profile.
- Perform the following eligibility checks:

    - Check if the eUICC can install one more Profile. Otherwise, the SM-DP+ SHALL return a status code "eUICC - Insufficient memory".
    - If the Profile is a Test Profile, the SM-DP+ SHALL check if the Device is operating in Device Test Mode. Otherwise, the SM-DP+ SHALL return a status code "Profile type - Stopped on warning".
    - If the Profile is an Enterprise Profile, the SM-DP+ MAY check if the Device is an Enterprise Capable Device and if the eUICC supports Enterprise

Profiles. If this check fails, the SM-DP+ SHALL return a status code "Profile type - Stopped on warning".
  - If the Profile Owner has disallowed Profile download to Field-Test eUICCs: If the target eUICC indicates V255.255.255 in `ppVersion`, the SM-DP+ SHALL return a status code "Profile Type – Stopped on warning"
  - The SM-DP+ MAY perform additional Eligibility checks as described in Annex F.

- Attach the PK.EUICC.SIG to the ongoing RSP session.
- Verify if this order requires a Confirmation Code verification. If yes, the SM-DP+ SHALL set the ccRequiredFlag data field of the `smdpSigned2` data object to true.
- Determine if the Profile is already bound to the EID from a previous unsuccessful download attempt. If yes, the SM-DP+ MAY include the otPK.EUICC.KA obtained in the previous session in the `smdpSigned2` data object.
- Determine if an RPM Package is pending for the `eid`. If yes, the SM-DP+ MAY include the `rpmPending` in the `smdpSigned2` data object.
- Generate an `smdpSigned2` data object as defined in "ES10b.PrepareDownloadRequest".
- Compute the `smdpSignature2` over the concatenated data objects `smdpSigned2` and `euiccSignature1` using the SK.DPpb.SIG.
- Generate the Profile Metadata of the Profile. If the Profile Metadata contains Profile Policy Rules and the eUICC Info indicates `EuiccRspCapability.serviceProviderMessageSupport`, then the SM-DP+ MAY include a Service Provider message to be displayed to the End User. If the Device Info includes language preferences, the SM-DP+ SHOULD provide a Service Provider message using the most preferred language that it can support. The method by which the SM-DP+ receives this/these localised message(s) from the Operator/Service Provider is out of the scope of this specification.

NOTE 3: When providing messages for delivery by the SM-DP+, the Operator/Service Provider should consider challenges displaying lengthy text on a device with a limited display.

NOTE 4: Deleting the Event Record corresponding to a MatchingId that is no longer valid allows to avoid repeated attempts to download the same package. The SM-DP+ should consider whether it is appropriate to delete an Event Record corresponding to a MatchingId that it has never known.

End of Profile Download operation

Beginning of RPM operation

The SM-DP+ SHALL:

- Verify there is a pending RPM order for the incoming eUICC. For that, the SM-DP+ SHALL:

  - If there is a pending RPM order associated with this EID and ICCID is received:

    - If `ctxParamsForCommonAuthentication` contains a `MatchingId`, then:
      - Verify that it matches the `MatchingID` for this pending RPM order. Otherwise:
        - The SM-DP+ SHALL return a status code "MatchingID - Refused".
        - If `ctxParamsForCommonAuthentication` contains a `matchingIdSource` data object set to an SM-DS OID, the SM-DP+ SHOULD delete the Event Record from the SM-DS identified by that OID (see Note 4 above).
      - Verify that the received ICCID matches the target Profile of this pending RPM order. Otherwise, the SM-DP+ SHALL return a status code "ICCID - Refused".
    - If the `MatchingID` is missing in `ctxParamsForCommonAuthentication`, any pending RPM order for the target Profile associated to this EID MAY be selected.

  - If there is a pending RPM order associated with this EID and ICCID is not received:

    - If `ctxParamsForCommonAuthentication` contains a `MatchingId`, then:
      - Verify that it matches the `MatchingID` for this pending RPM order. Otherwise:
        - The SM-DP+ SHALL return a status code "MatchingID - Refused"
        - If `ctxParamsForCommonAuthentication` contains a `matchingIdSource` data object set to an SM-DS OID, the SM-DP+ SHOULD delete the Event Record from the SM-DS identified by that OID (see Note 4 above).
    - If the MatchingID is missing in `ctxParamsForCommonAuthentication`, any pending RPM order associated to this EID MAY be selected.

  - If there is no pending RPM order associated with this EID, this SHALL be considered as verification failure and a status code "EID - Refused" SHALL be returned.

- Identify the RPM Package corresponding to the pending RPM order.
- Increment the number of attempts for the Profile.
- Determine if another RPM Package is pending for the `eid`. If yes, the SM-DP+ MAY include `rpmPending` in the `smdpSigned3` data object.
- Generate an `smdpSigned3` data object as defined in "ES10b.LoadRpmPackage".
- Compute the `smdpSignature3` over the concatenated data objects `smdpSigned3` and `euiccSignature1` using the SK.DPauth.SIG.

<u>End of RPM operation</u>

<u>Beginning of Device Change operation</u>

The SM-DP+ SHALL:

- Return an error status "Device Change – Unsupported" if the SM-DP+ does not support Device Change.
- Identify the Profile for Device Change by ICCID contained in the `ctxParamsForDeviceChange`. If the Profile cannot be identified, the SM-DP+ SHALL return an error status "ICCID – Unknown".
- Verify that the identified Profile is associated with the EID of the incoming eUICC, i.e., the eUICC of the old Device. If the Profile is not associated with the EID, the SM-DP+ SHALL return an error status "EID – Refused".
- Call ES2+.HandleDeviceChangeRequest function with the ICCID and, if present in the `ctxParamsForDeviceChange` data object, the EID and/or TAC of the new Device contained in the `ctxParamsForDeviceChange` as per Service Provider's configuration. If the SM-DP+ receives any error status, the SM-DP+ SHALL return the received error status.
- Verify that the identified Profile is eligible for Device Change. If the Profile is not eligible for Device Change, the SM-DP+ SHALL return an error status "Device Change – Not Allowed".
- Attach the EID of the new device, if present, to the ongoing RSP session.
- Attach the CERT.EUICC.SIG, CERT.EUM.SIG, and CERT.EUMSubCA.SIG (if exists) received in this function call to the ongoing RSP session.
- If a Confirmation Code verification is required for the Device Change of the identified Profile, set the `ccRequiredFlag` data field of the `smdpSigned4` data object to true; otherwise set to false.
- Generate an `smdpSigned4` data object as defined in "ES10b.PrepareDeviceChangeRequest".
- Compute the signature `smdpSignature4` over the concatenated data objects `smdpSigned4` and `euiccSignature1` using SK.DPauth.SIG.
- Prepare Service Provider Message for Device Change if either configured as per Service Provider's configuration or received in ES2+.HandleDeviceChangeRequest.
- Notify the Service Provider of the Device Change progress by calling "ES2+.HandleNotification" function with the identification 'Device Change request (12)' and an the notificationEventStatus indicating 'Execution-Success' if configured as per Service Provider's configuration.

End of Device Change operation

Beginning of Profile Recovery operation

The SM-DP+ SHALL:

- Identify the Profile by ICCID contained in the `ctxParamsForProfileRecovery`, and verify that the Profile was processed for Device Change previously. If the Profile cannot be identified or the Profile was not processed for Device Change previously, the SM-DP+ SHALL return an error status "ICCID – Unknown".
- Verify that the identified Profile is associated with the EID of the incoming eUICC, i.e., the eUICC of the old Device. If the Profile is not associated with the EID of the eUICC of the old Device, the SM-DP+ SHALL return an error status "EID – Refused".

- Verify that there was a permanent error whilst installing the prepared Profile on the new Device for Device Change, corresponding to the ICCID contained in the `ctxParamsForProfileRecovery`. If verification fails, the SM-DP+ SHALL return a status code "Profile – Not allowed".
- Prepare a Profile for recovery and the associated Activation Code for the old Device. The SM-DP+ MAY interact with the Service Provider for the Profile preparation.
- Generate an `smdpSigned4` data object including the Activation Code for Profile Recovery as defined in "ES10b.PrepareDeviceChangeRequest".
- Compute the signature `smdpSignature4` over the concatenated data objects `smdpSigned4` and `euiccSignature1` using SK.DPauth.SIG.

End of Profile Recovery operation

The SM-DP+ MAY perform additional operations, which are out of scope of this specification.

This function SHALL return one of the following:

- A 'Function execution status' with 'Executed-Success' indicating that the eUICC has been successfully authenticated.
- A 'Function execution status' indicating 'Failed' with a status code as defined in section 5.2.6 or a specific status code as defined in the following table.

Additional Input Data:

Input data name	Description	Type	No.	MOC
`transactionId`	Transaction ID as generated by the SM-DP+ (section 3.0.1).	Binary[1-16]	1	M
`authenticateServerResponse`	Authenticate Server Response.	Binary(1)	1	M
`deleteNotificationForDc`	Delete Notification for Device Change	Binary	1	O(2)
NOTE 1: AuthenticateServerResponse data object defined in section 5.7.13 (function "ES10b.AuthenticateServer").				

Table 41: AuthenticateClient Additional Input Data

Additional Output Data:

Output data name	Description	Type	No.	MOC
`transactionId`	Transaction ID as generated by the SM-DP+ (section 3.0.1).	Binary[1-16]	1	M
`profileMetadata`	Profile Metadata for the purpose of display by the LPA.	Binary(1)	1	C(4)
`smdpSigned2`	The data to be signed by the SM-DP+.	Binary(1)	1	C(4)
`smdpSignature2`	SM-DP+ signature.	Binary(1)	1	C(4)
`smdpCertificate`	SM-DP+ Certificate (CERT.DPpb.SIG).	Binary(1)	1	C(4)
`smdpSigned3`	The data to be signed by the SM-DP+.	Binary(2)	1	C(5)
`smdpSignature3`	SM-DP+ signature.	Binary(2)	1	C(5)

`smdpSigned4`	The data to be signed by the SM-DP+.	Binary(3)	1	C(6)
`smdpSignature4`	SM-DP+ signature.	Binary(3)	1	C(6)
`serviceProviderMessageForDc`	Service Provider Message for the purpose of display by the LPA to provide information for Device Change as defined in Section 6.6.2.2.	Binary	1	O(6)
NOTE 1: `profileMetadata` is the data object `StoreMetadataRequest` defined in section 5.5.3 (function "ES8+.StoreMetadata"); `smdpSigned2`, `smdpSignature2` and `smdpCertificate` are data objects defined in section 5.7.5 (function "ES10b.PrepareDownload"). They SHALL be returned as encoded data objects including the tags defined for them in the `StoreMetadataRequest`/`PrepareDownloadRequest` data object. NOTE 2: `smdpSigned3` and `smdpSignature3` are the data objects defined in section 5.7.25 (function "ES10b.LoadRpmPackage"); they SHALL be returned as encoded data objects including the tags defined for them in the `LoadRpmPackageRequest` data object. NOTE 3: `smdpSigned4` and `smdpSignature4` are the data objects defined in section 5.7.26 (function "ES10b.PrepareDeviceChange"); they SHALL be returned as encoded data objects including the tags defined for them in the `PrepareDeviceChangeRequest` data object. NOTE 4: `profileMetadata`, `smdpSigned2`, `smdpSignature2` and `smdpCertificate` SHALL be provided when this function is called in the context of the Profile Download and Installation procedure as described in section 3.1.3. NOTE 5: `smdpSigned3` and `smdpSignature3` SHALL be provided when this function is called in the context of the RPM Download and Execution procedure as described in section 3.7.2. NOTE 6: `smdpSigned4` and `smdpSignature4` SHALL be provided when this function is called in the context of the Device Change procedure as described in section 3.11.1. and the Profile Recovery procedure as described in section 3.11.2. Additionally, `serviceProviderMessageForDc` MAY be provided when this function is called in the context of the Device Change procedure as described in section 3.11.1.				

Table 42: AuthenticateClient Additional Output Data

Specific Status Codes

Subjectcode	Subject	Reasoncode	Reason	Description
8.1.3	eUICC Certificate	6.1	Verification Failed	eUICC Certificate or any Certificate in the trust chain is invalid.
8.1.3	eUICC Certificate	6.3	Expired	eUICC Certificate or any Certificate in the trust chain has expired.
8.1	eUICC	6.1	Verification Failed	eUICC signature is invalid or serverChallenge is invalid.
8.1	eUICC	4.8	Insufficient Memory	eUICC does not have sufficient space for this Profile.
8.11.1	CI Public Key	3.9	Unknown	Unknown eSIM CA RootCA Public Key. The eSIM CA is not a trusted root for the SM-DP+, or not the root elected by the SM-DP+ in InitiateAuthentication response.
8.2	Profile	1.2	Not allowed	Profile has not yet been released.
8.10.1	TransactionId	3.9	Unknown	The RSP session identified by the TransactionID is unknown.

8.2.6	MatchingID	3.8	Refused	MatchingID (AC_Token or EventID) does not match a valid order.
8.1.1	EID	3.8	Refused	EID doesn’t match the expected value, or there is no pending RSP session for this eUICC.
8.2.5	Profile Type	4.3	Stopped on warning	No eligible Profile for this eUICC/Device.
8.8.5	Download order	4.10	Time to Live Expired	The Download order has expired.
8.8.5	Download order	6.4	Maximum number of attempts exceeded	The maximum number of attempts for the Profile download order has been exceeded.
8.1	eUICC	3.11	Value has Changed	#SupportedFromV3.0.0# eUICC RSP Capabilities have changed compared to those previously received.
8.8.6	RPM Order	4.10	Time to Live Expired	#SupportedFromV3.0.0# The RPM order has expired.
8.8.6	RPM Order	6.4	Maximum number of attempts exceeded	#SupportedFromV3.0.0# The maximum number of attempts for the RPM Package has been exceeded.
8.12	LPA	3.11	Value has Changed	#SupportedFromV3.0.0# LPA RSP Capabilities have changed compared to those previously received.
8.10.3	Device Change	1.2	Not Allowed	#SupportedForDcV3.0.0# Device Change is not allowed for the identified Profile.
8.10.3	Device Change	3.1	Unsupported	#SupportedForDcV3.0.0# Device Change is not supported by the SM-DP+.
8.2.1	Profile ICCID	3.9	Unknown	#SupportedForDcV3.0.0# The Profile identified by the ICCID is unknown to the SM-DP+.
1.6	Function	3.1	Unsupported	#SupportedFromV3.0.0# The requested function is not supported by the SM-DP+.

Table 43: AuthenticateClient Specific Status Codes

5.6.4 Function: HandleNotification

Related Procedures: Notifications

Function Provider Entity: SM-DP+

Description:

This function SHALL be called by the LPA to notify the SM-DP+ that a Profile Management Operation has successfully been performed on the eUICC.

The SM-DP+ SHALL manage the Notification according to section 3.5 and acknowledge the LPA of the processing.

The SM-DP+ MAY perform additional operations which are out of scope of this specification.

Additional Input Data:

Input data name	Description	Type	No.	MOC
`pendingNotification`	PendingNotification data object as defined in section 5.7.10	Binary(1)	1	M
NOTE 1: pendingNotification SHALL be provided as an encoded PendingNotification data object				

Table 44: HandleNotification Additional Input Data

Additional Output Data:

No additional output data.

5.6.5 Function: CancelSession

Related Procedures: Profile Download and Installation, Remote Profile Management

Function Provider Entity: SM-DP+

Description:

This function is to request the cancellation of an on-going RSP session upon a decision of the End User. This function MAY be used in different procedures.

This function is correlated to a previous normal execution of an "ES9+.AuthenticateClient" function through a transactionId delivered by the SM-DP+.

On reception of this function call, the SM-DP+ SHALL:

- Verify that the received `transactionId` is known and relates to an ongoing RSP session. Otherwise, the SM-DP+ SHALL return a status code "TransactionId - Unknown".
- Verify the eUICC signature (`euiccCancelSessionSignature`) using the PK.EUICC.SIG attached to the ongoing RSP session as described in (section 5.7.14 "ES10b.CancelSession"). Otherwise, the SM-DP+ SHALL return a status code "eUICC - Verification Failed".
- Verify that the received `smdpOid` corresponds to the SM-DP+ (i.e., is the same value as the one contained in the CERT.DPauth.SIG used during the Common Mutual Authentication Procedure). Otherwise, the SM-DP+ SHALL return a status code "SM-DP+ - Invalid Association".

The SM-DP+ SHALL perform additional operations depending on the context and the reason received, as described hereunder.

When used within a Profile Download and Installation procedure, and if the cancel session reason contained in `euiccCancelSessionSigned` indicates a terminal code (see section 5.7.14), the SM-DP+ SHALL:

1. Notify the Operator using the function "ES2+.HandleNotification" function with the identification of the step reached in the on-going procedure and an operation status indicating 'Failed' with status code according to mapping given in section 5.3.5.
2. Terminate the corresponding pending download process.
3. If required, execute the SM-DS Event Deletion procedure described in section 3.6.3.

NOTE: The operations 1), 2) and 3) are described as performed in the context of this function execution. Alternatively they MAY be done asynchronously by the SM-DP+. Operation 2) and 3) MAY not be performed depending on the agreed SM-DP+ behaviour with the Operator. If the operations are not performed, the Operator has the responsibility to take care of the management of the Download Order, e.g., by calling the "ES2+.CancelOrder" on reception of the notification "ES2+.HandleNotification".

The SM-DP+ MAY perform additional operations, which are out of scope of this specification.

This function SHALL return one of the following:

- A 'Function execution status' with 'Executed-Success' indicating that the RSP session has been cancelled.
- A 'Function execution status' indicating 'Failed' with a status code as defined in section 5.2.6 or a specific status code as defined in the following table.

Additional Input Data:

Input data name	Description	Type	No.	MOC
`transactionId`	Transaction ID as generated by the SM-DP+ (section 3.0.1).	Binary[1-16]	1	M
`cancelSessionResponse`	Defined in "ES10b.CancelSession" function, section 5.7.14	Binary(1)	1	M
NOTE 1: cancelSessionResponse SHALL be provided as an encoded `CancelSessionResponse` data object				

Table 45: CancelSession Additional Input Data

Additional Output Data:

No output data.

Specific Status Codes

Subject Code	Subject	Reason code	Reason	Description
8.10.1	TransactionId	3.9	Unknown	The RSP session identified by the TransactionID is unknown.
8.1	eUICC	6.1	Verification Failed	eUICC signature is invalid.
8.8	SM-DP+	3.10	Invalid Association	The provided SM-DP+ OID is invalid.

Table 46: CancelSession Specific status codes

5.6.6 Function: ConfirmDeviceChange

Related Procedures: Device Change

Function Provider Entity: SM-DP+

Description:

This function is to deliver the confirmation result of the Device Change upon a decision of the End User.

This function is correlated to a previous normal execution of an "ES9+.AuthenticateClient" function in the context of the Device Change through a `transactionId` delivered by the SM-DP+.

On reception of this function call, the SM-DP+ SHALL:

- Verify that the received `transactionId` is known and relates to an ongoing RSP session. Otherwise, the SM-DP+ SHALL return a status code "TransactionId – Unknown".
- Verify the `euiccSignature3` computed over `euiccSigned3` and `smdpSignature4` using the PK.EUICC.SIG attached to the ongoing RSP session. If the signature is invalid, the SM-DP+ SHALL return a status code "eUICC – Verification failed".
- If Confirmation Code verification is required: verify that the received Hashed Confirmation Code matches the expected hash value as follows:
  - If the Hashed Confirmation Code is not received, the SM-DP+ SHALL return a status code "Confirmation Code – Mandatory Element Missing".
  - The SM-DP+ SHALL calculate the expected hash value by using the Confirmation Code value known by the SM-DP+ and TransactionId.

    expected hash value = SHA256(SHA256(Confirmation Code) | TransactionId)

    If the value does not match, the SM-DP+ SHALL increment the number of incorrect Confirmation Code attempts. If the maximum number of incorrect attempts for Confirmation Code verification is not exceeded, the SM-DP+ SHALL return a status code "Confirmation Code - Refused". If it is exceeded, the corresponding Device Change procedure SHALL be terminated and the SM-DP+ SHALL return a status code "Confirmation Code - Maximum number of attempts exceeded".

1. If configured by the Service Provider or the Service Provider set the isNewProfileRequired to true in the response to ES2+.HandleDeviceChangeRequest function, notify the Service Provider using "ES2+.HandleNotification" function with the notificationEvent indicating 'Device Change confirmation (13)' and the notificationEventStatus indicating 'Executed-Success'.
2. If the Service Provider set the isNewProfileRequired to true in the response to ES2+.HandleDeviceChangeRequest function or if it is configured by the Service Provider, wait for the completion of the Download Preparation Process, as defined in 3.1.1.2 and optionally the Subscription Activation Process, as defined in 3.1.1.4.
   If the Service Provider set the isNewProfileRequired to false in the response to ES2+.HandleDeviceChangeRequest function or if it is configured by the Service Provider, the SM-DP+ SHALL:
   - Prepare a Profile for download with the Profile Package of the same Profile Package.
   - If an EID was provided in the Device Change Request of a previous normal execution of an "ES9+.AuthenticateClient" function, link the prepared Profile download with the EID.
   - Prepare the Activation Code by either generating it on behalf of the Service Provider or being provided by the Service Provider.
   - Associate the Activation Code Token of the Activation Code to the Profile for Download.
3. Prepare deviceChangeData and append the Service Provider Message for Device Change if configured by the Service Provider.
   - If the deletion of the installed Profile is required, the SM-DP+ SHALL include a deleteOldProfile data object in the deviceChangeData. The SM-DP+ SHALL include a deleteOldProfile data object if the same Profile Package was prepared for Profile download.
   - If the SM-DP+ supports the processing of the Delete Notification for Device Change, the SM-DP+ SHALL:
     - Include a deleteNotificationForDcSupport and notificationAddress data object in the deviceChangeResponse.
     - Attach the ICCID of the Profile that has to be deleted and the certificates attached to the ongoing RSP session (i.e., CERT.EUICC.SIG, CERT.EUM.SIG and CERT.EUMSubCA.SIG as described in section 5.6.3) to the prepared Profile download.
   - If the SM-DP+ supports the Profile Recovery of the deleted Profile in the old Device, the SM-DP+ SHALL:
     - Include a profileRecoverySupport and profileRecoveryValidityPeriod data object in the deviceChangeResponse.
     - Maintain the association of the deleted Profile and the EID of the old Device until the expiration of time indicated in profileRecoveryValidityPeriod, successful Device Change, or successful Profile Recovery, whichever comes first.

3a. If the eUICC indicates `encryptedDeviceChangeDataSupport` and the deletion of the installed Profile is required, the SM-DP+ SHALL:

- Generate a one-time KA key pair (otPK.DP.KA, otSK.DP.KA) for key agreement using the parameters indicated by the `subjectPublicKeyInfo.algorithmIdentifier.parameters` field of CERT.DPauth.SIG.
- Generate the session keys (S-ENC and S-MAC) and the initial MAC chaining value using the CRT, otPK.EUICC.KA and otSK.DP.KA.
- Generate encrypted and MACed `sequenceOf87` with the `deviceChangeData` TLVs, and erase otSK.DP.KA.

4. Generate an `smdpSigned5` data object as defined in section 5.7.27, "ES10b.VerifyDeviceChange".
5. Compute the `smdpSignature5` over the concatenated data objects `smdpSigned5` and `euiccSignature3` using the SK.DPauth.SIG.
6. If configured by the Service Provider, notify the Service Provider using "ES2+.HandleNotification" function with the notificationEvent indicating 'Profile preparation for Device Change (15)' and the notificationEventStatus indicating 'Executed-Success'.

NOTE: Depending on the agreed behaviour between the Service Provider and the SM-DP+, from step 2 to step 5 MAY be performed by the Service Provider using "ES2+.DownloadOrder", "ES2+.ConfirmOrder", and "ES2+.ReleaseProfile" functions as defined in section 3.1.1.

- Otherwise, the SM-DP+ SHALL notify the Operator using the function "ES2+.HandleNotification" function with the notificationEvent indicating "Device Change confirmation Failure (14)' and the notificationEventStatus indicating 'Failed' with status code according to mapping given in section 5.3.5.

The SM-DP+ MAY perform additional operations, which are out of scope of this specification.

This function SHALL return one of the following:

- A 'Function execution status' with 'Executed-Success' indicating that the Device Change request has been successfully processed.
- A 'Function execution status' indicating 'Failed' with a status code as defined in section 5.2.6 or a specific status code as defined in the following table.

Additional Input Data:

Input data name	Description	Type	No.	MOC
`transactionId`	Transaction ID as generated by the SM-DP+ (section 3.0.1).	Binary[1-16]	1	M
`prepareDeviceChangeResponse`	PrepareDeviceCHangeResponse data object defined in section 5.7.5.	Binary$^{(1)}$	1	M
NOTE 1: `prepareDeviceChangeResponse` SHALL be provided as an encoded PrepareDeviceChangeResponse data object.				

Table 46a: ConfirmDeviceChange Additional Input Data

Additional Output Data:

Output data name	Description	Type	No.	MOC
`transactionId`	Transaction ID as generated by the SM-DP+ (section 3.0.1).	Binary[1-16]	1	M
`smdpSigned5`	The data to be signed by the SM-DP+	Binary(1)	1	C
`smdpSignature5`	SM-DP+ signature	Binary(1)	1	C
NOTE 1: `smdpSigned4` and `smdpSignature4` are the data objects defined in section 5.7.26 (function "ES10b.VerifyDeviceChange"); they SHALL be returned as encoded data objects including the tags defined for them in the `VerifyDeviceChangeRequest` data object.				

Table 46b: ConfirmDeviceChange Additional Output Data

Specific Status Codes

Subject code	Subject	Reason code	Reason	Description
8.10.1	TransactionId	3.9	Unknown	The RSP session identified by the TransactionID is unknown.
8.1	eUICC	6.1	Verification Failed	eUICC signature is invalid.
8.2.7	Confirmation Code	2.2	Mandatory Element Missing	Confirmation Code is missing.
8.2.7	Confirmation Code	3.8	Refused	Confirmation Code is refused.
8.2.7	Confirmation Code	3.12	Invalid Match	The received hashed Confirmation Code is different from the expected value.
8.2.7	Confirmation Code	6.4	Maximum number of attempts exceeded	The maximum number of incorrect attempts for the Confirmation Code has been exceeded.

Table 46c: ConfirmDeviceChange Specific status codes

5.7 ES10x (LPA -- eUICC)

ES10 contains 3 different interfaces described below.

The ES10a is an interface defined between the LDSd and ISD-R (LPA Services).

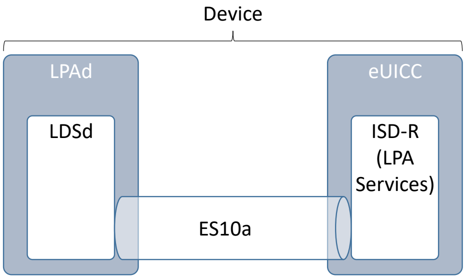


Figure 35: ES10a

The ES10b is an interface defined between the LPDd and ISD-R (LPA Services).

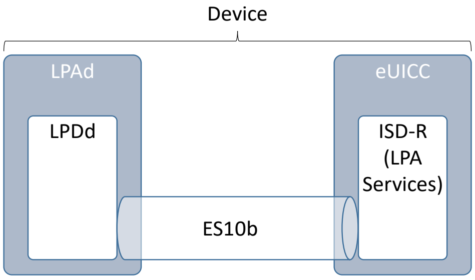


Figure 36: ES10b

The ES10c is an interface defined between the LUI and ISD-R (LPA Services).

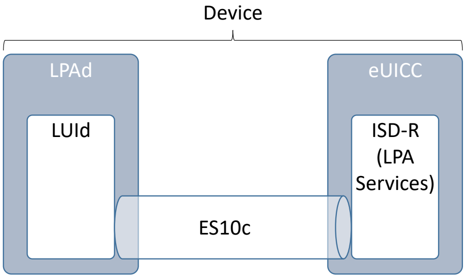


Figure 37: ES10c

Even if originally specified for Local Profile Management, some of the functions specified in this section are also used for RPM. If applicable, this is indicated in the "Related Procedures" entry.

For MEP, the functions specified in this section are executed on the Command Port. The rules for the different MEP modes on which eSIM Port can be used as Command Port and which eSIM Port can be the Target Port, as well as the mapping for SEP are defined in the procedures section.

5.7.1 ISD-R Selection and LPAe Activation

Before sending any command to the eUICC, the LPAd SHALL establish a logical channel and select the ISD-R.

The opening of the logical channel and the selection of the ISD-R SHALL be done explicitly using, respectively, the MANAGE CHANNEL command and the SELECT command defined in GlobalPlatform Card Specification [8]. This MANAGE CHANNEL and SELECT commands can be intrinsically used via a dedicated Device OS API (e.g., OMAPI defined by GlobalPlatform [69] if provided).

The Device SHALL ensure that only the LPAd, but no other application on the Device, is permitted to select the ISD-R, except that the ISD-R MAY also be selected during eUICC initialisation as defined in section 3.4.1.

In order to provide information about the capabilities supported by the eUICC at an early point in time, additional information is provided by the ISD-R.

On the reception of the SELECT ISD-R Command, the following data SHALL be returned within the FCI template after the objects defined in GlobalPlatform Card Specification [8]:

```
-- ASN1START
ISDRProprietaryApplicationTemplate ::= [PRIVATE 0] SEQUENCE { -- Tag 'E0'
   lowestSvn [2] VersionType,
   euiccConfiguration BIT STRING {
      lpaeUsingCatSupported(0), -- LPA in the eUICC using Card Application Toolkit
      lpaeUsingScwsSupported(1), -- LPA in the eUICC using Smartcard Web Server
      enabledProfile(2), -- eUICC contains an Enabled Profile
      lpaeUsingE4ESupported(3) -- LPA in the eUICC using 'E4' ENVELOPEs
   } OPTIONAL -- #MandatoryFromV3.0.0#
}
-- ASN1STOP
```

NOTE: eUICCs according to version 1.X of this specification will not return this data structure.

The `lowestSvn` field is deprecated and is only present for backward compatibility with the previous version of this specification. The LPAd SHALL determine the capabilities of the eUICC by using the ES10b.GetEUICCInfo function.

NOTE: The `lowestSvn` field was called `svn` in the previous versions of the specification.

If the Device supports the requirements for the LPAe using CAT or SCWS as defined in section 5.11 and the eUICC indicated support for that option in the ISDRProprietaryApplicationTemplate, the Device MAY activate this option by sending an LpaeActivationRequest to the ISD-R.

NOTE: The Device can deactivate LPAe by performing a reset of the eUICC.

If the Device indicates support for LUId, LPDd and LDSd and it does not send an LpaeActivationRequest, the eUICC SHALL NOT activate the LPAe.

If the Device indicates support for LPAe using E4E, the eUICC SHALL activate the LPAe using E4E.

NOTE: This requirement assumes that a device supporting LUIe using E4E will always be paired with an eUICC supporting LPAe using E4E. Otherwise, the first "E4" ENVELOPE command sent by the Device will be terminated with an error as defined in ETSI TS 102 223 [31].

In all other cases, the eUICC MAY activate the LPAe.

The LpaeActivationRequest SHALL be sent to the ISD-R using the transport mechanism defined in section 5.7.2.

The command data SHALL be coded as follows:

```
-- ASN1START
LpaeActivationRequest ::= [66] SEQUENCE { -- Tag 'BF42'
   lpaeOption BIT STRING {
      activateCatBasedLpae(0), -- LPAe with LUIe based on CAT
      activateScwsBasedLpae(1) -- LPAe with LUIe based on SCWS
   }
}
-- ASN1STOP
```

The Device SHALL set exactly one bit in `lpaeOption`.

The response data SHALL be coded as follows:

```
-- ASN1START
LpaeActivationResponse ::= [66] SEQUENCE { -- Tag 'BF42'
   lpaeActivationResult INTEGER {ok(0), notSupported(1)}
}
-- ASN1STOP
```

5.7.2 Transport Command

One generic APDU is used on the interfaces ES10a, ES10b and ES10c to transport all command request and command response data.

Command Message

All functions use the command message STORE DATA as defined in GlobalPlatform Card Specification [8] with the specific coding defined below.

Code	Value	Meaning
CLA	'80'-'83' or 'C0'-'CF'	See GlobalPlatform Card Specification [8] section 11.1.4
INS	'E2'	STORE DATA
P1	'11' or '91'	See below
P2	'xx'	Block number
Lc	Var.	Length of data field
Data	'xx xx…'	The data field SHALL be one of the data object command DER encoded defined in ES10x
Le	'00'	

Table 47: ES10x STORE DATA command APDU

Parameter P1

The P1 SHALL be coded as defined in the following table.

b8	b7	b6	b5	b4	b3	b2	b1	Meaning
0/1	-	-	-	-	-	-	-	More blocks/Last block
-	0	0	-	-	-	-	-	No general encryption information or non-encrypted data
-	-	-	1	0	-	-	-	BER-TLV format of the command data field
-	-	-	-	-	-	-	1	Case 4 command as defined in GlobalPlatform Amd A [9]
-	-	-	-	-	X	X	-	RFU

Table 48: ES10x STORE DATA P1

This interface is defined with command functions that are mostly handled with a single APDU command and response pair. When multiple STORE DATA commands are required, it is indicated by the use of the 'more commands' bit in the P1 byte as defined in GlobalPlatform Card Specification [8], and procedure bytes controlling the return of additional data (e.g., '61 XX'). In particular if the size of the response is bigger than 256 bytes, the chaining of the commands SHALL be done as defined in ISO/IEC 7816-4 [14]. The responses SHALL be retrieved by the Device using several GET RESPONSE commands.

Data Field

The command data field contains the command request data for each function.

Response Message

Data Field

The response data field contains the command response data for each function.

Processing State Returned in the Response Message

eUICC SHALL indicate an APDU header coding error as defined in GlobalPlatform Card Specification [8] section 11.11.3.2.

A successful execution of the APDU command SHALL be indicated by the status bytes '90 00' if no proactive command is pending and by '91 XX' if a proactive command (e.g., REFRESH) is pending. All function specific errors SHALL be indicated in the response data field.

An incorrect/invalid data field encoding (i.e., not a DER data object) SHALL be indicated by status bytes '6A 80' (Incorrect values in command data).

An unsupported or unknown command request in the data field SHALL be indicated by status bytes '6A 88' (Reference data not found).

While a Profile state change is ongoing, i.e., a command was sent to the eUICC which mandates a REFRESH, but the REFRESH was not yet successfully executed (i.e., no Terminal Response with result "command performed successfully" received or reset of the eUICC), the eUICC MAY reject any other ES10 command with the status word '69 85' (Conditions of use not satisfied).

5.7.3 Function (ES10a): GetEuiccConfiguredData

Related Procedures: SM-DS / Default SM-DP+ address Retrieval

Function Provider Entity: ISD-R (LPA Services)

NOTE: Prior to version 3, this function was called GetEuiccConfiguredAddresses.

Description:

This function retrieves the following, if configured:

- the Root SM-DS address(es)
- the Default SM-DP+ address and its allowed eSIM CA RootCA public key identifier from the eUICC
- the list of eSIM CA RootCA public key identifiers supported by the eUICC for signature verification together with a human-readable name of each eSIM CA, which MAY be used by the LUI when the End User selects an allowed eSIM CA for a new Default SM-DP+.

Command Data

The command data SHALL be coded as follows:

```
-- ASN1START
EuiccConfiguredDataRequest ::= [60] SEQUENCE {  -- Tag 'BF3C'
}
-- ASN1STOP
```

Response Data

The response data SHALL be coded as follows:

```
-- ASN1START
EuiccConfiguredDataResponse ::= [60] SEQUENCE {  -- Tag 'BF3C'
   defaultDpAddress UTF8String OPTIONAL,  -- Default SM-DP+ address
   rootDsAddress UTF8String,  -- Root SM-DS address
   additionalRootDsAddresses SEQUENCE OF UTF8String OPTIONAL, --
#SupportedFromV3.0.0#
   allowedCiPKId SubjectKeyIdentifier OPTIONAL, -- #SupportedFromV3.0.0# PKID
allowed for the Default SM-DP+
   ciList SEQUENCE OF SEQUENCE {    -- #SupportedFromV3.0.0#
      ciPKId SubjectKeyIdentifier,  -- List of eSIM CA RootCA public key identifiers
supported
      ciName UTF8String             -- on the eUICC together with a readable name
   } OPTIONAL
}
-- ASN1STOP
```

Server addresses are coded as FQDN.

For compatibility with previous versions of this specification, the first Root SM-DS address is provided in `rootDsAddress` and additional Root SM-DS addresses (if any) are provided in `additionalRootDsAddresses`. If no Root SM-DS address is configured, then `rootDsAddress` SHALL contain a zero-length string.

A removable eUICC SHALL have at least one Root SM-DS address configured.

5.7.4 Function (ES10a): SetDefaultDpAddress

Related Procedures: Set/Edit Default SM-DP+ Address

Function Provider Entity: ISD-R (LPA Services)

Description:

This function is used to update the Default SM-DP+ address and associated allowed eSIM CA RootCA public key identifier.

`allowedCiPKId` SHALL NOT be present if the version of the eUICC is lower than 3.

If `allowedCiPKId` is present then the eUICC SHALL verify that it supports this public key identifier for verification. If this verification fails then the eUICC SHALL return the error status `unsupportedCiPKId`.

If the provided UTF8 string (`defaultDpAddress`) is not empty, it SHALL constitute the new Default SM-DP+ address. If `allowedCiPKId` is present then it SHALL replace the allowed eSIM CA RootCA public key identifier list in the eUICC; otherwise, the allowed eSIM CA RootCA public key identifier in the eUICC SHALL be removed.

If the provided UTF8 string (`defaultDpAddress`) is empty, an existing Default SM-DP+ address and its associated allowed eSIM CA RootCA public key identifier SHALL be removed, irrespective of the value of `allowedCiPKId`.

<u>Command Data</u>

The command data SHALL be coded as follows:

```
-- ASN1START
SetDefaultDpAddressRequest ::= [63] SEQUENCE { -- Tag 'BF3F'
   defaultDpAddress UTF8String, -- Default SM-DP+ address as an FQDN
   allowedCiPKId SubjectKeyIdentifier OPTIONAL -- #SupportedFromV3.0.0# PKID
allowed for the Default SM-DP+
}
-- ASN1STOP
```

<u>Response Data</u>

The response data SHALL be coded as follows:

```
-- ASN1START
SetDefaultDpAddressResponse ::= [63] SEQUENCE { -- Tag 'BF3F'
   setDefaultDpAddressResult INTEGER {
   ok (0),
   unsupportedCiPKId(8), -- #SupportedFromV3.0.0#
   undefinedError (127)}
}
-- ASN1STOP
```

5.7.5 Function (ES10b): PrepareDownload

Related Procedures: Profile Download and Installation

Function Provider Entity: ISD-R (LPA Services)

Description:

This function initiates a Bound Profile Package download after a successful authentication of an SM-DP+.

On reception of this command, the eUICC SHALL:

- Verify that the SM-DP+ has been previously authenticated. Otherwise, the eUICC SHALL return an error code noSession.
- Verify the validity of the CERT.DPpb.SIG (using the ECASD service) as defined in section 4.5.2.2. Otherwise, the eUICC SHALL return an error code invalidCertificate.
- Verify that CERT.DPauth.SIG and CERT.DPpb.SIG belong to the same entity (i.e., same OID in subjectAltName). Otherwise, the eUICC SHALL return an error code invalidCertificate.
- Verify that CERT.DPauth.SIG of the on-going RSP session and CERT.DPpb.SIG are certified by the same certificate. Otherwise, the eUICC SHALL return an error code invalidCertificate.
- Upon any error returned in these verifications, the eUICC SHALL terminate the RSP session.
- Extract the public key of the CERT.DPpb.SIG and attach it to the RSP session.
- Verify the smdpSignature2 using the PK.DPpb.SIG. If the signature is invalid, the eUICC SHALL return an error code invalidSignature.
- Verify that the received transactionId contained in the smdpSigned2 matches the one of the on-going RSP session. Otherwise, the eUICC SHALL return an error code invalidTransactionId.
- If bppEuiccOtpk is provided in smdpSigned2 and it corresponds to a stored one-time KA key pair (otPK.EUICC.KA, otSK.EUICC.KA) for this SM-DP+: use this key pair for the RSP session. Otherwise: generate a new one-time KA key pair (otSK.EUICC.KA, otPK.EUICC.KA) using the parameters indicated by the subjectPublicKeyInfo.algorithmIdentifier.parameters field of the CERT.DPpb.SIG, and attach otSK.EUICC.KA to the RSP session.
- Generate euiccSigned2 data object as defined hereunder which MAY include vendor-specific additional information (e.g., as described in Annex P).
- Compute the euiccSignature2 over the concatenated data objects euiccSigned2 and smdpSignature2 using the SK.EUICC.SIG that was used in the "ES10b.AuthenticateServer response".

<u>Command Data</u>

The command data SHALL be coded as follows.

```
-- ASN1START
PrepareDownloadRequest ::= [33] SEQUENCE { -- Tag 'BF21'
   smdpSigned2 SmdpSigned2,                    -- Signed information
   smdpSignature2 [APPLICATION 55] OCTET STRING,      -- tag '5F37'
   hashCc Octet32 OPTIONAL, -- Hash of confirmation code
   smdpCertificate Certificate    -- CERT.DPpb.SIG
}

SmdpSigned2 ::= SEQUENCE {
   transactionId [0] TransactionId,      -- The TransactionID generated by the
SM-DP+
   ccRequiredFlag BOOLEAN, -- Indicates if the Confirmation Code is required
   bppEuiccOtpk [APPLICATION 73] OCTET STRING OPTIONAL,      -- otPK.EUICC.KA
already used for binding the BPP, tag '5F49'
   rpmPending NULL OPTIONAL -- #SupportedForRpmV3.0.0#
}
-- ASN1STOP
```

Response Data

The response data SHALL be coded as follows.

```
-- ASN1START
PrepareDownloadResponse ::= [33] CHOICE { -- Tag 'BF21'
   downloadResponseOk PrepareDownloadResponseOk,
   downloadResponseError PrepareDownloadResponseError
}

PrepareDownloadResponseOk ::= SEQUENCE {
   euiccSigned2 EUICCSigned2,           -- Signed information
   euiccSignature2 [APPLICATION 55] OCTET STRING      -- tag '5F37'
}

EUICCSigned2 ::= SEQUENCE {
   transactionId [0] TransactionId,
   euiccOtpk [APPLICATION 73] OCTET STRING,           -- otPK.EUICC.KA, tag '5F49'
   hashCc Octet32 OPTIONAL,             -- Hash of confirmation code
   additionalInformation VendorSpecificExtension OPTIONAL -- #SupportedFromV3.0.0#
}

PrepareDownloadResponseError ::= SEQUENCE {
   transactionId [0] TransactionId,
   downloadErrorCode DownloadErrorCode
}

DownloadErrorCode ::= INTEGER {invalidCertificate(1), invalidSignature(2),
noSession(4), invalidTransactionId(5), undefinedError(127)}
-- ASN1STOP
```

In case of the error `invalidTransactionId`, the `transactionId` in the `PrepareDownloadResponse` SHALL be set to the value from the `AuthenticateServerRequest`.

5.7.6 Function (ES10b): LoadBoundProfilePackage

Related Procedures: Profile Download and Installation

Function Provider Entity: ISD-R (LPA Services)

Description:

This function transfers a Bound Profile Package to the eUICC. The transfer is done by calling repeatedly this function with blocks of 255 bytes or lower according to the structure of the Bound Profile Package, i.e., each TLV of the BPP that is up to 255 bytes is transported in one APDU. Larger TLVs are sent in blocks of 255 bytes for the first blocks and a last block that MAY be shorter.

The eUICC SHALL erase the otSK.EUICC.KA attached to this RSP session no later than the successful completion of the BPP installation.

If this function is called when there is no RSP session, or if it is called with a TLV that is not expected according to the structure of the Bound Profile Package, the eUICC SHALL return the status words '6A 88' (Reference data not found) or '69 85' (Conditions of use not satisfied) as the response of the transport command defined in section 5.7.2. See section 3.1.5 for more details on error handling.

NOTE: The LPA may respond to these errors by calling ES10b.CancelSession.

Command Data

The command data SHALL contain a block of data of the BPP. The transfer and slicing in blocks of data SHALL follow description given in section 2.5.5.

Response Data

The data presence in the response message depends on the block status:

- For an intermediate block of data of a BPP TLV, the response message SHALL NOT contain data field.
- For the last block of data of a BPP TLV, a response message containing a Profile Installation Result SHALL be present or absent as specified in section 2.5.6.

After delivering the Profile Installation Result, the eUICC SHALL end the RSP session.

5.7.7 Function (ES10b): GetEUICCChallenge

Related Procedures: Common Mutual Authentication

Function Provider Entity: ISD-R (LPA Services)

Description:

This function initiates a RSP session between an RSP Server and the ISD-R. The initiation of the RSP session is materialized on the eUICC by the creation of a context containing an eUICC challenge.

Only one RSP session can be managed by the ISD-R at a time. So an on-going RSP session SHALL be completed before requesting the opening of a new one.

On reception of this function, the eUICC SHALL:

- Determine if a previous session was not completed. If so, then:

  - The eUICC MAY store the unused otPK.EUICC.KA and otSK.EUICC.KA, together with the SM-DP+ OID, for future retry.
  - An eUICC supporting the re-use of unused one-time key pairs SHALL limit the number of sessions in which one key pair can be used.
  - The eUICC SHALL discard the previous session context.

- Create a new session context and generate a new random challenge attached to this RSP session.

<u>Command Data</u>

The command data SHALL be coded as follows:

```
-- ASN1START
GetEuiccChallengeRequest ::= [46] SEQUENCE { -- Tag 'BF2E'
}
-- ASN1STOP
```

<u>Response Data</u>

The response data SHALL be coded as follows:

```
-- ASN1START
GetEuiccChallengeResponse ::= [46] SEQUENCE { -- Tag 'BF2E'
   euiccChallenge Octet16  -- random eUICC challenge
}
-- ASN1STOP
```

5.7.8 Function (ES10b): GetEUICCInfo

Related Procedures: Profile Download and Installation

Function Provider Entity: ISD-R (LPA Services)

Description:

This function gets the eUICC Information as defined in section 4.3. This function MAY be called at any time.

<u>Command Data</u>

The command data SHALL be coded as follows to retrieve EUICCInfo1:

```
-- ASN1START
GetEuiccInfo1Request ::= [32] SEQUENCE { -- Tag 'BF20'
}
-- ASN1STOP
```

The command data SHALL be coded as follows to retrieve EUICCInfo2:

```
-- ASN1START
GetEuiccInfo2Request ::= [34] SEQUENCE { -- Tag 'BF22'
}
-- ASN1STOP
```

Response Data

EUICCInfo1 or EUICCInfo2 SHALL be returned for GetEUICCInfo1Request or GetEUICCInfo2Request, respectively, as specified in section 4.3.

5.7.9 Function: (ES10b): ListNotification

Related Procedures: Notifications

Function Provider Entity: ISD-R

Description:

This function is used by the LPA to list all available pending notifications from an eUICC before retrieving a specific notification.

Command Data

The command data SHALL be coded as follows:

```
-- ASN1START
ListNotificationRequest ::= [40] SEQUENCE { -- Tag 'BF28'
   profileManagementOperation [1] NotificationEvent OPTIONAL
}
-- ASN1STOP
```

The profileManagementOperation data object can be used to filter the list of notifications that the eUICC SHALL return. A bit set to 1 in the profileManagementOperation indicates that the eUICC SHALL return all the notifications corresponding to this type. The type notificationInstall SHALL include ProfileInstallationResult.

If profileManagementOperation data object is omitted, the eUICC SHALL return all stored notifications whatever their type.

If profileManagementOperation data object indicates no event (all bits set to 0), the eUICC SHALL return an empty list or undefinedError(127) .

Response Data

The response data SHALL contain the 'List Notification Response' data object if available, and filtered according to profileManagementOperation data object received in the command data. The eUICC MAY provide the notifications in any order. The list SHALL be empty if there are no pending notification matching the filtering criteria.

```
-- ASN1START
ListNotificationResponse ::= [40] CHOICE { -- Tag 'BF28'
   notificationMetadataList SEQUENCE OF NotificationMetadata,
   listNotificationsResultError INTEGER {undefinedError(127)}
}

NotificationMetadata ::= [47] SEQUENCE { -- Tag 'BF2F'
   seqNumber [0] INTEGER,
   profileManagementOperation [1] NotificationEvent, -- Only one bit SHALL be set
to 1
   notificationAddress UTF8String, -- FQDN to forward the notification
   iccid Iccid OPTIONAL
}
-- ASN1STOP
```

5.7.10 Function (ES10b): RetrieveNotificationsList

Related Procedures: Notifications

Function Provider Entity: ISD-R (LPA Services)

Description:

This function retrieves the list of Pending notifications for installed Profiles including their confirmation required and the related data.

Command Data

The command data SHALL be coded as follows:

```
-- ASN1START
RetrieveNotificationsListRequest ::= [43] SEQUENCE { -- Tag 'BF2B'
   searchCriteria CHOICE {
      seqNumber [0] INTEGER,
      profileManagementOperation [1] NotificationEvent
   } OPTIONAL
}
-- ASN1STOP
```

The `searchCriteria` data object can be used to filter the list of notifications that the eUICC SHALL return, filtering can be done on sequence number or notification type. A bit set to 1 in the `profileManagementOperation` indicates that the eUICC SHALL return all the notifications corresponding to this type. The type `notificationInstall` SHALL include `ProfileInstallationResult`.

If `searchCriteria` data object is omitted, the eUICC SHALL return all stored Notifications.

Response Data

The response data SHALL contain the list of PendingNotification data objects. The list SHALL be filtered according to the notification seqNumber or indicated operation type that generates notifications provided in the command data. The eUICC MAY provide the notifications in any order. The list SHALL be empty if there are no pending notifications matching the filtering criteria.

The following is the definition of the RetrieveNotificationsListResponse data object

```
-- ASN1START
RetrieveNotificationsListResponse ::= [43] CHOICE { -- Tag 'BF2B'
   notificationList SEQUENCE OF PendingNotification,
   notificationsListResultError INTEGER { undefinedError(127)}
}

PendingNotification ::= CHOICE {
   profileInstallationResult [55] ProfileInstallationResult, -- tag 'BF37'
   otherSignedNotification OtherSignedNotification,
   loadRpmPackageResultSigned [1] LoadRpmPackageResultSigned
}
```

```
OtherSignedNotification ::= SEQUENCE {
   tbsOtherNotification NotificationMetadata,
   euiccNotificationSignature EuiccSign,
   euiccCertificate Certificate, -- eUICC Certificate (CERT.EUICC.SIG)
   nextCertInChain Certificate, -- The certificate certifying the eUICC Certificate
   otherCertsInChain [1] CertificateChain OPTIONAL -- #SupportedFromV3.0.0# Other
Certificates in the eUICC certificate chain, if any
}
-- ASN1STOP
```

`euiccNotificationSignature` SHALL be created using the SK.EUICC.SIG and verified using the PK.EUICC.SIG as described in section 2.6.9. `euiccNotificationSignature` SHALL apply on the `tbsOtherNotification` data object.

When generating the `euiccNotificationSignature`, the eUICC SHALL use credentials related to the `euiccCiPKIdToBeUsed` parameter received from the SM-DP+ during the Profile Download and Installation Procedure.

The `nextCertInChain` data object SHALL contain the Certificate certifying the CERT.EUICC.SIG.

The `otherCertsInChain` data object, if present, SHALL contain the remaining part of the certificate chain certifying the `nextCertInChain` data object. See section 4.5.2.0a.

NOTE: The `nextCertInChain` data object was called `eumCertificate` in previous versions of this specification.

5.7.11 Function (ES10b): RemoveNotificationFromList

Related Procedures: Notifications

Function Provider Entity: ISD-R (LPA Services)

Description:

This function informs the eUICC that a specific Notification has been sent to the recipient address.

On reception of this command, the eUICC SHALL:

- Verify that the Notification identified by its sequence number exists. Otherwise, the eUICC SHALL return an error code `nothingToDelete`.
- Remove such Notification from the Pending Notifications List.
- Return `deleteNotificationStatus` with value `ok`.

Command Data

The command data SHALL be coded as follows:

```
-- ASN1START
NotificationSentRequest ::= [48] SEQUENCE { -- Tag 'BF30'
   seqNumber [0] INTEGER
}
```

```
-- ASN1STOP
```

Response Data

The response data SHALL be coded as follows:

```
-- ASN1START
NotificationSentResponse ::= [48] SEQUENCE { -- Tag 'BF30'
   deleteNotificationStatus INTEGER {ok(0), nothingToDelete(1),
undefinedError(127)}
}
-- ASN1STOP
```

5.7.12 Function (ES10b): LoadCRL

Related Procedures: None

Function Provider Entity: ISD-R (LPA Services)

Description:

This function, that was defined prior to version 3, is no longer supported by the eUICC.

5.7.13 Function (ES10b): AuthenticateServer

Related Procedures: Common Mutual Authentication

Function Provider Entity: ISD-R (LPA Services)

Description:

This function performs the authentication of the RSP Server by the eUICC.

On reception of this command, the eUICC SHALL:

- Verify that a RSP session exists (i.e., "ES10b.GetUICCChallenge" function has been previously called). Otherwise, the eUICC SHALL return a noSession error code.
- Verify the validity of the RSP Server Certificate chain for authentication (using the ECASD service), as described in section 4.5.2.2 using the PK.CI.SIG identified by the Authority Key Identifier contained in the last Certificate of the chain. If the PK.CI.SIG is unknown, the eUICC SHALL return a ciPKUnknown error code. If the RSP Server Certificate or the Certificate chain is invalid the eUICC SHALL return an invalidCertificate error code.
- Verify that the RSP Server Certificate is either a CERT.DPauth.SIG or a CERT.DSauth.SIG, as described in section 4.5.2.2. Otherwise, the eUICC SHALL return an invalidOid error code.
- Verify the signature (serverSignature1) of the RSP Server created over serverSigned1 as described hereunder. If the signature is invalid the eUICC SHALL return an invalidSignature error code.
- Verify that the euiccChallenge attached to the ongoing RSP session matches the serverSigned1.euiccChallenge returned by the RSP Server. Otherwise, the eUICC SHALL return an euiccChallengeMismatch error code.

- Verify that the eSIM CA RootCA Public Key Identifier indicated in either `euiccCiPKIdToBeUsed` or `euiccCiPKIdToBeUsedV3` is supported and related credentials are available for signing. Otherwise, the eUICC SHALL return an `unsupportedCurve` error code. If both values are empty or omitted, the eUICC SHALL also return an `unsupportedCurve` error code.
- If RSP Server indicates `crlStaplingV3Used`:
    - Verify that the `crlList` data object is present. If not, the eUICC SHALL return a `missingCrl` error code.
    - Verify the signature of each CRL in the list, as defined in RFC 5280 [17], using the Public Key identified in its authorityKeyIdentifier extension (this Public Key may be contained in one of the received Certificate(s) or may be an eSIM CA RootCA Public Key known by the eUICC). If a CRL signature is invalid, the eUICC SHALL return an `invalidCrlSignature` error code.
    - Update its time reference as described in section 4.6.3.1 using the received CRL from the eSIM CA RootCA, and the CRL from the eSIM CA SubCA, if any.
    - Verify that no Certificate in the chain is revoked as follows:
        - Verify that the time information contained in all the Certificates and the CRLs is acceptable as described in section 4.6.3.2. Otherwise, the eUICC SHALL return an `invalidCertOrCrlTime` error code.
        - For each Certificate in the chain that contains a `cRLDistributionPoints` extension:
            - Search for the CRL in the `crlList` input data that has the same issuer, same authority key identifier in its `authorityKeyIdentifier` extension, and contains a matching distribution point name in its `issuingDistributionPoint` (IDP) extension (i.e., search for the CRL that has the Certificate in its scope). If not present, the eUICC SHALL return a `missingCrl` error code.
            - If `onlyContainsUserCerts` is set in the IDP CRL extension, verify that the Certificate does not include the `basicConstraints` extension with `cA` set. If not, the eUICC SHALL return an `invalidCertOrCrlConfiguration` error code.
            - If `onlyContainsCACerts` is set in the IDP CRL extension, verify that the Certificate includes the basic constraints extension with `cA` set. If not, the eUICC SHALL return an `invalidCertOrCrlConfiguration` error code.
            - Verify that `onlyContainsAttributeCerts` is not set. If not, the eUICC SHALL return an `invalidCertOrCrlConfiguration` error code.
            - Verify that `deltaCRLIndicator` and `freshestCRL` extensions are not set in the CRL. If at least one is set, the eUICC SHALL return an `invalidCertOrCrlConfiguration` error code.
            - Verify that the `keyUsage` extension in the CRL issuer’s Certificate contains the `cRLSign` indicator. If not, the eUICC SHALL return an `invalidCertOrCrlConfiguration` error code.
            - Verify that the Certificate serial number is not present in the CRL. If it is present, the eUICC SHALL return a `revokedCert` error code.

NOTE: As the number of SubCAs is unpredictable, the eUICC is not expected to track CRL numbers.

- If `ctxParams1` includes `ctxParamsForDeviceChange`, verify that the Profile identified by the `iccid` is installed in the eUICC. If not, the eUICC SHALL return an `invalidIccid` error code.
- Upon any error returned in these verifications, the eUICC SHALL terminate the RSP session.
- Attach the received `transactionId` to the RSP session.
- Attach the received RSP Server Certificate to the RSP session.
- Generate `euiccSigned1` data object as defined hereunder.
- Generate the `euiccSignature1` as defined hereunder, with the SK.EUICC.SIG related to the CERT.EUICC.SIG as requested by the RSP Server.

Command Data

The command data SHALL be coded as follows.

```
-- ASN1START
AuthenticateServerRequest ::= [56] SEQUENCE { -- Tag 'BF38'
   serverSigned1 ServerSigned1,                       -- Signed information
   serverSignature1 [APPLICATION 55] OCTET STRING,    -- tag '5F37'
   euiccCiPKIdToBeUsed SubjectKeyIdentifier OPTIONAL, -- eSIM CA RootCA Public Key
Identifier to be used; MAY also have zero length
   serverCertificate Certificate, -- RSP Server Certificate CERT.XXauth.SIG
   ctxParams1 CtxParams1,
   otherCertsInChain [1] CertificateChain OPTIONAL, -- #SupportedFromV3.0.0# The
remaining part of the CERT.XXauth.SIG certificate chain (if any)
   crlList [2] SEQUENCE OF CertificateList OPTIONAL -- #SupportedFromV3.0.0# as
specified in RFC 5280
}

ServerSigned1 ::= SEQUENCE {
   transactionId [0] TransactionId,            -- The Transaction ID generated by
the RSP Server
   euiccChallenge [1] Octet16,          -- The eUICC Challenge
   serverAddress [3] UTF8String,  -- The RSP Server address as an FQDN
   serverChallenge [4] Octet16,         -- The RSP Server Challenge
   sessionContext [5] SessionContext OPTIONAL, -- #SupportedFromV3.0.0#
   serverRspCapability [6] ServerRspCapability OPTIONAL -- #SupportedFromV3.0.0#
}

CtxParams1 ::= CHOICE {
   ctxParamsForCommonAuthentication[0] CtxParamsForCommonAuthentication,
   ctxParamsForDeviceChange [1] CtxParamsForDeviceChange,
   ctxParamsForProfileRecovery [2] CtxParamsForProfileRecovery,
   ctxParamsForPushServiceRegistration [3] CtxParamsForPushServiceRegistration
-- New contextual data objects MAY be defined for extensibility.
}

CtxParamsForCommonAuthentication ::= SEQUENCE {
   matchingId [0] UTF8String OPTIONAL, -- The MatchingId could be the Activation
code token or EventID or empty
   deviceInfo [1] DeviceInfo, -- The Device information
   operationType [2] OperationType DEFAULT {profileDownload}, --
#SupportedFromV3.0.0#
   iccid Iccid OPTIONAL, -- ICCID, tag '5A' #SupportedFromV3.0.0#
   matchingIdSource [3] MatchingIdSource OPTIONAL, -- #SupportedFromV3.0.0#
   vendorSpecificExtension [4] VendorSpecificExtension OPTIONAL --
#SupportedFromV3.0.0#
}
```

```
CtxParamsForDeviceChange ::= SEQUENCE { -- #SupportedForDcV3.0.0#
   iccid Iccid,
   deviceInfo [1] DeviceInfo,
   targetEidValue [APPLICATION 26] Octet16 OPTIONAL,
   targetTacValue [2] Octet4 OPTIONAL,
   vendorSpecificExtension [3] VendorSpecificExtension OPTIONAL
}

CtxParamsForProfileRecovery ::= SEQUENCE { -- #SupportedForDcV3.0.0#
   iccid Iccid,
   deviceInfo [1] DeviceInfo,
   vendorSpecificExtension [2] VendorSpecificExtension OPTIONAL
}

CtxParamsForPushServiceRegistration ::= SEQUENCE { --
#SupportedForPushServiceV3.0.0#
   selectedPushService [0] OBJECT IDENTIFIER,
   pushToken [1] UTF8String
}

MatchingIdSource ::= CHOICE {
   none [0] NULL,
   activationCode [1] NULL,
   smdsOid [2] OBJECT IDENTIFIER
}

OperationType ::= BIT STRING {
   profileDownload(0),
   rpm(1)
}

-- Records information agreed along the session
SessionContext ::= SEQUENCE {
   serverSvn [0] VersionType, -- RSP Server SVN (provided for information only)
   crlStaplingV3Used [1] BOOLEAN, -- Indicates CRLs were attached to the RSP Server
response
   euiccCiPKIdToBeUsedV3 [2] SubjectKeyIdentifier OPTIONAL,
   supportedPushServices [3] SEQUENCE OF OBJECT IDENTIFIER OPTIONAL
}

-- Definition of ServerRspCapability
ServerRspCapability ::= BIT STRING {
   crlStaplingV3Support (0), -- support for CRL stapling
   eventListSigningV3Support (1), -- support for Event Record signing
   pushServiceV3Support (2), -- support for Push Service
   cancelForEmptySpnPnSupport (3)
}
-- ASN1STOP
```

`serverSignature1` SHALL be created using the private key associated to the RSP Server Certificate for authentication, and verified by the eUICC using the contained public key as described in section 2.6.9. `serverSignature1` SHALL apply on `serverSigned1` data object.

`serverSvn` indicates the highest Specification Version Number of this specification supported by the RSP Server and is provided for information only (see Annex M).

Either `euiccCiPKIdToBeUsed` or `euiccCiPKIdToBeUsedV3` SHALL contain an eSIM CA RootCA Public Key Identifier. `euiccCiPKIdToBeUsed` indicates the Variant O certificate chain related to the Private Key that the eUICC SHALL use for signing. `euiccCiPKIdToBeUsedV3` indicates one of the other certificate chain Variants related to the Private Key that the eUICC SHALL use for signing.

If the CERT.DPauth.SIG or CERT.DSauth.SIG is in a certificate chain Variant O, `otherCertsInChain` SHALL be omitted. Otherwise it SHALL be present and contain the remaining part of the CERT.DPauth.SIG or CERT.DSauth.SIG certificate chain (see table 36 in section 5.6.1 and section 5.8.1).

If `matchingIdSource` is not present, the source is unknown.

A `vendorSpecificExtension` data object MAY provide additional implementation-specific information.

A `supportedPushServices` data object MAY be present in case of the SM-DS supporting at least one Push Service and indicates the list of supported Push Service by the SM-DS.

Description of `ServerRspCapability`:

Refer to Annex M that describes how a version 3 RSP Server SHALL be configured.

- The `crlStaplingV3Support` bit SHALL be set to '1' if and only if the RSP Server supports the CRL stapling during the Common Mutual Authentication procedure.
- The `eventListSigningV3Support` bit SHALL be set to '1' if and only if the SM-DS supports signing the list of Event Records in the response of ES11.AuthenticateClient.
- The `pushServiceV3Support` bit SHALL be set to '1' if and only if the SM-DS supports at least one Push Service.
- The `cancelForEmptySpnPnSupport` bit SHALL be set to '1' if and only if the SM-DP+ supports the cancel session reason codes defined for eSIM V3 in the command data of ES10b.CancelSession.

The eUICC SHALL neither check the values provided in `ctxParams1` other than ICCID in `ctxParamsForDeviceChange` nor correlate them with subsequent session activities (e.g., whether `OperationType.rpm` was set for an RPM session).

Response Data

The response data SHALL be coded as follows.

```
-- ASN1START
AuthenticateServerResponse ::= [56] CHOICE { -- Tag 'BF38'
   authenticateResponseOk [0] AuthenticateResponseOk,
   authenticateResponseError [1] AuthenticateResponseError
}

AuthenticateResponseOk ::= SEQUENCE {
   euiccSigned1 EuiccSigned1,            -- Signed information
   euiccSignature1 [APPLICATION 55] OCTET STRING,     --EUICC_Sign1, tag 5F37
   euiccCertificate Certificate,  -- eUICC Certificate (CERT.EUICC.SIG)
   nextCertInChain Certificate,   -- The Certificate certifying the eUICC
Certificate
   otherCertsInChain [0] CertificateChain OPTIONAL -- #SupportedFromV3.0.0# Other
Certificates in the eUICC certificate chain, if any
}

EuiccSigned1 ::= SEQUENCE {
   transactionId [0] TransactionId,
   serverAddress [3] UTF8String, -- The RSP Server address as an FQDN
   serverChallenge [4] Octet16,   -- The RSP Server Challenge
   euiccInfo2 [34] EUICCInfo2,
   ctxParams1 CtxParams1
}

AuthenticateResponseError ::= SEQUENCE {
   transactionId [0] TransactionId,
   authenticateErrorCode AuthenticateErrorCode
}

AuthenticateErrorCode ::= INTEGER {invalidCertificate(1), invalidSignature(2),
unsupportedCurve(3), noSession(4), invalidOid(5), euiccChallengeMismatch(6),
ciPKUnknown(7),
transactionIdError (8), -- #SupportedFromV3.0.0#
missingCrl(9), -- #SupportedFromV3.0.0#
invalidCrlSignature(10), -- #SupportedFromV3.0.0#
revokedCert(11), -- #SupportedFromV3.0.0#
invalidCertOrCrlTime(12), -- #SupportedFromV3.0.0#
invalidCertOrCrlConfiguration(13), -- #SupportedFromV3.0.0#
invalidIccid(14), -- #SupportedForDcV3.0.0#
undefinedError(127)}
-- ASN1STOP
```

euiccSignature1 SHALL be created using the SK.EUICC.SIG and verified using the PK.EUICC.SIG as described in in section 2.6.9. euiccSignature1 SHALL apply on euiccSigned1 data object.

The nextCertInChain data object SHALL contain the Certificate certifying the CERT.EUICC.SIG.

The otherCertsInChain data object, if present, SHALL contain the remaining part of the certificate chain certifying the nextCertInChain data object. See section 4.5.2.0a.

NOTE: The nextCertInChain data object was called eumCertificate in previous versions of this specification.

In case of a missing or damaged Transaction ID in the command data, transactionIdError SHALL be reported in the AuthenticateServerResponse with the transactionId value being empty.

5.7.14 Function (ES10b): CancelSession

Related Procedures: Profile Download and Installation, Remote Profile Management, Remote eUICC Management

Function Provider Entity: ISD-R (LPA Services)

Description:

This function allows the LPAd to cancel an on-going RSP session on the eUICC and to get a signed data structure that is necessary for the LPAd to request the same session cancellation to the RSP Server.

On reception of this command the eUICC SHALL:

- Verify that the Transaction ID provided as input data is known and matches the one of the ongoing RSP session. Otherwise the eUICC SHALL return an error code `invalidTransactionId`.
- Generate an `euiccCancelSessionSigned` data object as defined hereunder.
- Generate the `euiccCancelSessionSignature`, as defined hereunder, with the SK.EUICC.SIG.
- If the reason is `postponed` or `timeout`, the eUICC MAY store the unused otPK.EUICC.KA and otSK.EUICC.KA, together with the SM-DP+ identity, for future retry.
- Discard the on-going RSP session.
- Respond with `cancelSessionResponseOk` defined hereunder.

The eUICC MAY perform additional internal operations, which are out of scope of this specification.

<u>Command Data</u>

The command data SHALL be coded as follows.

```
-- ASN1START
CancelSessionRequest ::= [65] SEQUENCE { -- Tag 'BF41'
   transactionId TransactionId,    -- The TransactionID generated by the RSP Server
   reason CancelSessionReason
}

CancelSessionReason ::= INTEGER {
   endUserRejection(0),
   postponed(1),
   timeout(2),
   pprNotAllowed(3),
   metadataMismatch(4),
   loadBppExecutionError(5),
   sessionAborted(16), -- #SupportedFromV3.0.0#
   enterpriseProfilesNotSupported(17), -- #SupportedFromV3.0.0#
   enterpriseRulesNotAllowed(18), -- #SupportedForEnterpriseV3.0.0#
   enterpriseProfileNotAllowed(19), -- #SupportedForEnterpriseV3.0.0#
   enterpriseOidMismatch(20), -- #SupportedForEnterpriseV3.0.0#
   enterpriseRulesError(21), -- #SupportedForEnterpriseV3.0.0#
   enterpriseProfilesOnly(22), -- #SupportedForEnterpriseV3.0.0#
   lprNotSupported(23), -- #SupportedForLpaProxyV3.0.0#
   lprNetworkDataNotAllowed(24), -- #SupportedForLpaProxyV3.0.0#
   emptyProfileOrSpName(25), -- #SupportedFromV3.0.0#
   rpmDisabled(27), -- #SupportedForRpmV3.0.0#
   invalidRpmPackage(28), -- #SupportedFromV3.0.0#
   loadRpmPackageError(29), -- #SupportedForRpmV3.0.0#
   undefinedReason(127)
}
-- ASN1STOP
```

NOTE: Cancel session reason values added since v3.0.0 are aligned with the error reason values in section 2.5.6.

<u>Response Data</u>

The response data SHALL be coded as follows.

```
-- ASN1START
CancelSessionResponse ::= [65] CHOICE { -- Tag 'BF41'
   cancelSessionResponseOk CancelSessionResponseOk,
   cancelSessionResponseError INTEGER {invalidTransactionId(5),
undefinedError(127)}
}

CancelSessionResponseOk ::= SEQUENCE {
   euiccCancelSessionSigned EuiccCancelSessionSigned,        -- Signed information
   euiccCancelSessionSignature [APPLICATION 55] OCTET STRING -- tag '5F37'
}

EuiccCancelSessionSigned ::= SEQUENCE {
   transactionId TransactionId,
   smdpOid OBJECT IDENTIFIER, -- SM-DP+ OID as contained in CERT.DPauth.SIG
   reason CancelSessionReason
}
-- ASN1STOP
```

`euiccCancelSessionSignature` SHALL be created using the SK.EUICC.SIG and verified using the PK.EUICC.SIG as described in section 2.6.9. `euiccCancelSessionSignature` SHALL apply on the data object `euiccCancelSessionSigned`.

5.7.15 Function (ES10c): GetProfilesInfo

Related Procedures: Local and Remote Profile Management – List Profiles

Function Provider Entity: ISD-R (LPA Services)

Description:

This function retrieves the list of Profile information for installed Profiles including their current state (Enabled/Disabled) and their associated Profile Metadata. This function MAY also be used to retrieve this information for a particular Profile.

If there is a Profile state change ongoing (i.e., during an enable, disable or eUICCMemoryReset command requiring a REFRESH) and the Profile state is requested, the eUICC SHALL terminate the command with the status word '69 85' (Conditions of use not satisfied) for Local Profile Management and with error `profileChangeOngoing` for Remote Profile Management (if the LoadRpmPackage command which contained this command was not rejected by the eUICC, see section 5.7.2).

Command Data

The command data consists of the search criteria and the tag list. For Local Profile Management, it SHALL be coded as follows:

```
-- ASN1START
ProfileInfoListRequest ::= [45] SEQUENCE { -- Tag 'BF2D'
   searchCriteria [0] CHOICE {
      isdpAid [APPLICATION 15] OctetTo16, -- AID of the ISD-P, tag '4F'
      iccid Iccid, -- ICCID, tag '5A'
      profileClass [21] ProfileClass -- Tag '95'
   } OPTIONAL,
   tagList [APPLICATION 28] OCTET STRING OPTIONAL -- tag '5C'
}
-- ASN1STOP
```

For Remote Profile Management, it SHALL be coded as follows:

```
-- ASN1START
ListProfileInfo ::= [5] SEQUENCE {
   searchCriteria [0] CHOICE {
      iccid Iccid,
      profileOwnerOid [0] OBJECT IDENTIFIER
   },
   tagList [APPLICATION 28] OCTET STRING OPTIONAL
}
-- ASN1STOP
```

The following SHALL apply for Local Profile Management:

- It SHALL be possible to search for all the Profiles using no search criterion.

The following SHALL apply for Remote Profile Management:

- If an ICCID is specified in the command, the eUICC SHALL identify the target Profile, and SHALL verify that:
  - the SM-DP+ that sent the RPM command is included in the Managing SM-DP+ List of the target Profile. Otherwise, an empty `ProfileInfoListOk` SHALL be returned.
  - the allowed CI public key identifier (if present) matches the Subject Key Identifier of the CI corresponding to the CERT.DPauth.SIG of the SM-DP+ that sent the RPM Command. Otherwise, an empty `ProfileInfoListOk` SHALL be returned.
- If a Profile Owner OID is specified in the command, the eUICC SHALL identify all Profiles where the RPM configuration fulfils all of the following conditions:
  - Profile Owner matches the Profile Owner OID in the RPM Command, and
  - Managing SM-DP+ List includes the SM-DP+ that sent the RPM Command, and
  - allowed CI public key identifier (if present) matches the Subject Key Identifier of the CI corresponding to the CERT.DPauth.SIG of the SM-DP+ that sent the RPM Command.

For each identified Profile, the eUICC SHALL return a `ProfileInfo` data object in `profileInfoListOk`. `profileInfoListOk` SHALL be empty if the list of identified Profiles is empty.

The content of each `ProfileInfo` data object is defined by the tag list as follows:

The value field of the tag list (tag '5C') contains a concatenation of tags (without delimitation) indicating the data objects to include in the response for each Profile matching all given search criteria.

If a requested data object is not present for a matching Profile, the data object SHALL simply be omitted in the response for that Profile.

The eUICC SHALL support the following tags in the tag list:

- ICCID, tag '5A' (*)
- ISD-P AID, tag '4F' (*)
- Profile state, tag '9F70' (*)

- Profile Nickname, tag '90' (*)
- Service provider name, tag '91' (*)
- Profile name, tag '92' (*)
- Icon type, tag '93' (*)
- Icon, tag '94' (*)
- Profile Class, tag '95' (*)
- Notification Configuration Info, tag 'B6'
- Profile Owner, tag 'B7'
- SM-DP+ proprietary data, tag 'B8'
- Profile Policy Rules, tag '99'
- Service Specific Data stored in eUICC, tag 'BF22'
- RPM Configuration, tag 'BA'
- HRI Server address, tag '9B'
- LPR Configuration, tag 'BC'
- Enterprise Configuration, tag 'BD'
- Service Description, tag '9F1F'
- Device Change configuration, tag 'BF20'
- Enabled on eSIM Port, tag '9F24'
- Profile Size, tag '9F25'

If no tag list is present, the eUICC SHALL return the default ProfileInfo: the ProfileInfo data objects marked with (*) for each Profile matching the selection criterion.

NOTE: For RPM, the SM-DP+ SHOULD take the maximum result size supported by the eUICC into account when constructing the command.

Example of use for Local Profile Management:

- Retrieve the default ProfileInfo for all installed Profiles. The command data field SHALL be coded as 'BF2D 00'.
- Retrieve ICCID, Profile State, Profile name, Profile Class and Profile Owner of a particular Profile/ISD-P having the following AID: A0 00 00 05 59 10 10 FF FF FF FF 89 00 00 10 00. The command data field SHALL be coded as 'BF2D 1C A0 12 4F 10 A0 00 00 05 59 10 10 FF FF FF FF 89 00 00 10 00 5C 06 5A 9F70 92 95 B7'.
- Retrieve ICCID and Profile state for all installed Profiles. The command data field SHALL be coded as 'BF 2D 05 5C 03 5A 9F70'.

Response Data

The following is the definition of the `ProfileInfoListResponse` data object:

```
-- ASN1START
-- Definition of ProfileInfoListResponse
ProfileInfoListResponse ::= [45] CHOICE { -- Tag 'BF2D'
   profileInfoListOk SEQUENCE OF ProfileInfo,
   profileInfoListError ProfileInfoListError
}

ProfileInfo ::= [PRIVATE 3] SEQUENCE { -- Tag 'E3'
   iccid Iccid OPTIONAL,
   isdpAid [APPLICATION 15] OctetTo16 OPTIONAL, -- AID of the ISD-P containing the
Profile, tag '4F'
```

```
   profileState [112] ProfileState OPTIONAL, -- Tag '9F70'
   profileNickname [16] UTF8String (SIZE(0..64)) OPTIONAL, -- Tag '90'
   serviceProviderName [17] UTF8String (SIZE(0..32)) OPTIONAL, -- Tag '91'
   profileName [18] UTF8String (SIZE(0..64)) OPTIONAL, -- Tag '92'
   iconType [19] IconType OPTIONAL, -- Tag '93'
   icon [20] OCTET STRING (SIZE(0..1024)) OPTIONAL, -- Tag '94',
   profileClass [21] ProfileClass OPTIONAL, -- Tag '95'
   notificationConfigurationInfo [22] SEQUENCE OF
NotificationConfigurationInformation OPTIONAL, -- Tag 'B6'
   profileOwner [23] OperatorId OPTIONAL, -- Tag 'B7'
   dpProprietaryData [24] DpProprietaryData OPTIONAL, -- Tag 'B8'
   profilePolicyRules [25] PprIds OPTIONAL, -- Tag '99'
   serviceSpecificDataStoredInEuicc [34] VendorSpecificExtension OPTIONAL, --
#SupportedFromV2.4.0# Tag 'BF22'
   rpmConfiguration [26] RpmConfiguration OPTIONAL, -- #SupportedForRpmV3.0.0# Tag
'BA'
   hriServerAddress [27] UTF8String OPTIONAL, -- #SupportedFromV3.0.0# Tag '9B'
   lprConfiguration [28] LprConfiguration OPTIONAL, -- #SupportedForLpaProxyV3.0.0#
Tag 'BC'
   enterpriseConfiguration [29] EnterpriseConfiguration OPTIONAL,
-- #SupportedForEnterpriseV3.0.0# Tag 'BD'
   serviceDescription [31] ServiceDescription OPTIONAL, -- #SupportedFromV3.0.0#
Tag '9F1F'
   deviceChangeConfiguration [32] DeviceChangeConfiguration OPTIONAL, --
#SupportedForDcV3.0.0# Tag 'BF20'
   enabledOnEsimPort [36] INTEGER OPTIONAL, -- #SupportedForMEPV3.0.0# Tag '9F24'
   profileSize [37] INTEGER OPTIONAL -- #SupportedFromV3.0.0# Tag '9F25'
}

IconType ::= INTEGER {jpg(0), png(1)}
ProfileState ::= INTEGER {disabled(0), enabled(1)}
ProfileClass ::= INTEGER {test(0), provisioning(1), operational(2)}
ProfileInfoListError ::= INTEGER {
   incorrectInputValues(1),
   profileChangeOngoing (11), -- #SupportedForRpmV3.0.0#
   undefinedError(127)
}
-- ASN1STOP
```

For SEP, RPM, MEP-A1 and MEP-A2, the `profileState` data object SHALL indicate the current state of the Profile. For LPM with MEP-B, the value of the data object is EUM specific.

The `enabledOnEsimPort` data object is applicable for LPM with MEP only and SHALL indicate on which eSIM Port a Profile is in Enabled state. It SHALL NOT be provided by the eUICC for a Profile in Disabled state, for SEP and RPM.

The `profileOwner` data object can only be returned if Profile Owner has been provided in Profile Metadata or if files EF_{IMSI}, EF_{GID1} or EF_{GID2} are not PIN protected.

The `profilePolicyRules` data object SHALL contain the identifiers of all Profile Policy Rules of the Profile.

The `profileSize` data object contains the estimated size of the installed Profile in the non-volatile memory, expressed in bytes. The indicated size MAY be a rounded value. The way the eUICC estimates and rounds the Profile size is implementation dependent. It is optional for the eUICC to support this data object.

5.7.16 Function (ES10c): EnableProfile

Related Procedures: Local and Remote Profile Management – Enable Profile

Function Provider Entity: LPA Services

Description:

This function is used to enable a Profile on the eUICC. The function makes the Target Profile enabled, and disables implicitly the Profile currently enabled on the Target Port, if any. This SHALL be performed in an atomic way, meaning that in case of any error during the command execution, the command SHALL stop and SHALL leave the involved Profiles in their original states prior to command execution.

If this function is called by RPM, it SHALL be treated as if the refreshFlag is set.

NOTE: Before calling the EnableProfile function with the refreshFlag not being set, the Device has the responsibility to ensure that the relevant conditions for use are met, as indicated in section 3.2.1.

For SEP, MEP-A1, and MEP-B, upon reception of the EnableProfile function, if the refreshFlag is not set, the eUICC SHALL:

- Check whether there is a proactive session ongoing on the Target Port (which the Device did not terminate). If so, the eUICC SHALL do one of the following:
  - terminate the EnableProfile command and return an error code `catBusy`.
  - internally terminate the proactive session on the Target Port and ignore any incoming TERMINAL RESPONSE from that proactive session.
- Close all logical channels on the Target Port which still have an application of the currently enabled Profile selected (which the Device did not close), without generating an error.

Regardless of the value of refreshFlag, the eUICC SHALL:

- Verify that the Profile identified by its AID or ICCID exists. Otherwise, the eUICC SHALL return an error code `iccidOrAidNotFound` for LPM and `commandError` for RPM.
- If the command is sent via RPM:
  - Verify that the SM-DP+ that sent the RPM Command is included in the Managing SM-DP+ List and is authorised to perform the RPM Command. Otherwise, the eUICC SHALL return an error code `commandError`.
  - If the Profile Metadata specifies an allowed eSIM CA RootCA public key identifier for the Managing SM-DP+: verify that the Subject Key Identifier of the eSIM CA RootCA Certificate corresponding to CERT.DPauth.SIG matches that value. Otherwise, the eUICC SHALL return an error code `commandError`.
- Verify that the Target Profile is in the Disabled state. Otherwise, the eUICC SHALL return an error code `profileNotInDisabledState`.
- If the command is sent via RPM:
  - If the Enable command was preceded by a Disable command in the same `LoadRpmPackageRequest`, the eSIM Port where the Profile was disabled is the Target Port for the Enable command. Otherwise, the Target Port SHALL be set as follows:

    - If no eSIM Port is available (i.e., has no Enabled Profile assigned), the eUICC SHALL return an error code `noEsimPortAvailable`.
    - For MEP-B, if `targetEsimPort` is not provided in the `LoadRpmPackageRequest` and if the Command Port is available, the eUICC SHALL select the Command Port as Target Port.
    - For MEP-A1 and MEP-B, if `targetEsimPort` is provided in the `LoadRpmPackageRequest` and if the indicated eSIM Port is available, the eUICC SHALL select this eSIM Port as Target Port.
    - For MEP-A2, the eUICC SHALL select any available eSIM Port as Target Port.

NOTE: The behaviour of the eUICC for all other combinations is implementation specific. (E.g., no eSIM Port or a non-available eSIM Port indicated for MEP-A1 and the eUICC selecting the eSIM Port or the eUICC generating an error.)

  - Verify that the Target Profile is not a Test Profile. Otherwise, the eUICC SHALL return an error code `disallowedForRpm`.
- If the Target Profile is not a Test Profile: check if the Profile Policy Rules of the Profile currently Enabled on the Target Port allow its disabling. Otherwise, the eUICC SHALL return an error code `disallowedByPolicy`.
- If the Target Profile is not an Enterprise Profile or a Test Profile: verify that the maximum number of non-Enterprise Profiles that can be Enabled according to the Reference Enterprise Rule is not exceeded. Otherwise, the eUICC SHALL return an error code `disallowedByEnterpriseRule.`
- If the Reference Enterprise Rule indicates `priorityEnterpriseProfile` and the Profile with the Reference Enterprise Rule is currently disabled: verify that the Target Profile is the correct Enterprise Profile or a Test Profile. Otherwise, the eUICC SHALL return an error code `disallowedByEnterpriseRule.`
- If the Profile currently Enabled on the Target Port is a Test Profile: verify
  - that the Target Profile is (another) Test Profile, or
  - if an Operational Profile was in Enabled state before the (first) Test Profile was enabled on the Target Port, that the Target Profile is this Operational Profile.
  - Otherwise, the eUICC SHALL return an error code `wrongProfileReenabling`.

If the refreshFlag is not set:

- For MEP-A2, the eUICC SHALL select an available eSIM port as Target Port. If no eSIM port is available, then the eUICC SHALL return an error code `noEsimPortAvailable.`
- the eUICC SHALL reset the PIN status on the Target Port
- the eUICC SHALL disable the Profile currently Enabled on the Target Port (if any), Enable the Target Profile and implicitly select the MF on the basic logical channel.
- The eUICC SHALL return OK to the LPAd

NOTE: For subsequent actions by the Device see section 3.2.1.

If the refreshFlag is set:

- If CAT is not initialised, the eUICC SHALL return an error code `commandError`.
- For MEP-A2, the eUICC MAY return an error code `commandError`.
- If there is a proactive session ongoing on the Target Port, the eUICC SHALL do one of the following:
    - terminate the EnableProfile command and return an error code `catBusy`.
    - internally terminate the proactive session on the Target Port. If a TERMINAL RESPONSE is still outstanding, the REFRESH command according to the step below SHALL only be sent after reception of the TERMINAL RESPONSE.

NOTE: In the case of the `catBusy` error, the Device MAY take implementation-dependent action to terminate the proactive command session, and MAY send the Enable command again without any further End User interaction.

- The eUICC SHALL mark the currently enabled Profile (if any) as "to be disabled", mark the Target Profile as "to be enabled" on the Target eSIM Port and return a status `ok` to the LPAd.
- The eUICC sends a REFRESH proactive command via one of following methods:
    - For MEP-A1, the eUICC SHALL first send an LSI COMMAND proactive command with the action "Proactive session request" on the Command Port. The "Proactive session request" action includes the Target Port number where a proactive command is pending. When the Device checks for pending proactive commands on the Target Port, the eUICC SHALL send the REFRESH command with "eUICC Profile State Change" mode (if supported by the Device) or "UICC Reset" mode to the Device on the Target Port where the Target Profile SHALL be enabled.
    - For SEP and MEP-B, the eUICC SHALL send the REFRESH command with "eUICC Profile State Change" mode (if supported by the Device) or "UICC Reset" mode to the Device on the Target Port where the Target Profile SHALL be enabled.

An MEP-Capable Device SHALL reset the Target Port instead of the UICC Interface when receiving a REFRESH with mode "UICC Reset" as specified in ETSI TS 102 223 [31].

- Upon reception of the Terminal Response with result "command performed successfully" or upon reset of either the eUICC or the Target Port in case of an MEP-Capable eUICC, the eUICC SHALL disable the currently Enabled Profile (if any) and Enable the Target Profile on the Target Port.
- Upon reception of the Terminal Response with result "temporary problem with executing command" or "permanent problem with executing command", the eUICC SHALL NOT disable the currently Enabled Profile (if any) nor Enable the Target Profile. Subsequent actions are implementation specific.

NOTE: In the case the Device supports the Single Wire Protocol interface to the eUICC, the Device SHOULD take the appropriate actions regarding this interface when Profile state is changed.

The eUICC MAY start a new proactive UICC session on the Target Port only after a new TERMINAL PROFILE command is executed.

- If a previously Enabled Profile was successfully disabled, the eUICC SHALL generate as many Notifications as configured in its metadata (`notificationConfigurationInfo`) in the format of `OtherSignedNotification`.
- If the Target Profile is successfully enabled, the eUICC SHALL generate as many Notifications as configured in its metadata (`notificationConfigurationInfo`) in the format of `OtherSignedNotification`.
- However, no Notifications SHALL be generated if the Target Profile or the previously Enabled Profile is a Test Profile.
- In failure cases, the eUICC SHALL not generate any notifications.

Command Data

The command data SHALL be coded as follows:

```
-- ASN1START
EnableProfileRequest ::= [49] SEQUENCE { -- Tag 'BF31'
   profileIdentifier CHOICE {
      isdpAid [APPLICATION 15] OctetTo16, -- AID, tag '4F'
      iccid Iccid -- ICCID, tag '5A'
   },
   refreshFlag BOOLEAN, -- indicating whether REFRESH is required
   targetEsimPort  INTEGER OPTIONAL-- #SupportedForMEPV3.0.0#
}
-- ASN1STOP
```

The LPA SHALL only provide `targetEsimPort` for MEP-A1.

Response Data

The response data SHALL be coded as follows:

```
-- ASN1START
EnableProfileResponse ::= [49] SEQUENCE { -- Tag 'BF31'
   enableResult INTEGER {
      ok(0),
      iccidOrAidNotFound(1),
      profileNotInDisabledState(2),
      disallowedByPolicy(3),
      wrongProfileReenabling(4),
      catBusy(5),
      disallowedByEnterpriseRule(6), -- #SupportedForEnterpriseV3.0.0#
      commandError(7),  -- #SupportedFromV3.0.0#
      disallowedForRpm(9),     -- #SupportedForRpmV3.0.0#
      noEsimPortAvailable(10), -- #SupportedForMEPV3.0.0# and
                                -- #SupportedForRpmV3.0.0#
      undefinedError(127)
   },
   targetEsimPort INTEGER OPTIONAL -- #SupportedForMEPV3.0.0#
}
-- ASN1STOP
```

`targetEsimPort` SHALL only be present in the response of a successful enabling of a Profile with LPM for MEP-A2.

Alternative Case 3 Command

In addition to the command data exchange described above, the following alternative command is defined for the function. The eUICC SHALL support this alternative if it supports only SEP. Otherwise, the usage of this alternative is out of scope of this specification.

The command data is sent to the eUICC in a Case 3 STORE DATA command:

- The STORE DATA APDU SHALL be coded as defined in section 5.7.2, with the exception of P1 which SHALL be set to '90'.
- The command data SHALL be coded as defined above, the response data SHALL NOT be present.

- The following additional status bytes are defined:
  '6A 82': Profile not found
  '69 85': Profile not in disabled state, command not allowed by Profile Policy Rules, Enterprise Rule or for RPM, or re-enabling wrong Profile
  '93 00': CAT is busy. Command cannot be executed at present, further normal commands are allowed

5.7.17 Function (ES10c): DisableProfile

Related Procedures: Local and Remote Profile Management – Disable Profile

Function Provider Entity: LPA Services

Description:

This function is used to disable a Profile on the eUICC.

If this function is called by RPM, it SHALL be treated as if the refreshFlag is set.

NOTE: Before calling the DisableProfile function with the refreshFlag not being set, the Device has the responsibility to ensure that the relevant conditions for use are met, as indicated in section 3.2.2.

Upon reception of the DisableProfile function, if the refreshFlag is not set, the eUICC SHALL:

- Check whether there is a proactive session ongoing on the Target Port (which the Device did not terminate). If so, the eUICC SHALL do one of the following:
  - terminate the DisableProfile function and return an error code `catBusy`.
  - internally terminate the proactive session on the Target Port and ignore any incoming TERMINAL RESPONSE from that proactive session.
- Close all logical channels which still have an application of the currently enabled Profile selected (which the Device did not close), without generating an error.

Regardless of the value of refreshFlag, the eUICC SHALL:

- Verify that the Profile identified by its AID or ICCID exists. Otherwise, the eUICC SHALL return an error code `iccidOrAidNotFound` for LPM and `commandError` for RPM.
- If the command is sent via RPM:

  - Verify that the SM-DP+ that sent the RPM Command is included in the Managing SM-DP+ List and is authorised to perform the RPM Command. Otherwise, the eUICC SHALL return an error code `commandError`.
  - If the Profile Metadata specifies an allowed eSIM CA RootCA public key identifier for the Managing SM-DP+: verify that the Subject Key Identifier of the eSIM CA RootCA Certificate corresponding to CERT.DPauth.SIG matches that value. Otherwise, the eUICC SHALL return an error code `commandError`.
  - Verify that the Target Profile is not a Test Profile. Otherwise, the eUICC SHALL return an error code `disallowedForRpm`.
- Verify that the Target Profile is in the Enabled state. Otherwise, the eUICC SHALL return an error code `profileNotInEnabledState`.
- If the currently Enabled Profile is not a Test Profile: check if the Profile Policy Rules of the Target Profile allows its disabling. Otherwise, the eUICC SHALL return an error code `disallowedByPolicy`.

If the refreshFlag is not set:

- the eUICC SHALL reset the PIN status.
- the eUICC SHALL disable the currently Enabled Profile.
- If the Target Profile is a Test Profile and an Operational Profile was in Enabled state before the (first) Test Profile was enabled on the Target Port: enable this Operational Profile on the Target Port. If this Operational Profile was deleted while the Test Profile was enabled, no error SHALL be generated and the function execution SHALL continue.
- The eUICC SHALL implicitly select the MF on the basic logical channel.
- The eUICC SHALL return OK to the LPAd.

NOTE: For subsequent actions by the Device see section 3.2.2.

If the refreshFlag is set:

- For MEP-A2, the eUICC MAY return an error code `commandError`.
- If there is a proactive session ongoing, the eUICC SHALL do one of the following:
  - terminate the DisableProfile function and return an error code `catBusy`
  - internally terminate the proactive session on the Target Port. If a TERMINAL RESPONSE is still outstanding, the REFRESH command SHALL only be sent after reception of the TERMINAL RESPONSE.

NOTE: In the case of the `catBusy` error, the Device MAY take implementation-dependent actions to terminate the proactive command session, and MAY send the Disable command again without any further End User interaction.

- The eUICC SHALL mark the currently enabled Profile as "to be disabled" and return a status `ok` to the LPAd.
- If CAT has been initialised on the Target Port, the eUICC SHALL send a REFRESH command via one of following methods:

- For MEP-A1, the eUICC SHALL first send an LSI COMMAND proactive command with the action "Proactive session request" on the Command Port. The "Proactive session request" action includes the Target Port number where a proactive command is pending. When the Device checks for pending proactive commands on the Target Port, the eUICC SHALL send the REFRESH command with "eUICC Profile State Change" mode (if supported by the Device) or "UICC Reset" mode to the Device on the Target Port where the Target Profile SHALL be disabled.
- For SEP and MEP-B, the eUICC SHALL send the REFRESH command with "eUICC Profile State Change" mode (if supported by the Device) or "UICC Reset" mode to the Device on the Target Port where the Target Profile SHALL be disabled.

An MEP-Capable Device SHALL reset the Target Port instead of the UICC Interface when receiving a REFRESH with mode "UICC Reset" as specified in ETSI TS 102.223 [31].

- Upon reception of the Terminal Response with result "command performed successfully" or upon reset of either the eUICC or the Target Port in case of an MEP-Capable eUICC or immediately if CAT has not been initialised on the Target Port, the eUICC SHALL disable the currently Enabled Target Profile on the Target Port. If the Target Profile is a Test Profile, an Operational Profile was in Enabled state before the (first) Test Profile was enabled on the Target Port and this Operational Profile was not deleted while a Test Profile was Enabled: the eUICC SHALL enable this Operational Profile on the Target Port. If this Operational Profile was deleted while the Test Profile was enabled, no error SHALL be generated and the function execution SHALL continue.
- Upon reception of the Terminal Response with result "temporary problem with executing command" or "permanent problem with executing command", the eUICC SHALL NOT disable the currently Enabled Profile. Subsequent actions are implementation specific.

NOTE: In the case the Device supports the Single Wire Protocol interface to the eUICC, the Device SHOULD take the appropriate actions regarding this interface when Profile state is changed.

The eUICC MAY start a new proactive UICC session on the Target Port only after a new TERMINAL PROFILE command is executed.

- If the Target Profile is successfully disabled, the eUICC SHALL generate as many Notifications as configured in its metadata (`notificationConfigurationInfo`) in the format of `OtherSignedNotification`.
- However, no Notifications SHALL be generated if the Target Profile is a Test Profile.
- In failure cases, the eUICC SHALL not generate any notifications.

Command Data

The command data SHALL be coded as follows:

```
-- ASN1START
DisableProfileRequest ::= [50] SEQUENCE { -- Tag 'BF32'
   profileIdentifier CHOICE {
      isdpAid [APPLICATION 15] OctetTo16, -- AID, tag '4F'
      iccid Iccid -- ICCID, tag '5A'
   },
   refreshFlag BOOLEAN -- indicating whether REFRESH is required
}
-- ASN1STOP
```

Response Data

The response data SHALL be coded as follows:

```
-- ASN1START
DisableProfileResponse ::= [50] SEQUENCE { -- Tag 'BF32'
   disableResult INTEGER {
      ok(0),
      iccidOrAidNotFound(1),
      profileNotInEnabledState(2),
      disallowedByPolicy(3),
      catBusy(5),
      commandError(7),  -- #SupportedFromV3.0.0#
      disallowedForRpm(10),     -- #SupportedForRpmV3.0.0#
      undefinedError(127)
   }
}
-- ASN1STOP
```

Alternative Case 3 Command

In addition to the command data exchange described above, the following alternative command is defined for the function. The eUICC SHALL support this alternative if it supports only SEP. Otherwise, the usage of this alternative is out of scope of this specification.

The command data is sent to the eUICC in a Case 3 STORE DATA command:

- The STORE DATA APDU SHALL be coded as defined in section 5.7.2, with the exception of P1 which SHALL be set to '90'.
- The command data SHALL be coded as defined above, the response data SHALL NOT be present.
- The following additional status bytes are defined:
  '6A 82': Profile not found.
  '69 85': Profile not in enabled state or command not allowed by Profile Policy Rules or for RPM.
  '93 00': CAT is busy. Command cannot be executed at present, further normal commands are allowed.

5.7.18 Function (ES10c): DeleteProfile

Related Procedures: Local and Remote Profile Management – Delete Profile

Function Provider Entity: ISD-R (LPA Services)

Description:

This function deletes a Profile from eUICC. This function can be used at any time by the LPAd. The Profile to be deleted can be identified by ISD-P AID or ICCID.

On reception of this command the eUICC SHALL:

- Verify that the target Profile identified by its AID or ICCID exists. Otherwise, the eUICC SHALL return an error iccidOrAidNotFound for LPM and commandError for RPM.
- If the command is sent via RPM:
  - Verify that the SM-DP+ that sent the RPM Command is included in the Managing SM-DP+ List and is authorised to perform the RPM Command. Otherwise, the eUICC SHALL return an error code commandError.
  - If the Profile Metadata specifies an allowed eSIM CA RootCA public key identifier for the Managing SM-DP+: verify that the Subject Key Identifier of the eSIM CA RootCA Certificate corresponding to CERT.DPauth.SIG matches that value. Otherwise, the eUICC SHALL return an error code commandError.
- Verify that the target Profile is not in Enabled state. Otherwise, the eUICC SHALL return an error profileNotInDisabledState.
- Verify that the Profile Policy Rules allow the deletion of the target Profile. Otherwise, the eUICC SHALL return an error disallowedByPolicy.
- If the currently enabled Profile is a Test Profile and the target Profile is not a Test Profile, the eUICC SHALL return an error disallowedInTestMode.
- Delete the ISD-P containing the target Profile and its related Profile Metadata.

If the target Profile is successfully deleted, the eUICC SHALL generate as many Notifications as configured in its metadata (notificationConfigurationInfo) in the format of OtherSignedNotification and return deleteResult with status code ok.

In failure cases, the eUICC SHALL not generate any notifications.

Command Data

The command data SHALL be coded as follows:

```
-- ASN1START
DeleteProfileRequest ::= [51] CHOICE { -- Tag 'BF33'
   isdpAid [APPLICATION 15] OctetTo16, -- AID, tag '4F'
   iccid Iccid -- ICCID, tag '5A'
}
-- ASN1STOP
```

Response Data

The response data SHALL be coded as follows:

```
-- ASN1START
DeleteProfileResponse ::= [51] SEQUENCE { -- Tag 'BF33'
   deleteResult INTEGER {
      ok(0),
      iccidOrAidNotFound(1),
      profileNotInDisabledState(2),
      disallowedByPolicy(3),
      disallowedInTestMode(4),  -- #SupportedFromV3.0.0#
      commandError(7),  -- #SupportedFromV3.0.0#
      undefinedError(127)
   }
}
```

```
-- ASN1STOP
```

Alternative Case 3 Command

In addition to the command data exchange described above, the following alternative command is defined for the function. The eUICC SHALL support this alternative if it supports only SEP. Otherwise, the usage of this alternative is out of scope of this specification.

The command data is sent to the eUICC in a Case 3 STORE DATA command:

- The STORE DATA APDU SHALL be coded as defined in section 5.7.2, with the exception of P1 which SHALL be set to '90'.
- The command data SHALL be coded as defined above, the response data SHALL NOT be present.
- The following additional status bytes are defined:
  '6A 82': Profile not found.
  '69 85': Profile not in disabled state or command not allowed by Profile Policy Rules or command not allowed in Test Mode.

5.7.19 Function (ES10c): eUICCMemoryReset

Related Procedures: eUICC Memory Reset

Function Provider Entity: LPA Services

Description:

This function deletes selected subsets of the Profiles stored on an eUICC regardless of their enabled status or any Profile Policy Rules. The following subsets are defined:

- Operational Profiles
- Test Profiles that were not pre-installed (i.e., Test Profiles that were "field loaded")
- Test Profiles that were pre-installed (i.e., Test Profiles that were loaded in the factory)
- Provisioning Profiles

NOTE: The identification of pre-installed Test Profiles is out of the scope of this specification.

The eUICC returns a status indicating whether any Profiles were deleted.

This function can also be used to reset the Default SM-DP+ address to its initial value.

Any combination MAY be specified.

If the eUICC does not support Test Profiles, then a request to delete them is ignored.

The eUICC Memory Reset SHALL be performed in an atomic and non-reversible way in case of external interruptions (e.g., power loss): the eUICC SHALL continue the processing of that command upon the next eUICC power on. In case of any other error during the command execution, the command SHALL stop and SHALL leave the eUICC and the involved Profiles in their original states prior to command execution.

Upon reception of the eUICCMemoryReset function and dependent on the parameters set, the eUICC SHALL do the following:

- In case there is a proactive session ongoing on the Command Port, the eUICC SHALL do one of the following:
  - terminate the command and return an error code `catBusy`.
  - internally terminate the proactive session and send the REFRESH command(s) as the next proactive command on the Command Port. If a TERMINAL RESPONSE is still outstanding, the REFRESH command SHALL only be sent after reception of the TERMINAL RESPONSE.

NOTE: In the case of the `catBusy` error, the Device MAY take implementation-dependent action to terminate the proactive command session, and MAY send the eUICC Memory Reset command again without any further End User interaction.

- Delete all the selected ISD-Ps with their Profiles regardless of their enabled status or Profile Policy Rules and all related Profiles Metadata stored in the ISD-R. The ISD-R SHALL send a proactive command to the Device to reset the eUICC. For SEP, the ISD-R SHALL send a REFRESH with mode "UICC Reset". For MEP, the ISD-R SHALL send an LSI COMMAND with "UICC Platform Reset".
- Reset the Default SM-DP+ address and its associated allowed eSIM CA RootCA public key to their default values, respectively.
  For each deleted Profile, the eUICC SHALL generate as many Notifications as configured in its Profile Metadata (`notificationConfigurationInfo`) in the format of `OtherSignedNotification`.
- If the command was sent while a Test Profile was in Enabled state and this Profile was deleted by the command and an Operational Profile was in Enabled state before the (first) Test Profile was enabled and this Operational Profile was not deleted by the command, then this previous Operational Profile SHALL be enabled again.

For each deleted Profile, the eUICC SHALL generate as many Notifications as configured in its Profile Metadata (`notificationConfigurationInfo`) in the format of `OtherSignedNotification`.

Command Data

The command data SHALL be coded as follows:

```
-- ASN1START
EuiccMemoryResetRequest ::= [52] SEQUENCE { -- Tag 'BF34'
   resetOptions [2] BIT STRING {
      deleteOperationalProfiles(0),
      deleteFieldLoadedTestProfiles(1),
      resetDefaultSmdpAddress(2),
      deletePreLoadedTestProfiles(3), -- #SupportedFromV3.0.0#
      deleteProvisioningProfiles(4)} -- #SupportedFromV3.0.0#
      -- setting bits 0, 1, 3 and 4 wipes all Profiles
}
-- ASN1STOP
```

Response Data

The response data SHALL be coded as follows:

```
-- ASN1START
EuiccMemoryResetResponse ::= [52] SEQUENCE { -- Tag 'BF34'
   resetResult INTEGER {ok(0), nothingToDelete(1), catBusy(5), undefinedError(127)}
}
-- ASN1STOP
```

5.7.20 Function (ES10c): GetEID

Related Procedures: Profile Download Initiation

Function Provider Entity: ISD-R (LPA Services)

Description:

This function gets the EID from the eUICC. This function can be used at any time by the LPA, and for instance during the Profile Download Initiation when the End user MAY have to provide the EID to the contracting Service Provider/Operator, and when the EID is not available by another mean, e.g., the End User MAY have lost the physical material where it was printed on.

Command Data

The data field SHALL indicate EID data object '5C 01 5A' (tag '5A' identifies the EID).

The command data SHALL be coded as follows:

```
-- ASN1START
GetEuiccDataRequest ::= [62] SEQUENCE { -- Tag 'BF3E'
   tagList [APPLICATION 28] Octet1  -- tag '5C', the value SHALL be set to '5A'
}
-- ASN1STOP
```

Response Data

The response data SHALL be coded as follows:

```
-- ASN1START
GetEuiccDataResponse ::= [62] SEQUENCE { -- Tag 'BF3E'
   eidValue [APPLICATION 26] Octet16  -- tag '5A'
}
-- ASN1STOP
```

In case the provided `tagList` is invalid or unsupported, the eUICC SHALL return an error status word.

5.7.21 Function (ES10c): SetNickname

Related Procedures: Set/Edit Nickname

Function Provider Entity: LPA Services

Description:

This function is used to add or update a Profile Nickname associated to one Profile present on-card.

Upon reception of the SetNickname function, the eUICC SHALL:

- Verify that the target Profile is present on the eUICC. Otherwise, the eUICC SHALL return an error code `iccidNotFound`.
- Update the target Profile nickname with the provided data and return `setNicknameResult` with status code `ok`.

In case a Profile Nickname already exists for the indicated Profile, the Profile Nickname SHALL be updated with the new value. In case the new value is an empty string, the Profile Nickname SHALL be removed. Removing a non-existing Profile Nickname SHALL NOT be considered an error.

Command Data

The command data SHALL be coded as follows:

```
-- ASN1START
-- Definition of Profile Nickname Information
SetNicknameRequest ::= [41] SEQUENCE { -- Tag 'BF29'
   iccid Iccid,
   profileNickname [16] UTF8String (SIZE(0..64))
}
-- ASN1STOP
```

Response Data

The response data SHALL be coded as follows:

```
-- ASN1START
SetNicknameResponse ::= [41] SEQUENCE { -- Tag 'BF29'
   setNicknameResult INTEGER {ok(0), iccidNotFound (1), undefinedError(127)}
}
-- ASN1STOP
```

5.7.22 Function (ES10b): GetRAT

Related Procedures: Profile Download and Installation

Function Provider entity: ISD-R (LPA Services)

Description:

This function retrieves the Rules Authorisation Table (RAT) from the eUICC. It can be called at any time. The RAT is used by the LPA to determine if a Profile containing PPRs can be installed, conditionally with End User Consent, on the eUICC as defined in section 2.9.2.4.

Command data

The command data SHALL be coded as follows:

```
-- ASN1START
GetRatRequest ::= [67] SEQUENCE { -- Tag 'BF43'
   -- No input data
}
```

```
-- ASN1STOP
```

Response data

The response data SHALL be coded as follows:

```
-- ASN1START
GetRatResponse ::= [67] SEQUENCE { -- Tag 'BF43'
   rat RulesAuthorisationTable
}

RulesAuthorisationTable ::= SEQUENCE OF ProfilePolicyAuthorisationRule
ProfilePolicyAuthorisationRule ::= SEQUENCE {
   pprIds PprIds,
   allowedOperators SEQUENCE OF OperatorId,
   pprFlags BIT STRING {consentRequired(0)}
}
-- ASN1STOP
```

The list of `ProfilePolicyAuthorisationRule` data objects SHALL be returned in the same order as stored in the eUICC. This list MAY be empty.

The `pprIds` data object SHALL identify at least one PPR. The LPA SHALL ignore the `pprUpdateControl` bit.

The `allowedOperators` data object SHALL follow the description given in section 2.9.2.1.

The `consentRequired` bit set to 1 indicates that the End User consent is required.

5.7.23 Function (ES10c): LPA alerting

Related Procedures: Metadata Update, Pending operation alerting

Function Provider Entity: LPAd

Description:

This function alerts the LPAd about:

- Changes in the Metadata of a Profile after the execution of an Update Metadata function on ES6 (see section 5.4.1), which MAY require the LPAd to take an action.
- Pending events, which the LPAd is requested to retrieve.

If support for Metadata update alerting is indicated in the RSP Device Capabilities and Metadata objects are updated via ES6, the eUICC SHALL alert the LPA by sending a REFRESH proactive command to the Device with the following parameters:

- Refresh mode SHALL be set to "Application Update".
- AID SHALL contain the AID of the ISD-R.
- Refresh enforcement policy SHALL be absent.
- The value field of the application specific refresh data SHALL contain the `AlertData` object defined below.

An application within a Profile MAY also use this command to alert the LPAd about pending RSP operations. Support for pending operation alerting is optional for the Device. If supported, it SHALL be indicated by setting `pendingOperationAlertingSupport` in `LpaRspCapability`. This allows a Profile Owner to appropriately configure an application within a Profile.

Definition of `AlertData`:

```
-- ASN1START
AlertData ::= [74] CHOICE { -- Tag 'BF4A' #SupportedFromV3.0.0#
   metadataUpdateEnabledProfile [0] MetadataUpdateEnabledProfile,
   pendingOperationAlert [1] ServerWithPendingOperation
}

MetadataUpdateEnabledProfile ::= SEQUENCE {
   iccid Iccid OPTIONAL,
   tagList [APPLICATION 28] OCTET STRING -- tag '5C'
}

ServerWithPendingOperation ::= CHOICE {
      pollingAddress [0] NULL,
      rootSmds [1] NULL,
      defaultSmdp [2] NULL,
      explicitAddress [3] UTF8String
}
-- ASN1STOP
```

The `iccid` SHALL be provided for MEP-A1 and MEP-A2 and MAY be provided for SEP and MEP-B. If provided, it SHALL contain the ICCID of the Profile processing the ES6 command.

The `tagList` SHALL contain all tags of objects that were included in the UpdateMetadata function (see section 5.4.1).

If the Device supports refresh mode "Application Update" and accepts other `AlertData` settings (e.g., for Metadata update alerting) where the AID indicates the ISD-R, but does not support event alerting, it SHALL respond with "command data not understood by terminal" to the proactive command.

NOTE: This is the same error code that the Device will use to indicate that it does not have interest for the refresh information related to a given AID, see ETSI TS 102 223 [31].

5.7.24 Function (ES10a): VerifySmdsResponse

Related Procedures: Event Retrieval and Push Service Registration

Function Provider Entity: ISD-R (LPA Services)

#SupportedFromV3.0.0#

Description:

This function verifies the signature of `smdsSigned2` returned to the LPA by the SM-DS as a response from "ES11.AuthenticateClient" function.

On reception of this command the eUICC SHALL:

- Verify that the `transactionId` matches the one in the ongoing RSP session. Otherwise, the eUICC SHALL return an error code `invalidTransactionId`.

- Verify that the SM-DS has been previously authenticated and a CERT.DSauth.SIG is attached to the ongoing RSP session. Otherwise, the eUICC SHALL return an error code `noSession`.

- Verify that `smdsSignature2` created over the concatenated data objects `smdsSigned2` (as described hereunder) and `euiccSignature1` is valid. Otherwise, the eUICC SHALL return an error code `invalidSignature`.

- Return `verifySmdsResponseOk` if all of these verifications succeed.

- End the RSP session in both cases (error and ok).

Command Data

The command data SHALL be coded as follows:

```
-- ASN1START
VerifySmdsResponseRequest ::= [69] SEQUENCE { -- Tag 'BF45' #SupportedFromV3.0.0#
   smdsSigned2 [1] SmdsSigned2, -- must have tag [1]
   smdsSignature2 [APPLICATION 55] OCTET STRING
}

SmdsSigned2 ::= SEQUENCE {
   transactionId [0] TransactionId,
   requestSpecificData CHOICE {
      eventList [0] SEQUENCE {
         eventEntries [1] SEQUENCE OF EventRecordV3,
         ecId [2] OCTET STRING(SIZE(16..32)) OPTIONAL, --
#SupportedForEventCheckingV3.0.0# Event Checking ID
         pushServiceRefreshTime [3] GeneralizedTime OPTIONAL --
#SupportedForPushServiceV3.0.0# date and time to re-register a Push Token to the
SM-DS
      },
      pushServiceRegistrationResult [1] SEQUENCE {
         pushServiceRefreshTime [3] GeneralizedTime OPTIONAL --
#SupportedForPushServiceV3.0.0# date and time to re-register a Push Token to the
SM-DS
      }
   }
}

EventRecordV3 ::= SEQUENCE { -- #SupportedFromV3.0.0#
   eventId UTF8String,
   rspServerAddress UTF8String,
   eventType INTEGER, -- either 1 (for Profile Download) or 2 (for RPM)
   hashedIccids SEQUENCE OF OCTET STRING (SIZE(32)) OPTIONAL, -- hashed ICCID(s)
calculated as either SHA256(ICCID) or SHA256(ICCID|Salt)
   salt OCTET STRING (SIZE(8..16)) OPTIONAL, -- optional salt to be concatenated
with ICCID(s) for hashing
   serviceProviderName [17] UTF8String (SIZE(0..32)) OPTIONAL,
   operatorId [23] OperatorId OPTIONAL
}
-- ASN1STOP
```

`smdsSignature2` SHALL be created by the SM-DS using the SK.DSauth.SIG and verified by the eUICC using the PK.DSauth.SIG as described in section 2.6.9. The eUICC is not required to verify the validity and format of content within `smdsSigned2` other than `transactionId`. If the eUICC does verify other content within `smdsSigned2`, it SHALL ignore any unknown tags.

Response Data

The response data SHALL be coded as follows:

```
-- ASN1START
VerifySmdsResponseResponse ::= [69] CHOICE {  -- Tag 'BF45' #SupportedFromV3.0.0#
   verifySmdsResponseOk NULL,
   verifySmdsResponseError INTEGER {
      invalidSignature(2),
      noSession(4),
      invalidTransactionId(5),
      undefinedError(127)
   }
}
-- ASN1STOP
```

5.7.25 Function (ES10b): LoadRpmPackage

Related Procedures: RPM Download and Execution

Function Provider Entity: ISD-R (LPA Services)

Description:

This function initiates execution of an RPM Package.

On reception of this command, the eUICC SHALL:

- Verify that the SM-DP+ has been previously authenticated. Otherwise, the eUICC SHALL return an unsigned error code `noSession`.
- Verify `smdpSignature3` created over the concatenated data objects `smdpSigned3` and `euiccSignature1` using the PK.DPauth.SIG contained in CERT.DPauth.SIG of the on-going RSP session. If the signature is invalid, the eUICC SHALL return an `invalidSignature` error code.
- Verify that the `transactionId` contained in the `smdpSigned3` matches the one of the on-going RSP session. If it does not match, the eUICC SHALL return an `invalidTransactionId` error code.
- Execute the RPM Command(s) contained the RPM Package.
- Generate and return `LoadRpmPackageResult` containing the execution result of the RPM Package.

The eUICC SHALL end the RSP session in any case after processing this function.

Command Data

The command data SHALL be coded as follows:

```
-- ASN1START
LoadRpmPackageRequest ::= [68] SEQUENCE { -- #SupportedForRpmV3.0.0# Tag 'BF44'
   smdpSigned3 SmdpSigned3,
   smdpSignature3 [APPLICATION 55] OCTET STRING, -- tag '5F37'
   targetEsimPort INTEGER OPTIONAL
}
```

```
SmdpSigned3 ::= SEQUENCE {
   transactionId [0] TransactionId, -- The TransactionID generated by the SM-DP+
   rpmPackage [1] RpmPackage,
   rpmPending [2] NULL OPTIONAL
}
-- ASN1STOP
```

Response Data

The `LoadRpmPackageResult` as specified in section 2.10.2.

5.7.26 Function (ES10b): PrepareDeviceChange

Related Procedures: Device Change

Function Provider Entity: ISD-R (LPA Services)

Description:

This function initiates a Device Change or Profile Recovery after a successful authentication of an SM-DP+.

On reception of this command, the eUICC SHALL:

- Verify that the SM-DP+ has been previously authenticated. Otherwise, the eUICC SHALL return an error code `noSession`.
- Verify the `smdpSignature4` using the PK.DPauth.SIG contained in CERT.DPauth.SIG of the on-going RSP session. If the signature is invalid, the eUICC SHALL return an error code `invalidSignature`.
- Verify that the received `transactionId` contained in the `smdpSigned4` matches the one of the on-going RSP session. Otherwise, the eUICC SHALL return an error code `invalidTransactionId`.
- Upon any error returned in these verifications, the eUICC SHALL terminate the RSP session.
- If the eUICC supports the encrypted Device Change data in Device Change response: generate a new one-time KA key pair (otSK.EUICC.KA, otPK.EUICC.KA) using the parameters indicated by the `subjectPublicKeyInfo.algorithmIdentifier.parameters` field of the CERT.DPauth.SIG, and attach otSK.EUICC.KA to the RSP session.
- Generate `euiccSigned3` data object as defined hereunder which MAY include vendor-specific additional information (e.g., as described in Annex P).
- Compute the `euiccSignature3` over the concatenated data objects `euiccSigned3` and `smdpSignature3` using the SK.EUICC.SIG that was used in the "ES10b.AuthenticateServer response".

Command Data

The command data SHALL be coded as follows.

```
-- ASN1START
PrepareDeviceChangeRequest ::= [77] SEQUENCE { -- #SupportedForDcV3.0.0# Tag 'BF4D'
   smdpSigned4 SmdpSigned4, -- Signed information
   smdpSignature4 [APPLICATION 55] OCTET STRING, -- tag '5F37'
   hashCc Octet32 OPTIONAL -- Hash of confirmation code
}

SmdpSigned4 ::= SEQUENCE { -- #SupportedForDcV3.0.0#
   transactionId [0] TransactionId,      -- The TransactionID generated by the
SM-DP+
   ccRequiredFlag BOOLEAN, -- Indicates if the Confirmation Code is required
   activationCodeForProfileRecovery [1] UTF8String (SIZE(0..255)) OPTIONAL --
presents only in ES9+.AuthenticateClient response for a profileRecoveryRequest
}
-- ASN1STOP
```

Response Data

The response data SHALL be coded as follows.

```
-- ASN1START
PrepareDeviceChangeResponse ::= [77] CHOICE { -- #SupportedForDcV3.0.0# Tag 'BF4D'
   downloadResponseOk PrepareDeviceChangeResponseOk,
   downloadResponseError PrepareDeviceChangeResponseError
}

PrepareDeviceChangeResponseOk ::= SEQUENCE { -- #SupportedForDcV3.0.0#
   euiccSigned3 EUICCSigned3, -- Signed information
   euiccSignature3 [APPLICATION 55] OCTET STRING -- tag '5F37'
}

EUICCSigned3 ::= SEQUENCE { -- #SupportedForDcV3.0.0#
   transactionId [0] TransactionId,
   euiccOtpk [APPLICATION 73] OCTET STRING OPTIONAL, -- otPK.EUICC.KA, tag '5F49'
   hashCc Octet32 OPTIONAL, -- Hash of confirmation code
   additionalInformation VendorSpecificExtension OPTIONAL
}

PrepareDeviceChangeResponseError ::= SEQUENCE { -- #SupportedForDcV3.0.0#
   transactionId [0] TransactionId,
   downloadErrorCode DownloadErrorCode
}
-- ASN1STOP
```

In case of the error `invalidTransactionId`, the `transactionId` in the `PrepareDownloadResponse` SHALL be set to the value from the `AuthenticateServerRequest`.

5.7.27 Function (ES10b): VerifyDeviceChange

Related Procedures: Device Change, Profile Recovery

Function Provider Entity: ISD-R (LPA Service)

Description:

This function is used by the LPAd to verify the `smdpSignature5` returned by the SM-DP+ contained in the response of ES9+.ConfirmDeviceChange during the Device Change procedure or the `smdpSignature4` returned by the SM-DP+ contained in the response of ES9+.AuthenticateClient during the Profile Recovery procedure, via the eUICC.

On the reception of this command, the eUICC SHALL:

- Verify that the SM-DP+ has been previously authenticated. Otherwise, the eUICC SHALL return an error code `noSession`.
- Verify that the `transactionId` contained in the `smdpSigned4` or the `smdpSigned5` matches the one of the on-going RSP session. If it does not match, the eUICC SHALL return an error code `invalidTransactionId`.
- If the LPAd provide the `deviceChangeConfirmation,` then:
  - Verify the `smdpSignature4` using the PK.DPauth.SIG attached to the ongoing RSP session. If the signature is invalid, the eUICC SHALL return an error code `invalidSignature`.
  - If `DeviceChangeResponse` in `smdpSigned4` contains `encryptedDeviceChangeData`, then:
    - If the eUICC does not support encrypted Device Change data, an `invalidData` error SHALL be returned.
    - Verify that Control Reference Template describing the keys to generate matches the values defined here under (Command message part). Otherwise, an `unsupportedCrtValues` error SHALL be returned.
    - Generate the session keys (S-ENC and S-MAC) and the initial MAC chaining value from received otPK.DP.KA and previously generated otSK.EUICC.KA, using the key agreement algorithm determined according to section 2.6.5.
    - Perform decryption and MAC verification of `sequenceOf87` data object with the session key. On any failure during the processing of `sequenceOf87`, an `invalidData` error SHALL be returned.
    - If the Profile identified by the ICCID is not in disabled state, a `profileNotInDisabledState` error SHALL be returned.
    - Delete the Profile identified by the ICCID.
    - Return `verifyDeviceChangeData` containing decrypted `DeviceChangeData` TLV.
  - Otherwise:
    - Return `verifyDeviceChangeNull`.
- If the LPAd provide the `profileRecoveryResponse,` then:
  - Verify the `smdpSignature2` using the PK.DPauth.SIG attached to the ongoing RSP session. If the signature is invalid, the eUICC SHALL return an error code `invalidSignature`.
  - Return `verifyDeviceChangeNull` if all the above verifications succeed.
- Discard the RSP session in both cases.

Command Data

The command data SHALL be coded as follows.

```
-- ASN1START
VerifyDeviceChangeRequest ::= [75] CHOICE { -- Tag 'BF4B' #SupportedForDcV3.0.0#
   deviceChangeConfirmation [0] SEQUENCE {
      smdpSigned5 SmdpSigned5, -- Signed information
      smdpSignature5 [APPLICATION 55] OCTET STRING
   },
   profileRecoveryResponse [1] SEQUENCE {
      smdpSigned4 SmdpSigned4, -- Signed information
      smdpSignature4 [APPLICATION 55] OCTET STRING -- tag '5F37'
   }
}

SmdpSigned5 ::= SEQUENCE { -- #SupportedForDcV3.0.0#
   transactionId [0] TransactionId,
   deviceChangeResponse [1] DeviceChangeResponse OPTIONAL
}

DeviceChangeResponse ::= CHOICE {
   deviceChangeData [0] DeviceChangeData,
   encryptedDeviceChangeData [1] EncryptedDeviceChangeData
}

DeviceChangeData ::= SEQUENCE { -- #SupportedForDcV3.0.0#
   iccid Iccid,
   activationCodeForDc [0] UTF8String (SIZE(0..255)),
   deleteOldProfile [1] NULL OPTIONAL, -- Deletion of the installed Profile
required
   deleteNotificationForDcSupport [2] NULL OPTIONAL, -- Delete Notification for
Device Change supported
   notificationAddress [3] UTF8String OPTIONAL, -- FQDN that processes the Delete
Notification for Device Change
   profileRecoverySupport [4] NULL OPTIONAL,
   profileRecoveryValidityPeriod [5] GeneralizedTime OPTIONAL -- Absolute date and
time for Profile Recovery
}

EncryptedDeviceChangeData ::= SEQUENCE { -- #SupportedForDcV3.0.0#
   controlRefTemplate [6] IMPLICIT ControlRefTemplate,
   smdpOtpk [APPLICATION 73] OCTET STRING, -- okPK.DP.KA
   sequenceOf87 [1] SEQUENCE OF [7] OCTET STRING -- sequence of '87' TLVs
}
-- ASN1STOP
```

`sequenceOf87` data object contains protected TLV of `DeviceChangeData`. The concatenated TLVs are protected with session keys resulting from the key agreement (S-ENC, S-CMAC) (section 2.6.4).

The eUICC SHALL verify the values provided for key type and key length match the expected symmetric encryption algorithm according to section 2.6.5:

- When AES-128 is selected by the SM-DP+, `keyType` SHALL contain value '88' and `keyLen` SHALL contain '10'.
- When SM4 is selected by the SM-DP+, `keyType` SHALL contain value '89' and `keyLen` SHALL contain '10'.

NOTE: Key type values are assigned in the GlobalPlatform Card Specification [8].

Response Data

The response data SHALL be coded as follows:

```
-- ASN1START
VerifyDeviceChangeResponse ::= [75] CHOICE { -- Tag 'BF4B' #SupportedForDcV3.0.0#
   verifyDeviceChangeOk CHOICE {
      verifyDeviceChangeNull [0] NULL,
      verifyDeviceChangeData [1] DeviceChangeData
   },
   verifyDeviceChangeError INTEGER {
      invalidSignature(2),
      noSession(4),
      invalidTransactionId(5),
      unsupportedCrtValues(6),
      invalidData(7),
      profileNotInDisabledState(8),
      undefinedError(127)
   }
}
-- ASN1STOP
```

If the eUICC indicates `profileNotInDisabledState` error, the LPA MAY disable the target Profile and retry ES10b.VerifyDeviceChange function call.

5.8 ES11 (LPA -- SM-DS)

ES11 is the interface between:

- The LPA entity (more specifically the LDS endpoint).
- The SM-DS.

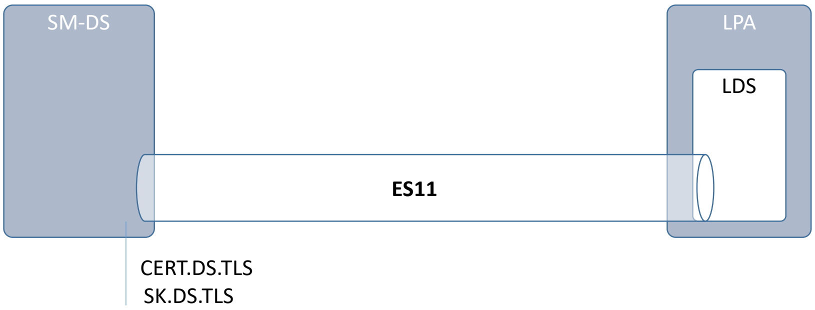


Figure 38: ES11

The LPA SHALL communicate with the SM-DS secured by TLS in server authentication mode as described in section 2.6.6.

The format of the TLS Certificates (CERT.DS.TLS) used for TLS connections is described in section 4.5.2.1.

During TLS establishment, LPA SHALL verify the received CERT.DS.TLS according to section 4.5.2.2. If any of these verifications fail, the TLS connection SHALL be rejected, and the on-going procedure SHALL fail.

5.8.1 Function: InitiateAuthentication

Related Procedures: Common Mutual Authentication for Event Retrieval

Function Provider Entity: SM-DS

Description:

This function is identical to "ES9+.InitiateAuthentication" where the SM-DS plays the role of SM-DP+ and where:

- The SM-DP+ SHALL be replaced by the SM-DS.
- CERT.DPauth.SIG SHALL be replaced by CERT.DSauth.SIG.

NOTE: The input data object 'smdpAddress' contains an SM-DS address in this case.

Specific Status Codes

Subject Code	Subject	Reason code	Reason	Description
8.9.1	SM-DS Address	3.8	Refused	Invalid SM-DS Address.
8.9.2	Security configuration	3.1	Unsupported	None of the proposed Public Key Identifiers is supported by the SM-DS.
8.9.4	SM-DS Certificate	3.7	Unavailable	The SM-DS has no CERT.DS.SIG which chains to one of the eSIM CA RootCA Certificate Public Key supported by the eUICC.

Table 48a: InitiateAuthentication Specific Status codes

5.8.2 Function: AuthenticateClient

Related Procedures: Common mutual authentication for Event Retrieval or Push Service Registration

Function Provider Entity: SM-DS

Description:

This function SHALL be called by the LPA to request the authentication of the eUICC by the SM-DS.

This function is correlated to a previous normal execution of an "ES11.InitiateAuthentication" function through a Transaction ID delivered by the SM-DS.

On reception of this function call, the SM-DS SHALL:

- Verify the validity of the eUICC certificate chain, as defined in section 4.5.2.2. If the eUICC Certificate (or any of the certificates in the chain) is invalid or expired, the SM-DS SHALL return a status code "eUICC Certificate - Verification failed" or "eUICC Certificate - Expired" respectively. If any of the certificates is missing in the chain, the SM-DS SHALL return a status code "eUICC Certificate - Verification failed".
- Verify that the Root Certificate of the eUICC certificate chain corresponds to the `euiccCiPKIdToBeUsed` or `euiccCiPKIdToBeUsedV3` that the SM-DS selected when executing the "ES11.InitiateAuthentication" function. If it doesn't correspond or the chain variant doesn't match, the SM-DS SHALL return a status code "CI Public Key - Unknown".
- Verify the eUICC signature (`euiccSignature1`) using the PK.EUICC.SIG as described in section 5.7.13 "ES10b.AuthenticateServer". Otherwise, the SM-DS SHALL return a status code "eUICC - Verification failed".
- Verify that the `transactionId` is known and relates to an ongoing RSP session. Otherwise, the SM-DS SHALL return a status code "TransactionId - Unknown".

- Verify that the serverChallenge attached to the ongoing RSP session matches the serverChallenge returned by the eUICC. Otherwise, the SM-DS SHALL return a status code "eUICC - Verification failed".
- If euiccRspCapability.euiccRspCapInInfo1 is set to '1', verify that euiccRspCapability was present in euiccInfo1 as received in ES11.InitiateAuthentication and that euiccRspCapability in euiccInfo2 matches the value received in euiccInfo1. Otherwise the SM-DS SHALL return a status code "eUICC - Value has Changed".
- Verify that lpaRspCapability in deviceInfo matches the value received in ES11.InitiateAuthentication. Otherwise the SM-DS SHALL return a status code "LPA - Value has Changed".

The SM-DS SHALL identify the type of operation requested by the LPA by checking the content of ctxParams1.

- If it contains ctxParamsForCommonAuthentication, the SM-DS SHALL regard this as an Event Retrieval request.
- If it contains ctxParamsForPushServiceRegistration, the SM-DS SHALL regard this as a Push Service Registration request.
- Otherwise, the SM-DS SHALL regard this as an error and SHALL return a status code "Function – Unsupported".

<u>Beginning of Event Retrieval operation</u>

The SM-DS SHALL:

- Ignore the iccid (if any), and retrieve the Event ID, RSP Server address, Event type, optional hashed ICCID, optional Salt, optional Service Provider name, and optional Operator ID tuple(s) for the requesting eUICC.
- Generate the list of Event Records. The list of Event Records SHALL be determined depending on the content of the received ctxParams1:
    - The SM-DS SHALL retrieve the operationType value contained in ctxParamsForCommonAuthentication. If it is not present, the SM-DS SHALL use the value of "Profile Download" in the following steps.
    - If ctxParamsForCommonAuthentication contains no matchingId data object or an empty value for matchingId, the SM-DS SHALL retrieve and return a list of the Event Records of which the EID and Event type match the requesting eUICC and operationType, respectively. This list MAY be empty. In addition to a list of the Event Records, if the SM-DS and the LPA use a Push Service, the SM-DS MAY return pushServiceRefreshTime.
    - If ctxParamsForCommonAuthentication contains a matchingId data object with an eventId value, the SM-DS SHALL retrieve and return the Event Records of which the EID, Event ID, and Event Type match the requesting eUICC, eventId, and operationType, respectively. If no Event Record is identified:
        - if ctxParamsForCommonAuthentication contains a matchingIdSource data object set to an SM-DS OID, the SM-DS SHOULD delete the Event Record from the SM-DS identified by that OID.

  - the SM-DS SHALL return a status code "Event Record - Unknown".

- If the LPA indicates eventCheckingSupport and the SM-DS supports the Event Checking, generate an ECID randomly and associate the ECID with the EID in the CERT.EUICC.SIG. The ECID SHALL be unique within the scope of the SM-DS and SHALL NOT contain the EID in plaintext.

NOTE: SM-DS may implement proprietary mechanisms to detect and delete unused ECID.

- If the LPA indicates signedSmdsResponseV3Support, generate an smdsSigned2 data object containing the Transaction ID, list of Event Records, optionally the ECID and pushServiceRefreshTime, and compute the signature smdsSignature2 over the concatenated data objects smdsSigned2 and euiccSignature1 as defined in "ES10a.VerifySmdsResponse" (section 5.7.24). Otherwise, create a response comprising the Transaction ID and list of Event Records as defined in version 2 of this specification.

<u>End of Event Retrieval operation</u>

<u>Beginning of Push Service Registration operation</u>

The SM-DS SHALL:

- Return an error status "Push Service – Unsupported" if the SM-DS does not support the Push Service
- Verify that the Push Service contained in the selectedPushService field is supported by the SM-DS. If verification fails, the SM-DS SHALL return an error status "Push Service Registration – Unsupported".
- Store the received Push Token and associate the Push Token with the EID of the requesting eUICC.

  NOTE: The SM-DS may perform additional operations to enable the selected Push Service e.g., interaction with the push server, which are out of scope of this specification.

- Generate an smdsSigned2 data object containing the Transaction ID and optionally pushServiceRefreshTime, and compute the signature smdsSignature2 over the concatenated data objects smdsSigned2 and euiccSignature1as defined in "ES10a.VerifySmdsResponse" (section 5.7.24).

<u>End of Push Service Registration operation</u>

The SM-DS MAY perform additional operations, which are out of scope of this specification.

This function SHALL return one of the following:

- A 'Function execution status' with 'Executed-Success' indicating that Event Retrieval has been successfully executed.

- A 'Function execution status' indicating 'Failed' with a status code as defined in section 5.2.6 or a specific status code as defined in the following table.

Additional Input Data:

Input data name	Description	Type	No.	MOC
transactionId	Transaction ID as generated by the SM-DS (section 3.0.1).	Binary[1-16]	1	M
authenticateServerResponse	Authenticate Server Response.	Binary (1)	1	M
NOTE 1: AuthenticateServerResponse data object defined in section 5.7.13 (function "ES10b.AuthenticateServer").				

Table 49: AuthenticateClient Additional Input Data

Additional Output Data:

Output data name	Description	Type	No.	MOC
transactionId	Transaction ID as generated by the SM-DS (section 3.0.1).	Binary[1-16]	1	M
eventEntries	The list of Event Records (Event ID and RSP Server address pair(s)) for the EID.	EVENT_RECORD	0..N	C(2)
smdsSigned2	Either Event Record(s) for the EID or Push Service Registration result.	Binary(1)	1	C(2)
smdsSignature2	SM-DS signature	Binary(1)	1	C(2)
NOTE 1: `smdsSigned2` and `smdsSignature2` are the data objects defined in section 5.7.24 (function "ES10a.VerifySmdsResponse"). They SHALL be returned as encoded data objects including the tags defined for them in the `VerifySmdsResponseRequest` data object. NOTE 2: In case of an Event Retrieval request, when the LPA indicates `signedSmdsResponseV3Support,` the SM-DS SHALL return `smdsSigned2` and `smdsSignature2` data objects, otherwise it SHALL return an eventEntries data object.				

Table 50: AuthenticateClient Additional Output Data

The **EVENT_RECORD** type is used only when the LPA does not support signed SM-DS event records. It is defined by the following data structure:

Data name	Description	Type	No.	MOC
eventId	Identification of the Event.	String	1	M
rspServerAddress	RSP Server address where the operation corresponding to the Event can be processed.	FQDN	1	M

Table 51: EVENT_RECORD

Specific Status Codes

Subject Code	Subject	Reason code	Reason	Description
8.1.3	eUICC Certificate	6.1	Verification Failed	eUICC Certificate or any certificate in the trust chain is invalid.

8.1.3	eUICC Certificate	6.3	Expired	eUICC Certificate or any certificate in the trust chain has expired.
8.1	eUICC	6.1	Verification Failed	eUICC signature is invalid or serverChallenge is invalid.
8.10.1	TransactionId	3.9	Unknown	The RSP session identified by the TransactionID is unknown.
8.11.1	CI Public Key	3.9	Unknown	Unknown eSIM CA RootCA Public Key. The eSIM CA is not a trusted root for the SM-DS, or not the root elected by the SM-DS in InitiateAuthentication response.
8.9.5	Event Record	3.9	Unknown	No Event identified by the Event ID for the EID exists.
8.1	eUICC	3.11	Value has Changed	#SupportedFromV3.0.0# eUICC RSP Capabilities have changed compared to those previously received.
8.12	LPA	3.11	Value has Changed	#SupportedFromV3.0.0# LPA RSP Capabilities have changed compared to those previously received.
8.9.x	Push Service	3.1	Unsupported	#SupportedForPushServiceV3.0.0# Push Service is not supported by the SM-DS.
8.10.x	Push Service Registration	3.1	Unsupported	#SupportedForPushServiceV3.0.0# Selected Push Service is not supported by the SM-DS
8.10.x	Push Service Registration	3.12	Invalid Match	#SupportedForPushServiceV3.0.0# The Selected Push Service and/or Push Token contained in Push Service Registration is different from those contained in MatchingID
1.6	Function	3.1	Unsupported	#SupportedFromV3.0.0# The requested function is not supported by the SM-DS.

Table 52: AuthenticateClient specific Status Codes

5.8.3 Function: CheckEvent

Related Procedures: Event Checking

Function Provider Entity: SM-DS

Description:

This function SHALL be called by the LPA to check the presence of pending Event Record(s) for the eUICC.

On reception of this function call, the SM-DS SHALL:

- Return an error status "Event Checking – Unsupported" if the SM-DS does not support Event Checking.
- Verify that the received address matches its own SM-DS address, where the comparison SHALL be case-insensitive. Otherwise, the SM-DS SHALL return a status code "SM-DS Address - Refused".

- Verify that the received ECID is valid. If the received ECID is known by the SM-DS but has been expired, the SM-DS SHALL return an error status "ECID – Expired". If the received ECID is unknown by the SM-DS, the SM-DS SHALL return an error status "ECID – Unknown".

NOTE: the lifetime of an ECID and its history is SM-DS implementation specific.

- Check if there exist any Event Record(s) corresponding to the received ECID.
- Set `isPendingEvent` to true if at least one Event Record corresponding to the received ECID exists. Otherwise, the SM-DS SHALL set it to false.

The SM-DS MAY perform additional operations, which are out of scope of this specification.

This function SHALL return one of the following:

- A 'Function execution status' with 'Executed-Success' indicating that the Event Checking has been successfully processed.
- A 'Function execution status' indicating 'Failed' with a status code as defined in section 5.2.6 or a specific status code as defined in the following table.

Additional Input Data:

Input data name	Description	Type	No.	MOC
`ecId`	EventCheckingID for the SM-DS	Binary[16-32]	1	M
`smdsAddress`	The SM-DS Address as known and provided by the LPA.	FQDN	1	M

Table 52a: CheckEvent Additional Input Data

Additional Output Data:

Output data name	Description	Type	No.	MOC
`isPendingEvent`	Indicates if an Event Record corresponding to the received ECID is pending	Boolean	1	M

Table 52b: CheckEvent Additional Output Data

Specific Status Codes

Subject code	Subject	Reason code	Reason	Description
8.10.5.	Event Checking	3.1	Unsupported	#SupportedForEventCheckingV3.0.0# Event Checking is not supported by the SM-DS.
8.9.6	ECID	6.4	Expired	#SupportedForEventCheckingV3.0.0# Expired ECID
8.9.6	ECID	3.9	Unknown	#SupportedForEventCheckingV3.0.0#

				Unknown ECID
8.9.1	SM-DS Address	3.8	Refused	Invalid SM-DS Address.

Table 52c: CheckEvent specific Status Codes

5.9 ES12 (SM-DP+ -- SM-DS)

The ES12 is used by the SM-DP+ to manage Event registration.

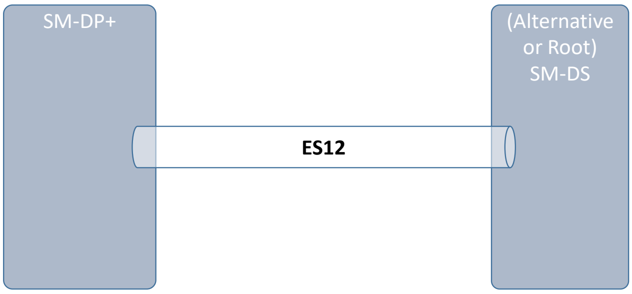


Figure 39: ES12

The SM-DP+ communicates with the SM-DS through a secure connection, by establishing a TLS connection with mutual authentication using CERT.DP.TLS and CERT.DS.TLS. Service level authorisation is required between SM-DS and SM-DP+. This is out of scope of this specification. Additional details about security requested on this interface and the level of data encryption are defined in section 2.6 and GSMA SAS SM specification [23].

5.9.1 Function: RegisterEvent

Related Procedures: Event Registration

Function Provider Entity: SM-DS (Alternative SM-DS or Root SM-DS)

Description:

This function registers an Event Record in the SM-DS (an Alternative SM-DS or a Root SM-DS).

The function caller MAY provide Hashed ICCID(s) and Salt in the input data. This information can be used by the LPA to identify the target Profile(s) of the Event Record. Hashed ICCID(s) SHALL be generated as follows, where a 19-digit ICCID SHALL be followed by the padding character 'F' for hashing, and the ICCID and Salt are encoded as UTF-8 strings.

- If Salt is not provided: SHA256(ICCID) for each target Profile.
- If Salt is provided: SHA256(ICCID|Salt) for each target Profile, where '|' means concatenation of data and Salt is randomly generated per function call. The same Salt SHALL be used to generate multiple hashes in one event.

The function caller MAY provide Service Provider Name and Operator ID in the input data. This information can be used by the LPA to identify the Service Provider and/or Operator of the Event Record.

The function caller MAY require that the Event registration is cascaded to a Root SM-DS using the input data 'forwardingIndicator'. If cascaded registration is requested, the function caller MAY also specify the Root SM-DS using the input data 'rootSmdsAddress'. If cascading is requested but the function caller did not specify the Root SM-DS, then the Root SM-DS used by the function provider is implementation-dependent.

A Root SM-DS SHALL ignore 'forwardingIndicator' and 'rootSmdsAddress'.

On reception of this function call, the SM-DS SHALL:

- Verify that the EventID is not already used by the function caller. Otherwise, the SM-DS SHALL return a status code "Event Record - Already in Use".
- Store the received Event Record, consisting of the EID, the RSP Server address, the EventID, the EventType, the HashedIccids (optional), the Salt (optional) the ServiceProviderName (optional), and the OperatorId (optional), all provided as function input data, together with the function caller identity (either an SM-DP+ or an SM-DS OID) received in the TLS Certificate. Additionally:
    - If EventType is not provided by the function caller, the SM-DS SHALL regard this as a function call from a version 2 SM-DP+ and SHALL use the value denoting 'Profile Download' in the Event Record.
- If required (forwardingIndicator input data set to 'true'), and if the function provider is an Alternative SM-DS
    - store the FQDN of the Root SM-DS with the Event Record for later use during Event Deletion.
- cascade the Event registration to the Root-SM-DS by calling the "ES15.RegisterEvent" with the relevant input data:
    - Same EID value.
    - Its own SM-DS address to be used for Event retrieval.
    - Its own generated Event ID corresponding to this incoming registration Event.
    - Same EventType value.
    - Same HashedIccids, Salt, ServiceProviderName, and OperatorId values (if any).
    - forwardingIndicator input data set to 'false'
- If registration cascading fails the SM-DS SHALL delete the locally stored Event Record and return a status code "SM-DS - Inaccessible" or "SM-DS - Execution error".

The SM-DS MAY perform additional verifications and operations, which are out of scope of this specification.

This function SHALL return one of the following:

- A 'Function execution status' with 'Executed-Success' indicating that the Event has been registered (and cascaded if required).

- A 'Function execution status' indicating 'Failed' with a status code as defined in section 5.2.6 or a specific status code as defined in the following table.

Additional Input Data:

Input data name	Description	Type	No.	MOC
`eid`	Identification of the targeted eUICC.	EID	1	M
`rspServerAddress`	RSP Server address where the operation corresponding to the registered Event can be executed.	FQDN	1	M
`eventId`	Identification of the Event. The EventID SHALL be unique in the context of the function caller.	String	1	M
`forwardingIndicator`	Indicates if the registration has to be made to the Root SM-DS. The Root SM-DS SHALL ignore this input data.	Boolean	1	M
`rootSmdsAddress`	Indicates the Root SM-DS to be used when `forwardingIndicator` is TRUE.	FQDN	1	C(4)
`eventType`	Type of the RSP Operation associated with the Event. This SHALL be one of the following values: 1 -> Profile Download 2 -> RPM	INTEGER	1	C(1)
`hashedIccids`	Identification of the target Profile(s).	Binary[32]	1..N	O
`salt`	Random string to be concatenated with ICCID(s) for hashing.	Binary[8-16]	1	C(2)
`serviceProviderName`	Name of the Service Provider that requested the RSP Operation.	String	1	O
`operatorId`	Identification of the Operator that requested the RSP Operation.	Binary(3)	1	O
NOTE 1: This field SHALL be provided if SM-DP+ is v3. NOTE 2: This field MAY be provided if and only if hashedIccids is present. NOTE 3: This field SHALL use the ASN.1 format defined in section 2.4a.1.2. NOTE 4: This field MAY be provided if and only if forwardingIndicator is TRUE.				

Table 53: RegisterEvent Additional Input Data

Additional Output Data:

No additional output data.

Specific Status Codes

Subject Code	Subject	Reason code	Reason	Description
8.9.5	Event record	3.3	Already in use	The Event Record already exist in the SM-DS (EventID duplicated).
8.9	SM-DS	5.1	Inaccessible	The cascade SM-DS registration has failed. Root SM-DS was unavailable.
8.9	SM-DS	4.2	Execution error	The cascade SM-DS registration has failed. Root SM-DS has raised an error.

Table 54: RegisterEvent specific status codes

5.9.2 Function: DeleteEvent

Related Procedures: Event Deletion

Function Provider Entity: SM-DS

Description:

This function deletes an Event Record in the SM-DS (an Alternative SM-DS or a Root SM-DS).

On reception of this function call, the SM-DS SHALL:

- Retrieve the Event Record corresponding to the provided Event ID and the function caller identity (SM-DP+ or SM-DS OID) received in the TLS Certificate. If the Event Record doesn't exist the SM-DS SHALL return a status code "Event Record - Unknown".
- Determine if the Event Record registration has been cascaded to Root SM-DS(s).
- If the Event Record registration was not cascaded, then the SM-DS SHALL delete the retrieved Event Record.
- If the Event Record registration was cascaded, then
    - the SM-DS SHALL retrieve the FQDN of the Root SM-DS that was stored with the Event Record.
    - the SM-DS SHALL cascade the deletion of the Event Record to the Root SM-DS by calling the "ES15.DeleteEvent" with the Event ID value generated by the SM-DS during Event Record registration.
    - If deletion at the Root SM-DS fails:
        - Because the Event record was not found (the ES15.DeleteEvent call has returned a status code "Event Record - Unknown"), the SM-DS SHALL ignore this error case and consider the deletion at the Root SM-DS has succeeded.
        - For any other reason, the SM-DS SHALL return a status code "SM-DS - Inaccessible" or "SM-DS - Execution error".
    - If deletion at the Root SM-DS has succeeded, the SM-DS SHALL delete the retrieved Event Record.

The SM-DS MAY perform additional verifications and operations, which are out of scope of this specification.

This function SHALL return one of the following:

- A 'Function execution status' with 'Executed-Success' indicating that the Event has been deleted (and cascaded if required).
- A 'Function execution status' indicating 'Failed' with a status code as defined in section 5.2.6 or a specific status code as defined in the following table.

<u>Additional Input Data:</u>

Input data name	Description	Type	No.	MOC
eid	Identification of the targeted eUICC.	EID	1	M
eventId	Identification of the Event. The EventID SHALL be unique in the context of the function caller.	String	1	M

Table 55: DeleteEvent Additional Input Data

<u>Additional Output Data:</u>

No additional output data

<u>Specific Status Codes</u>

Subject Code	Subject	Reason code	Reason	Description
8.9.5	Event Record	3.9	Unknown	The targeted Event Record doesn't exist.
8.9	SM-DS	5.1	Inaccessible	The cascade SM-DS deletion has failed. Root SM-DS was unavailable.
8.9	SM-DS	4.2	Execution error	The cascade SM-DS deletion has failed. Root SM-DS has raised an error.

Table 56: DeleteEvent specific Status Codes

5.10 ES15 (SM-DS -- SM-DS)

This interface is a particular case of the ES12 interface where an Alternative SM-DS is communicating to the Root SM-DS to manage the cascading of Events. This interface is functionally identical to ES12.

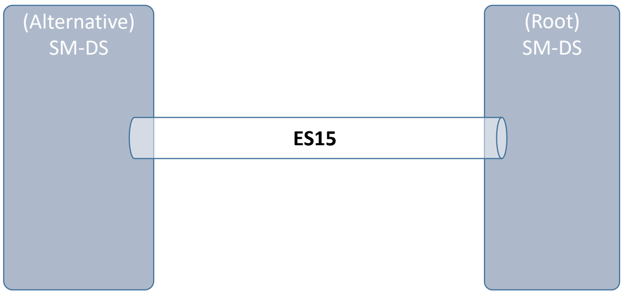


Figure 40: ES15

The Alternative SM-DS and the Root SM-DS communicate through a secure connection, by establishing a TLS connection with mutual authentication using their CERT.DS.TLS. Service level authorisation is required between both SM-DSs. This is out of scope of this specification. Additional details about security requested on this interface and the level of data encryption are defined in section 2.6 and GSMA SAS SM specification [23].

5.10.1 Function: RegisterEvent

Related Procedures: Event Registration

Function Provider Entity: (Root) SM-DS

Description:

This function is identical to "ES12.RegisterEvent".

5.10.2 Function: DeleteEvent

Related Procedures: Event Deletion

Function Provider Entity: (Root) SM-DS

Description:

This function is identical to "ES12.DeleteEvent".

5.11 ES25 (UIMe -- LUIe)

The implementation of the LUIe uses ES25 between the LPAe and the User Interface Module for the LPAe (UIMe) in the device.

For LUIe using CAT, the UIMe is the CAT interpreter, which is not RSP specific.

For LUIe using SCWS, the UIMe is the browser with its interface to the SCWS, which is not RSP specific.

For LUIe using E4E (ENVELOPE commands with tag 'E4'), the UIMe is implementation specific. It SHALL implement all requirements for the LPA related to the user interface.

A Device supporting LUIe SHALL support at least one of the mechanisms defined in this section.

NOTE 1: DeviceCapabilities MAY be empty if provided by the LPAe.

NOTE 2: Support of MEP on ES25 is FFS.

5.11.1 LUIe using CAT

In order to support this option, the Device SHALL support at least the CAT mechanisms defined in Annex C.4.

5.11.2 LUIe using SCWS

In order to support this option, the Device SHALL support at least the CAT mechanisms defined in Annex C.4.

In addition, the eUICC and the Device SHALL support the Smartcard Web Server as defined in [7].

When using High Resolution Icons, it is recommended to embed the URLs to retrieve the HRIs from the HRI server (see section 5.12.1) into the web pages created by the SCWS. This avoids passing the (large) HRI data through the eUICC.

5.11.3 LUIe using E4E

In order to support this option, the Device SHALL support at least the CAT mechanisms defined in Annex C.4.

The following data structure SHALL be sent by the device to the eUICC in an ENVELOPE command:

```
-- ASN1START
E4ERequest ::= [PRIVATE 4] CHOICE { -- Tag 'E4'
   startDownload [0] SEQUENCE {
      activationCode [0] UTF8String (SIZE(0..255))
   }, -- Start Download
   confirmDownload [1] SEQUENCE {
      enable [0] NULL OPTIONAL, -- enable Profile after download
      confirmationCode [1] UTF8String OPTIONAL, -- confirmation code
      pinCode [2] UTF8String (SIZE(4..8)) OPTIONAL -- LPAe PIN if used
   }, -- Confirm Download
   listProfiles [2] NULL, -- List Profiles
   enableProfile [3] SEQUENCE {iccid [APPLICATION 26] Iccid}, -- Enable Profile
   disableProfile [4] SEQUENCE {iccid [APPLICATION 26] Iccid},  -- Disable Profile
   deleteProfile [5] SEQUENCE {
      iccid [APPLICATION 26] Iccid,
      pinCode [1] UTF8String (SIZE(4..8)) OPTIONAL -- LPAe PIN if used
   }, -- Delete Profile
   euiccMemReset [6] SEQUENCE {
      pinCode [0] UTF8String (SIZE(4..8)) OPTIONAL -- LPAe PIN if used
   }, -- eUICC Memory Reset
   changeConfirmationPin [7] UTF8String (SIZE(9..17)), -- Change confirmation PIN
   setRpmAllow [8] BOOLEAN, -- Turn on/off Remote Profile Management
   pollRpmPackage [9] SEQUENCE {iccid [APPLICATION 26] Iccid OPTIONAL},
   -- Check for RPM packages for profile with iccid. No iccid means 'Update All'
   confirmRpmPackage [10] SEQUENCE {
      pinCode [0] UTF8String (SIZE(4..8)) OPTIONAL
      -- LPAe PIN, if used, with Strong Confirmation
   }, -- Confirms the pending RpmPackage
   cancelSession[11] NULL
   -- Cancels the pending profile download or RpmPackage execution
}
-- ASN1STOP
```

For `startDownload`, the UTF8 string, if non-zero length, SHALL contain the Activation Code as defined in section 4.1. A UTF8 string of zero length SHALL indicate to the eUICC to check for a new Profile at the Default SM-DP+ (if configured) and the SM-DS.

Even though the ASN.1 definition would allow larger data, each request and response SHALL be limited so that it can be included in a single APDU.

For `changeConfirmationPin`, the UTF8 string SHALL contain the old PIN followed by the new PIN, separated by a semicolon (which is not allowed as PIN character).

Operation pollRpmPackage starts the 'Update Profile' procedure as defined in section 3.2.7. The value of pollRpmPackage makes LPAe to behave as follows:

- 'Update' single Profile if the value includes the ICCID of the Profile. Error errorProfileDoesNotExist is returned if there is not an installed Profile with that ICCID.
- 'Update All' Profiles, if no ICCID is provided.

Operation cancelSession instructs the LPAe to perform the Common Cancel Procedure as defined in section 3.0.2, with the reason 'endUserRejection'.

NOTE: ENVELOPE commands with tag 'E4' (which is reserved by ETSI for GSMA) are also used in SGP.02 [2], starting from version 4.0. However, the first byte of the value part of the TLV is limited to '00' to '03', allowing easy discrimination from the values defined here.

Depending on the function contained in the ENVELOPE command, the UIMe SHALL return a response structure to the UIMe, either directly in the response to the ENVELOPE command, or via a subsequent REFRESH proactive command to the Device with the following parameters:

- Refresh mode SHALL be set to "Application Update".
- AID SHALL contain the AID of the ISD-R.
- Refresh enforcement policy SHALL be absent.
- The value field of the application specific refresh data SHALL contain the E4EResponse object defined below.

In case of the REFRESH command in "Application Update" mode is not supported by the Device, the E4EResponse object is stored in a $EF_{AS\text{-}RD}$ (Application Specific – Response Data) located in the DF_{GSMA} (which resides under the MF and has file ID '7F26' as specified in ETSI TS 102 221 [6]), and a REFRESH proactive command is sent to the Device with the following parameters:

- Refresh mode SHALL be set to "NAA File Change Notification".
- AID SHALL contain the AID of the ISD-R.
- Refresh enforcement policy SHALL be absent.
- The value field of the file list SHALL contain a reference to $EF_{AS\text{-}RD}$.

In the case of a profile state change after the E4E Response, a REFRESH in "UICC Reset" or "eUICC Profile State Change" mode is sent.

Table 56a below defines the correspondence between the functions and related responses, and how the responses SHALL be conveyed to the UIMe. Figure 40aa below specifies the flow visible from the UIMe for a Profile download.

Function contained in the ENVELOPE command	Data in the response to the ENVELOPE command	Response data sent in a subsequent proactive command
startDownload	None	startDownloadResponse
confirmDownload	None	confirmDownloadResponse
listProfiles	None	listProfilesResponse
enableProfile	None	None
disableProfile	None	None
deleteProfile	None	None
euiccMemReset	None	None
changeConfirmationPin	None	None
setRpmAllow	None	None
pollRpmPackage	None	pollRpmPackageResponse
confirmRpmPackage	None	None
cancelSession	None	None

Table 56a: Functions and their responses for E4E

Responses `startDownloadResponse` and `pollRpmPackageResponse` requires the User to confirm them using `confirmDownload` and `confirmRpmPackage,` respectively.

The structure of the response data is defined below:

```
-- ASN1START
E4EResponse ::= [PRIVATE 4] SEQUENCE { -- Tag 'E4'
   resultCode [0] E4EResultCode,
   resultData [1] CHOICE {
      startDownloadResponse [0] SEQUENCE {
         serviceProviderName [17] UTF8String (SIZE(0..32)), -- Tag '91'
         profileName [18] UTF8String (SIZE(0..64)), -- Tag '92'
         ccRequired [0] NULL OPTIONAL -- confirmation code required
      },
      listProfilesResponse [3] SEQUENCE OF SEQUENCE {
         iccid [APPLICATION 26] Iccid, -- Profile ICCID
         profileState [112] ProfileState, -- Tag '9F70'
         serviceProviderName [17] UTF8String (SIZE(0..32)), -- Tag '91'
         profileName [18] UTF8String (SIZE(0..64)) -- Tag '92'
         -- the eUICC MAY truncate these names so that the response fits
         -- into one APDU
      },
      pollRpmPackageResponse [4] SEQUENCE {
         rpmPackage [0] RpmPackage, -- RPM Package to be confirmed by user
         rpmPending [1] NULL OPTIONAL -- There are pending RPM Packages after this
      },
      confirmDownloadResponse [5] SEQUENCE {
         iccid [APPLICATION 26] Iccid -- Profile ICCID
      }
   } OPTIONAL
}

E4EResultCode ::= INTEGER {
   success (0),
   errorBusy (1), -- CAT not available due to another operation
   errorComm (2), -- Communication error with server
   errorAuth (3), -- Mutual Authentication Error
   errorNoProfile (4), -- No Profile available for download at SM-DP+
   errorEligibility (5), -- SM-DP+ rejected download due to Eligibility Check
   errorInstall (6), -- Error during Profile installation
   errorPin (7), -- Invalid PIN
   errorProfileRef (8), -- Referenced Profile does not exist
   errorAlreadyEnabled (9), -- Referenced Profile is already enabled
   errorAlreadyDisabled (10), -- Referenced Profile is already disabled
   errorConfirmationCode (11), -- Invalid Confirmation Code,
   errorRpmDisabled (12), -- Cannot pollRpmPackage, RPM is disabled
   errorProfileDoesNotExist (13), -- There is no profile with provided ICCID
   undefinedError (127)
}

-- ASN1STOP
```

NOTE: If there is no pending RPM command, rpmPackage data element in pollRpmPackageResponse is a zero-length sequence.

The flow for a profile download is defined below:

```
@startuml
hide footbox
skinparam sequenceMessageAlign center
skinparam sequenceArrowFontSize 11
skinparam noteFontSize 11
skinparam monochrome true
skinparam lifelinestrategy solid


participant "<b>End-User" as U
participant "<b>LPAe" as LPA
participant "<b>Device" as D
participant "<b>SM-DP+" as DP

U-->D: [Trigger addition of Profile]

D-->LPA: E4E with startDownload

rnote over LPA #FFFFFF
    (a) Get SM-DP+ Address, Parse Activation Code Token, [SM-DP+ OID], [CI PK ind.]
from AC, or
    (b) Get SM-DP+ Address and EventID from SM-DS, or
    (c) Get Default SM-DP+ Address, [CI PKID] from eUICC, or
End rnote

LPA-->D: Acnowledge E4E
LPA-->D: Open BIP Channel
LPA-->DP: [Contact server, Common Mutual Authenitcation, retrieve Profile metadata]

LPA-->D: Alerting with startDownloadResponse
D-->U: [Present content to user]
U-->D: [Optional confirmation code and/or LPAe PIN]
D-->LPA: E4E with confirmDownload
LPA-->D: Acnowledge E4E

LPA-->DP: [Retrieve BPP]
LPA-->LPA: [install Profile]

LPA-->D: Alerting with confirmDownloadResponse
D-->U: [Present result to user]
alt enable profile
    LPA-->D: REFRESH "UICC Reset" or "eUICC Status Change"
end
@enduml
```

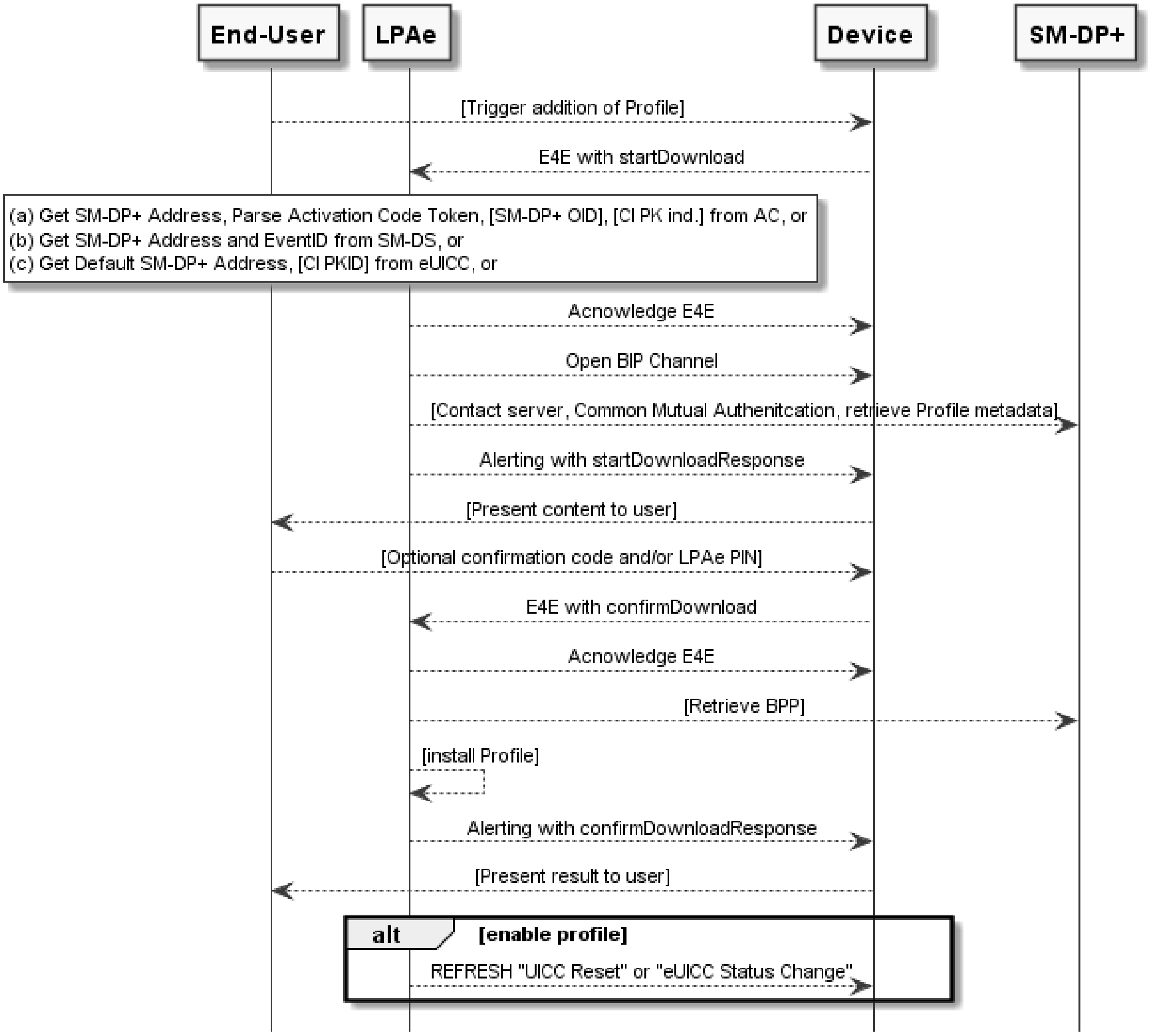

End-User
LPAe
Device
SM-DP+
[Trigger addition of Profile]
E4E with startDownload
(a) Get SM-DP+ Address, Parse Activation Code Token, [SM-DP+ OID], [CI PK ind.] from AC, or
(b) Get SM-DP+ Address and EventID from SM-DS, or
(c) Get Default SM-DP+ Address, [CI PKID] from eUICC, or
Acnowledge E4E
Open BIP Channel
[Contact server, Common Mutual Authenitcation, retrieve Profile metadata]
Alerting with startDownloadResponse
[Present content to user]
[Optional confirmation code and/or LPAe PIN]
E4E with confirmDownload
Acnowledge E4E
[Retrieve BPP]
[install Profile]
Alerting with confirmDownloadResponse
[Present result to user]
alt
[enable profile]
REFRESH "UICC Reset" or "eUICC Status Change"

Figure 40aa: Profile download using E4E

After a `startDownload`, the eUICC SHALL return the `serviceProviderName` and the `profileName` from the Metadata of the Profile to download. Thereafter, the eUICC SHALL be informed with `confirmDownload` if the End User accepts the download. After Bound Profile Package download and install process is performed, the eUICC SHALL inform the status of the download with `confirmDownloadResponse`.

The `listProfilesResponse` SHALL contain a list of all Profiles on the eUICC. `enableProfile`, `disableProfile`, and `deleteProfile` SHALL reference a Profile using its ICCID as returned by `listProfilesResponse`.

5.12 Other functions

5.12.1 Function: RetrieveIcon

Related Procedures: may be used in different procedures.

Function Provider Entity: HRI Server

Description:

This function is used to retrieve an icon related to the Profile from the HRI Server. This function MAY be used in different procedures.

The HRI Server MAY be operated by any entity, e.g., the SM-DP+, the Operator or any other party. If it is operated by the SM-DP+, the server infrastructure of the SM-DP+ SHALL allow the same TLS connection that is established for any other RSP procedure to be used for retrieving one or several icons.

On reception of this function call, the HRI Server SHALL:

- If required, customise the icon data (resize, crop, etc.) so that the resulting icon matches the request parameters.
- In the response send the icon to the LPA.

The HRI Server MAY perform additional operations, which are out of scope of this specification.

This function SHALL use an HTTP GET request instead of the function binding defined in sections 6.5 and 6.6.

The URL of the request SHALL be formatted as follows:

https://<iconLocation>?w=<width>&h=<height>&f=<format>

<iconLocation> is the string provided in the Metadata of the Profile. The other parameters SHALL be added by the LPA.

<width> is the width of the icon in pixels. Permitted range: 64 to 1024.

<height> is the height of the icon in pixels. Permitted range: 64 to 1536.

Permitted range of aspects ratios (height / width): 1.5 to 0.25.

<format> is the icon format. Permitted values: "jpg", "png".

Example for a valid request:
https://server.domain.com/mno-icon?w=320&h=240&f=jpg

The HRI Server SHALL return an icon in the requested format in the HTTP response.

See section 5.11.2 for handling by the LPAe.

5.13 ES22 (Device Application -- LPDd)

The ES22 is an interface defined between the Device Application and the LPDd.

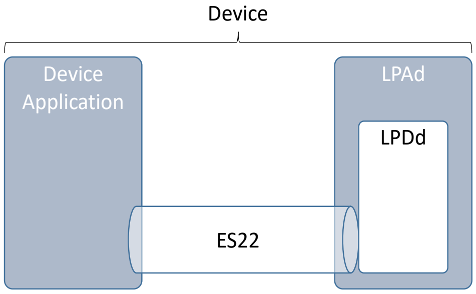


Figure 40a: ES22

The access to the LPDd via the ES22 interface SHALL be authorised by a Device manufacturer implementation-specific access control mechanism.

The coding and function binding of ES22 functions are Device manufacturer implementation-specific.

5.14 ES21 (Device Application -- LPRd)

ES21 is an interface defined between the Device Application and the LPRd, which initiates Profile Content Management sessions and optionally relays progress information from the PCMAA.

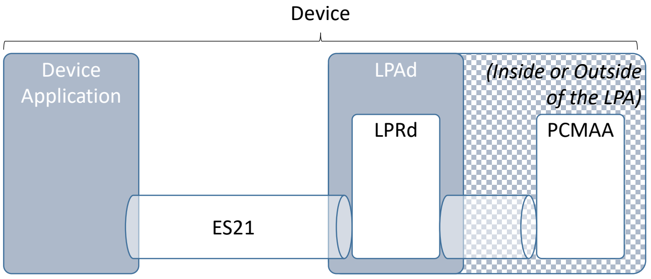


Figure 40b: ES21

The Device SHALL provide an implementation-specific mechanism by which the Device Application can determine whether the LPA Proxy feature is supported on the Device and the eUICC.

The specific function binding for the Command, Response, and Notification Data on the ES21 interface is Device manufacturer-specific.

5.14.1 Function: InitiateProfileContentManagement

Related Procedures: Profile Content Management

Function Provider Entity: LPRd

Description:

This function is used by the Device Application to trigger a Profile Content Management session, optionally with a specified DPI parameter, and to request progress information from the PCM session.

On reception of this command, the LPRd SHALL:

- Verify that there is an enabled Profile; otherwise, return the error noEnabledProfile.
- Retrieve the Profile Metadata for the currently enabled Profile by calling the "ES10c.GetProfilesInfo" function.
- Verify that the Device Application is authorised to trigger a PCM session on the enabled Profile; otherwise, return the error deviceApplicationNotAuthorised. The mechanism by which this authorisation is performed is Device manufacturer-specific.
- Verify that the Profile Metadata contains an lprConfiguration with a non-empty PCMP address; otherwise, return the error noLprConfiguration.
- Verify that either the End User has not disallowed mobile network data to be used for the LPA Proxy/PCM session and mobile network data is available, or that some other data connectivity is available. If data connectivity is unavailable, return the error dataConnectivityNotAvailable.
- If the Device Application has requested progress information, initialise the Device manufacturer-specific mechanism by which it is delivered. (For example, this could register a callback function, create a message channel, etc.).
- Calculate the initial URI by concatenating the PCMP address from the Metadata of the enabled Profile, and the DPI provided by the Device Application, if available.
- Trigger the PCMAA with the initial URI.
- Return ok (success) to the Device Application.

Command Data

Input data name	Description	Type	No.	MOC
dpi	Delegated Platform Identifier to be used for the PCM session.	UTF-8 String	1	O
progressRequest	Indicates that the Device application wishes to receive PCM progress information.	Boolean	1	O

Response Data

Output data name	Description	Type	No.	MOC
status	Status of the requested operation	Integer	1	M

Status Values

Status	Description
ok	Success (PCM session was triggered)
noEnabledProfile	No Profile is enabled

`deviceApplicationNotAuthorised`	Device Application is not authorised to trigger PCM on the enabled Profile
`noLprConfiguration`	No `LprConfiguration` or empty PCMP address in the Metadata of the enabled Profile
`dataConnectivityNotAvailable`	The PCM session cannot be triggered because there is no data connection

5.14.2 Notification: PcmProgressInformation

Related Procedures: Profile Content Management

Function Provider Entity: LPRd

Description:

A Device Application that triggers a Profile Content Management session MAY request progress information from that session. The method by which this information is delivered is Device manufacturer-specific. There are several notification types:

- A progress message delivered from the Profile Content Management Platform (server), containing binary data whose format and content is Profile Owner-specific.
- An indication of a script part delivery to the UICC, optionally along with the set of UICC status words for each processed APDU.
- An indication that an HTTP request has been sent to the PCMP, including the targeted URI and conditionally the ending status of a delegated PCM dialog.
- An indication that the PCM session has ended and its ending status.

The progress information MAY include additional content that is Device manufacturer-specific.

<u>Notification Data</u>

The progress information SHALL indicate the notification type and contain at least the following information based upon the type, as described hereunder. It MAY include additional content that is Device manufacturer-specific.

Progress Message:

Data name	Description	Type	No.	MOC
`progressMessage`	Progress information message from the PCMP	Binary	1	M

Script Part Delivery:

Data name	Description	Type	No.	MOC
`partIndex`	Index of the executed script part	INTEGER	1	M
`scriptStatus`	The status of the script delivery to the eUICC, containing one of the values defined for `X-Admin-Script-Status` in [74] section 4.3.5.7	String	1	M
`apduStatus`	The status words from each processed APDU in a script part. This MAY not be present if there are no status words (e.g., if an error prevented the script from being processed).	Binary	1	C

HTTP Request to PCMP:

Data name	Description	Type	No.	MOC
`uri`	Targeted URI	String	1	M
`dialogEndingStatus`	The last dialog status, containing one of the values defined for `X-Admin-Dialog-Status` in [74] section 4.3.5.11. This is provided when the HTTP request immediately follows the end of a dialog with each PCMP.	String	1	C

HTTP Response From PCMP:

Data name	Description	Type	No.	MOC
`partsCount`	The number of script parts received from the PCMP in the HTTP response.	INTEGER	1	M
`dialogEndingStatus`	The ending status of the dialog with the PCMP, containing one of the values defined for `X-Admin-Dialog-Status` in [74] section 4.3.5.11. This is provided for the final HTTP response in a PCMP dialog.	String	1	C

End of PCM Session:

Data name	Description	Type	No.	MOC
`dialogEndingStatus`	The ending status of the PCM session, containing one of the values defined for `X-Admin-Dialog-Status` in [74] section 4.3.5.11.	String	1	M

6 Interface binding over HTTP

This section defines how to use HTTP/1.1, defined in RFCs 7230 [81] and 7231 [82], and TLS, defined in RFC 5246 [16], as the transport layer to exchange ES2+, ES9+, ES11, ES12, and ES15 function requests and responses.

On ES9+ and ES11, the LPA always acts as an HTTP client and is in charge of managing the connection establishment to the RSP Server. The LPA SHALL use either JSON binding defined in section 6.5 or ASN.1 binding defined in section 6.6. The LPAe SHALL exchange the HTTPS POST requests and responses defined in this section using BIP over TCP.

On ES2+, ES12, and ES15 any RSP Server MAY act as an HTTP client or an HTTP server. JSON binding defined in section 6.5 SHALL be used.

In case of communication failure, the HTTP client is responsible for retry and reconnection management.

6.1 TLS Security

Transport Layer Security (TLS) secures the messages exchanged between a function requester and function provider. Refer to section 2.6.6 for TLS version and security details that the HTTP Client and the HTTP Server SHALL follow.

In case a procedure defined in section 3 is restarted, a new TLS session SHALL be established. In case of failure within a procedure, the TLS session MAY be resumed or re-established.

6.1.1 Identification/Authentication/Authorisation

If applicable on the interface, authentication of the sending party of a JSON message SHALL rely on the Transport layer security (using TLS Certificate of the sending party).

6.1.2 Integrity

The integrity of the message SHALL exclusively rely on the Transport Layer Security (TLS).

6.1.3 Confidentiality

The confidentiality of the message SHALL exclusively rely on the Transport Layer Security (TLS).

6.2 HTTP request and response

An HTTP POST request SHALL be used to transport a single function execution request. The corresponding function execution response SHALL be returned as defined in SGP.02 [02] depending on the used Message eXchange Pattern (MXP).

This specification uses the following MXPs:

- Synchronous Request-Response: the request payload SHALL be sent in the HTTP POST request, and the function execution response SHALL be returned in the HTTP POST response.
- Notification: the notification payload SHALL be sent in the HTTP POST request and the HTTP POST response body SHALL be empty.

NOTE: Following common practice in the Internet, Devices typically convert the FQDN contained in the Activation Code to lowercase when providing it in the SNI (Server Name Indication) extension of TLS and in the "`Host`" header field of the HTTP POST request. However, the Devices are not mandated to perform this conversion. Therefore, an issue may occur if the SM-DP+ or SM-DS does not perform a case-insensitive comparison.

HTTP POST request for ES9+ and ES11 SHALL contain a "`User-Agent`" header field defined according to RFC 7231 [82] as either of the following:

```
User-Agent: <product>
User-Agent: <product> (<comment>)
```

where *`<product>`* is either `gsma-rsp-lpad` or `gsma-rsp-lpae`. Additional information MAY be included in a comment delimited by parentheses as defined by RFC 7230 [81].

Alternatively, the "`User-Agent`" field MAY contain additional information after a semicolon, as defined in version 2 of this specification. An SM-XX compliant to this version of this specification SHALL accept this format. However, the LPA SHOULD NOT use this format; instead it SHOULD provide any additional information within an RFC 7230 [81] compliant comment.

NOTE: The use of a semicolon to indicate additional information is not compliant with RFC 7231 [82] and its support may be removed in a future version of this specification.

HTTP POST request and response SHALL contain an "`X-Admin-Protocol`" header field as defined hereunder:

```
X-Admin-Protocol: gsma/rsp/v<x.y.z>
```

Where:
<x.y.z> indicates a SVN of SGP.22 [This document].

On ES2+, ES12 and ES15, this indicates the highest SVN supported by the RSP Server acting as the function requester. The RSP Server acting as the function provider SHALL return an HTTP response as defined in the SGP.22 [This document] version corresponding to the lower of the function requester SVN and the function provider SVN.

On ES9+ and ES11, this field is deprecated and SHALL be set to v2.1.0 in both HTTP request and HTTP response.

NOTE: this value is required for interoperability with version 2 of this specification.

HTTP POST request and response SHALL contain a "`Content-type`" header field to indicate the nature of the binding. A JSON binding SHALL be indicated by the value "`application/json;charset=UTF-8`", which also mandates UTF-8 encoding. An ASN-1 binding SHALL be indicated by the value "`application/x-gsma-rsp-asn1`". The "`Content-type`" header field of an HTTP response SHOULD NOT be set when the body is empty (e.g., case of notification function response). If present, it SHALL be ignored.

NOTE: In version 2, the JSON encoding was not specified (see also Annex N).

HTTP POST request and response MAY contain additional header fields. Their use is out of scope of this specification.

6.3 HTTP response status codes

Standard HTTP status codes SHALL apply to this section.

Status codes '1xx' (Information), '3xx' (Redirection), '4xx' (HTTP client error) and '5xx' (HTTP server error) MAY be used by the RSP Server (i.e., the HTTP server).

The retry policy for HTTP request answered with status codes '4xx' and '5xx' is out of scope of this specification.

A normal request-response function execution status (MXP Synchronous request-response) SHALL be indicated by the HTTP status code '200' (OK) in the HTTP response, regardless whether the function response is an error or a success, as defined in SGP.02 [02].

A normal notification function execution status (MXP Notification) SHALL be indicated by the HTTP status code '204' (No Content) with an empty HTTP response body as defined in SGP.02 [02].

Other status codes '2xx' SHALL NOT be used by the RSP Server.

6.4 Secure Channel Set-Up on ES2+

The process of setting up secure channel is out of scope of this document. This process includes the exchange of the following information:

- Function requester and Function provider OIDs and identity SHALL be registered to GSMA Policy Authority and respective values have been communicated to each party.
- Function requester and Function provider URL SHALL have been communicated to each party.
- Function requester and Function provider parties’ trust SHALL have been established on an X-509 certificate chain basis.

6.5 Function Binding in JSON

JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is based on a subset of the JavaScript Programming Language. JSON is a text format that is completely language independent.

Only the characters that are mandatory to escape (Quotation mark, reverse solidus, and the control characters) SHALL be escaped in a JSON string value as specified in RFC 7159 [68] section 7. Other characters SHALL NOT be escaped.

6.5.1 JSON message definition

The Function Requester and the Function Provider SHALL exchange the JSON objects in HTTP messages as follows.

- HTTP Request SHALL have the following format.

```
HTTP POST <HTTP Path> HTTP/1.1
Host: <Server Address>
User-Agent: <User Agent>
X-Admin-Protocol: gsma/rsp/v<x.y.z>
```

```
Content-Type: application/json;charset=UTF-8
Content-Length: <Length of the JSON requestMessage>

<JSON requestMessage>
```

The <HTTP Path> is used to indicate which function execution is requested by the HTTP client. The list of defined <HTTP Path> are described in section 6.5.2.

- HTTP Response SHALL have the following format.

```
HTTP/1.1 <HTTP Status Code>
X-Admin-Protocol: gsma/rsp/v<x.y.z>
Content-Type: application/json;charset=UTF-8
Content-Length: <Length of the JSON responseMessage>

<JSON responseMessage>
```

6.5.1.1 Definition of <JSON requestMessage>

<JSON requestMessage> is the combination of:

- <JSON requestHeader>
- <JSON body> which depends on the function called

HTTP messages for ES9+ and ES11 SHALL NOT contain the <JSON requestHeader>.

6.5.1.2 Definition of <JSON responseMessage>

<JSON responseMessage> is the combination of:

- <JSON responseHeader>
- <JSON body> which depends on the function called

The HTTP POST response body SHALL be empty for MXP notification message (see section 6.3).

6.5.1.3 Definition of <JSON requestHeader>

The <JSON requestHeader> maps the function input header.

```
{
  "header": {
    "type": "object",
    "properties": {
      "functionRequesterIdentifier": {
        "type": "string",
        "description": "identification of the function requester, or the entity
on behalf of which the function requester operates"
      },
      "functionCallIdentifier": {
        "type": "string",
        "description": "identification of the function call"
      }
    },
    "required": ["functionRequesterIdentifier", "functionCallIdentifier"]
  }
}
```

6.5.1.4 Definition of <JSON responseHeader>

The <JSON responseHeader> maps the function output header.

```
{
   "header": {
      "type": "object",
      "properties": {
         "functionExecutionStatus": {
            "type": "object",
            "description": "Whether the function has been processed correctly or
not",
            "properties": {
               "status": {
                  "type": "string",
                  "description": " Executed-Success, Failed"
               },
               "statusCodeData": {
                  "type": "object",
                  "properties": {
                     "subjectCode": {
                        "type": "string",
                        "description": "OID of the subject code"
                     },
                     "reasonCode": {
                        "type": "string",
                        "description": "OID of the reason code"
                     },
                     "subjectIdentifier": {
                        "type": "string",
                        "description": "Identifier of the subject "
                     },
                     "message": {
                        "type": "string",
                        "description": "Textual and human readable explanation"
                     }
                  },
                  "required": ["subjectCode", "reasonCode"]
               }
            },
            "required": ["status"]
         }
      },
      "required": ["functionExecutionStatus"]
   }
}
```

6.5.1.5 Details on JSON types

`"format": "base64"`: unless specified otherwise below, the value of a JSON field of this format SHALL contain the base64 coding defined in RFC 4648 [71] of the DER encoded ASN.1 data object (including its tag and length fields), referenced in `"description"`.

NOTE In most of the cases, the ASN.1 data object is defined in ES10x request/responses. Otherwise, the 'description' of the base64 field references the section where the ASN.1 type is specified.

`"pattern": "^[0-9,A-F]{n,m}$"`: specifies the hexadecimal representation of the data referred to in `"description"`.

6.5.2 List of functions

	Function	Path	MXP
ES2+	DownloadOrder	/gsma/rsp2/es2plus/downloadOrder	Synchronous
	ConfirmOrder	/gsma/rsp2/es2plus/confirmOrder	Synchronous
	CancelOrder	/gsma/rsp2/es2plus/cancelOrder	Synchronous
	ReleaseProfile	/gsma/rsp2/es2plus/releaseProfile	Synchronous
	RpmOrder	/gsma/rsp3/es2plus/rpmOrder	Synchronous
	ConfirmDeviceChange	/gsma/rsp3/es2plus/confirmDeviceChange	Synchronous
	HandleNotification	/gsma/rsp3/es2plus/handleNotification	Notification
ES9+	InitiateAuthentication	/gsma/rsp2/es9plus/initiateAuthentication	Synchronous
	AuthenticateClient	/gsma/rsp2/es9plus/authenticateClient	Synchronous
	GetBoundProfilePackage	/gsma/rsp2/es9plus/getBoundProfilePackage	Synchronous
	HandleNotification	/gsma/rsp2/es9plus/handleNotification	Notification
	HandleDeviceChangeRequest	/gsma/rsp3/es9plus/handleDeviceChangeRequest	Synchronous
	CancelSession	/gsma/rsp2/es9plus/cancelSession	Synchronous
ES11	InitiateAuthentication	[As ES9+]	[As ES9+]
	AuthenticateClient	[As ES9+]	[As ES9+]
ES11	InitiateAuthentication	[As ES9+]	[As ES9+]
ES12	RegisterEvent	/gsma/rsp2/es12/registerEvent	Synchronous
	DeleteEvent	/gsma/rsp2/es12/deleteEvent	Synchronous
ES15	RegisterEvent	[As ES12]	[As ES12]
	DeleteEvent	[As ES12]	[As ES12]

Table 57: List of Functions

6.5.2.1 "ES2+.DownloadOrder" Function

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON requestMessage> corresponding to the "ES2+.DownloadOrder" function:

```
{
  "type": "object",
  "properties": {
    "eid": {
      "type": "string",
      "pattern": "^[0-9]{32}$",
      "description": "EID as desc in SGP.02"
    },
    "iccid": {
      "type": "string",
      "pattern": "^[0-9]{19}[0-9F]?$",
      "description": "ICCID as described in section 5.2.1"
    },
    "profileType": {
      "type": "string",
      "description": "content free information defined by the Operator"
    }
```

```
   }
}
```

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON responseMessage> corresponding to the "ES2+.DownloadOrder" function:

```
{
   "type": "object",
   "properties": {
      "iccid": {
         "type": "string",
         "pattern": "^[0-9]{19}[0-9F]?$",
         "description": "ICCID as described in section 5.2.1"
      }
   },
   "required": ["iccid"]
}
```

6.5.2.2 "ES2+.ConfirmOrder" Function

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON requestMessage> corresponding to the "ES2+.ConfirmOrder" function:

```
{
   "type": "object",
   "properties": {
      "iccid": {
         "type": "string",
         "pattern": "^[0-9]{19}[0-9F]?$",
         "description": "ICCID as described in section 5.2.1"
      },
      "eid": {
         "type": "string",
         "pattern": "^[0-9]{32}$",
         "description": "EID as described in section 4.3.1"
      },
      "matchingId": {
         "type": "string",
         "description": "as defined in section {5.3.2}"
      },
      "confirmationCode": {
         "type": "string",
         "description": "as defined in section {5.3.2}"
      },
      "smdsAddress": {
         "type": "string",
         "description": "SM-DS to register the event as defined in section {5.3.2}"
      },
      "rootSmdsAddress": {
         "type": "string",
         "description": "#SupportedFromV3.0.0# Root SM-DS addresses as defined in
section 5.3.2"
      },
      "releaseFlag": {
         "type": "boolean",
         "description": "as defined in section {5.3.2}"
      }
   },
   "required": ["iccid", "releaseFlag"]
}
```

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON responseMessage> corresponding to the "ES2+.ConfirmOrder" function:

```
{
   "type": "object",
   "properties": {
      "eid": {
         "type": "string",
         "pattern": "^[0-9]{32}$",
         "description": "EID as desc in SGP.02"
      },
      "matchingId": {
         "type": "string",
         "description": "as defined in section {5.3.2}"
      },
      "smdpAddress": {
         "type": "string",
         "description": "as defined in section {5.3.2}"
      }
   },
   "required": []
}
```

6.5.2.3 "ES2+.CancelOrder" Function

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON requestMessage> corresponding to the "ES2+.CancelOrder" function:

```
{
   "type": "object",
   "properties": {
      "iccid": {
         "type": "string",
         "pattern": "^[0-9]{19}[0-9F]?$",
         "description": "ICCID as described in section 5.2.1"
      },
      "eid": {
         "type": "string",
         "pattern": "^[0-9]{32}$",
         "description": "EID as desc in SGP.02"
      },
      "matchingId": {
         "type": "string",
         "description": "as defined in section {5.3.2}"
      },
      "finalProfileStatusIndicator": {
         "type": "string",
         "description": "as defined in section {5.3.4}"
      }
   }
}
```

6.5.2.4 "ES2+.ReleaseProfile" Function

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON requestMessage> corresponding to the "ES2+.ReleaseProfile" function:

```
{
   "type": "object",
   "properties": {
      "iccid": {
         "type": "string",
         "pattern": "^[0-9]{19}[0-9F]?$",
         "description": "ICCID as described in section 5.2.1"
         },
   "required": ["iccid"]
   }
}
```

6.5.2.5 "ES2+.HandleNotification" Function

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON requestMessage> corresponding to the "ES2+.HandleNotification" function:

```
{
   "type": "object",
   "properties": {
      "eid": {
         "type": "string",
         "pattern": "^[0-9]{32}$",
         "description": "EID as described in SGP.02"
      },
      "iccid": {
         "type": "string",
         "pattern": "^[0-9]{19}[0-9F]?$",
         "description": "ICCID as described in section 5.2.1"
      },
      "matchingId": {
         "type": "string",
         "description": "as defined in section 5.3.5"
      },
      "notificationReceiverIdentifier": {
         "type": "string",
         "description": "#SupportedFromV3.0.0# as defined in section 5.3.5"
      },
      "notificationIdentifier": {
         "type": "string",
         "description": "#SupportedFromV3.0.0# as defined in section 5.3.5"
      },
      "profileType": {
         "type": "string",
         "description": "Content free information defined by the Operator
(e.g.'P9054-2')"
      },
      "timestamp": {
         "type": "string",
         "pattern": "^[0-9]{4}-[0-9]{2}-[0-9]{2}T[0-9]{2}:[0-9]{2}:[0-
9]{2}(Z|([+\\-][0-9]{2}:[0-9]{2}))$",
         "description": "String format as specified by W3C: YYYY-MM-DDThh:mm:ssTZD
(E.g., 2001-12-17T09:30:47Z)"
      },
    "notificationEvent": {
      "type": "integer",
      "description": "Identification of the step reached in the procedure or the
forwarded event"
    },
    "notificationEventStatus": {
      "type": "object",
      "description": "ExecutionStatus Common Data Type",
      "properties": {
        "status": {
          "type": "string",
          "description": "Executed-Success, Failed"
        },
        "statusCodeData": {
          "type": "object",
          "properties": {
            "subjectCode": {
              "type": "string",
              "description": "OID of the subject code"
            },
            "reasonCode": {
              "type": "string",
              "description": "OID of the reason code"
            },
            "subjectIdentifier": {
              "type": "string",
              "description": "Identifier of the subject"
            },
            "message": {
              "type": "string",
              "description": "Textual and human readable explanation"
            }
          },
          "required": ["subjectCode", "reasonCode"]
        }
      },
      "required": ["status"]
    },
    "resultData": {
      "type": "string",
      "format": "base64",
      "description": "finalResult data object contained in the
ProfileInstallationResult or LoadRpmPackageResult"
    }
  },
  "required": ["timestamp", "notificationEvent"]
}
```

6.5.2.6 "ES9+.InitiateAuthentication" Function

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON requestMessage> corresponding to the "ES9+.InitiateAuthentication" function:

```
{
  "type": "object",
  "properties": {
    "euiccChallenge": {
      "type": "string",
         "format": "base64",
         "description": "base64 encoding of the value field of the eUICC Challenge 
defined in Section 5.6.1 (without tag and length fields)"
      },
      "euiccInfo1": {
         "type": "string",
         "format": "base64",
         "description": "euiccinfo1 defined in Section 5.6.1"
      },
      "smdpAddress": {
         "type": "string",
         "description": "SM-DP+ Address as defined in Section 5.6.1"
      },
      "lpaRspCapability": {
         "type": "string",
         "format": "base64",
         "description": "#SupportedFromV3.0.0# lpaRspCapability defined in Section 
5.6.1"
      }
   },
   "required": ["euiccChallenge", "euiccInfo1", "smdpAddress"]
}
```

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON responseMessage> corresponding to the "ES9+.InitiateAuthentication" function:

```
{
   "type": "object",
   "properties": {
      "transactionId": {
         "type": "string",
         "pattern": "^[0-9,A-F]{2,32}$",
         "description": "TransactionID defined in Section 5.6.1"
      },
      "serverSigned1": {
         "type": "string",
         "format": "base64",
         "description": "The data object as required by ES10b.AuthenticateServer"
      },
      "serverSignature1": {
         "type": "string",
         "format": "base64",
         "description": "The signature as required by ES10b.AuthenticateServer"
      },
      "euiccCiPKIdToBeUsed": {
         "type": "string",
         "format": "base64",
         "description": "The CI Public Key to be used as required by 
ES10b.AuthenticateServer"
      },
      "serverCertificate": {
         "type": "string",
         "format": "base64",
         "description": "The server Certificate as required by 
ES10b.AuthenticateServer"
      },
      "otherCertsInChain": {
         "type": "string",
         "format": "base64",
         "description": "#SupportedFromV3.0.0# The remaining part of the server
Certificate chain as described in 5.6.1"
      },
      "crlList": {
         "type": "string",
         "format": "base64",
         "description": "#SupportedFromV3.0.0# The CRL list as described in
ES10b.AuthenticateServer"
      }
   },
   "required": ["transactionId", "serverSigned1", "serverSignature1",
"serverCertificate"]
}
```

NOTE: LPA is in charge of transcoding the transactionId.

6.5.2.7 "ES9+.GetBoundProfilePackage" Function

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON requestMessage> corresponding to the "ES9+.GetBoundProfilePackage" function:

```
{
   "type": "object",
   "properties": {
      "transactionId": {
         "type": "string",
         "pattern": "^[0-9,A-F]{2,32}$",
         "description": "TransactionID defined in Section 5.6.2"
      },
      "prepareDownloadResponse": {
         "type": "string",
         "format": "base64",
         "description": "PrepareDownloadResponse defined in Section 5.6.2"
      }
   },
   "required": ["transactionId", "prepareDownloadResponse"]
}
```

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON responseMessage> corresponding to the "ES9+.GetBoundProfilePackage" function:

```
{
   "type": "object",
   "properties": {
      "transactionId": {
         "type": "string",
         "pattern": "^[0-9,A-F]{2,32}$",
         "description": "TransactionID defined in Section 5.6.2"
      },
      "boundProfilePackage": {
         "type": "string",
         "format": "base64",
         "description": "Bound Profile Package defined in Section 5.6.2"
      }
   },
   "required": ["transactionId", "boundProfilePackage"]
}
```

6.5.2.8 "ES9+.AuthenticateClient" Function

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON requestMessage> corresponding to the "ES9+.AuthenticateClient" function:

```
{
   "type": "object",
   "properties": {
      "transactionId": {
         "type": "string",
         "pattern": "^[0-9,A-F]{2,32}$",
         "description": "TransactionID defined in Section 5.6.3"
      },
      "authenticateServerResponse": {
         "type": "string",
         "format": "base64",
         "description": "AuthenticateServerResponse defined in Section 5.6.3"
      },
      "deleteNotificationForDc": {
         "type": "string",
         "format": "base64",
         "description": "#SupportedForDcV3.0.0# containing deleteNotificationForDc
TLV, see section 6.6.2.2"
      },
   },
   "required": ["transactionId", "authenticateServerResponse"]
}
```

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON responseMessage> corresponding to the "ES9+.AuthenticateClient" function:

```
{
   "type": "object",
   "properties": {
      "transactionId": {
         "type": "string",
         "pattern": "^[0-9,A-F]{2,32}$",
         "description": "TransactionID defined in Section 5.6.3"
      }
   },
   "required": ["transactionId"],
   "oneOf": [
      {
         "properties": {
            "profileMetadata": {
               "type": "string",
               "format": "base64",
               "description": "StoreMetadataRequest defined in section 5.5.3"
            },
            "smdpSigned2": {
               "type": "string",
               "format": "base64",
               "description": "SmdpSigned2 encoded data object"
            },
            "smdpSignature2": {
               "type": "string",
               "format": "base64",
               "description": "SM-DP+ signature as defined in ES10b.PrepareDownload"
            },
            "smdpCertificate": {
               "type": "string",
               "format": "base64",
               "description": "The Certificate as required by ES10b.PrepareDownload"
            }
         },
         "required":
["profileMetadata","smdpSigned2","smdpSignature2","smdpCertificate"]
      },{
         "properties": {
            "smdpSigned3": {
               "type": "string",
               "format": "base64",
               "description": "SmdpSigned3 encoded data object"
            },
            "smdpSignature3": {
               "type": "string",
               "format": "base64",
               "description": "SM-DP+ signature as defined in ES10b.LoadRpmPackage"
            }
         },
         "required": ["smdpSigned3","smdpSignature3"]
      },{
         "properties": {
            "smdpSigned4": {
               "type": "string",
               "format": "base64",
               "description": "SmdpSigned4 encoded data object"
            },
            "smdpSignature4": {
               "type": "string",
               "format": "base64",
               "description": "SM-DP+ signature as defined in
ES10b.PrepareDeviceChange"
            },
            "serviceProviderMessageForDc": {
               "type": "string",
               "format": "base64",
               "description": "#SupportedForDcV3.0.0# Service Provider Message For
Device Change defined in section 6.6.2.2."
            }
         },
         "required": ["smdpSigned4","smdpSignature4"]
      }
   ]
}
```

Depending on the targeted RSP Server (SM-DP+ or SM-DS) the response MAY be a <JSON body> corresponding to "ES9+.AuthenticateClient" or "ES11.AuthenticateClient" function.

6.5.2.9 "ES9+.HandleNotification" Function

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON requestMessage> corresponding to the "ES9+.HandleNotification" function.

```
{
  "type": "object",
  "properties": {
    "pendingNotification": {
      "type": "string",
      "format": "base64",
      "description": "PendingNotification defined in section 5.7.10"
    }
  },
  "required": ["pendingNotification"]
}
```

6.5.2.10 "ES9+.CancelSession" Function

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON requestMessage> corresponding to the "ES9+.CancelSession" function:

```
{
  "type": "object",
  "properties": {
    "transactionId": {
      "type": "string",
      "pattern": "^[0-9,A-F]{2,32}$",
      "description": "TransactionID defined in Section 5.6.5"
    },
    "cancelSessionResponse": {
      "type": "string",
      "format": "base64",
      "description": "CancelSessionResponse data object defined in Section 
5.7.14"
    }
  },
  "required": ["transactionId", "cancelSessionResponse"]
}
```

The "ES9+.CancelSession" function has no <JSON body> part of the <JSON responseMessage>.

6.5.2.11 "ES11.InitiateAuthentication" Function

The <JSON body> part of the <JSON requestMessage> and <JSON body> part of the <JSON responseMessage> corresponding to the "ES11.InitiateAuthentication" function is identical to the one defined for the "ES9+.InitiateAuthentication" function.

6.5.2.12 "ES11.AuthenticateClient" Function

The <JSON body> part of the <JSON requestMessage> corresponding to the "ES11.AuthenticateClient" function is identical to the one defined for the "ES9+.AuthenticateClient" function. Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON responseMessage> corresponding to the "ES11.AuthenticateClient" function:

```
{
  "type": "object",
  "properties": {
    "transactionId": {
         "type": "string",
         "pattern": "^[0-9,A-F]{2,32}$",
         "description": "TransactionID, see Section 5.8.2"
      },
   },
   "required": ["transactionId"],
   "oneOf": [
      {
         "properties": {
            "eventEntries": {
               "type": "array",
               "description": "#SupportedOnlyBeforeV3.0.0#",
               "items": {
                  "type": "object",
                  "description": "data containing EVENT_RECORD, see section 5.8.2",
                  "properties": {
                     "eventId": {
                        "type": "string",
                        "description": "Identification of the event"
                     },
                     "rspServerAddress": {
                        "type": "string",
                        "description": "RSP Server address where the event can be
found"
                     }
                  },
                  "required": ["eventId", "rspServerAddress"]
               }
            }
         },
         "required": ["eventEntries"]
      },{
         "properties": {
            "smdsSigned2": {
               "type": "string",
               "format": "base64",
               "description": "#SupportedFromV3.0.0# SM-DS response as defined in
ES10a.VerifySmdsResponse"
            },
            "smdsSignature2": {
               "type": "string",
               "format": "base64",
               "description": "#SupportedFromV3.0.0# SM-DS signature as defined in
ES10a.VerifySmdsResponse"
            },
         },
         "required": ["smdsSigned2", "smdsSignature2"]
      }
   ]
}
```

Depending on the targeted RSP Server (SM-DP+ or SM-DS) the response MAY be a <JSON body> corresponding to "ES9+.AuthenticateClient" or "ES11.AuthenticateClient" function.

6.5.2.13 "ES12.RegisterEvent" Function

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON requestMessage> corresponding to the "ES12.RegisterEvent" function:

```
{
   "type": "object",
   "properties": {
      "eid": {
         "type": "string",
         "pattern": "^[0-9]{32}$",
         "description": "EID as desc in SGP.02"
      },
      "rspServerAddress": {
         "type": "string",
         "description": "as defined in section 5.9.1"
      },
      "eventId": {
         "type": "string",
         "description": "as defined in section 5.9.1"
      },
      "forwardingIndicator": {
         "type": "boolean",
         "description": "indicates whether the event is cascaded to a Root SM-DS"
      },
      "rootSmdsAddress": {
         "type": "string",
         "description": "the FQDN of the Root SM-DS for cascaded registration"
      },
      "eventType": {
         "type": "integer",
         "description": "denotes the RSP Operation of this Event Record, see
Section 5.9.1"
      },
      "hashedIccids": {
         "type": "array",
         "description": "identification of the target Profile(s), each calculated
as SHA256(ICCID) or SHA256(ICCID|Salt)",
         "items": {
            "type": "string",
            "pattern": "^[0-9,A-F]{64}$"
         }
      },
      "salt": {
         "type": "string",
         "pattern": "^[0-9,A-F]{16,32}$",
         "description": "random value to calculate hashedIccids"
      },
      "serviceProviderName": {
         "type": "string",
         "description": "name of the Service Provider that requested the RSP
Operation of this Event Record"
      },
      "operatorId": {
         "type": "string",
         "format": "base64",
         "description": "identification of the Operator that requested the RSP
Operation of this Event Record, see Section 5.9.1"
      }
   },
   "required": ["eid", "rspServerAddress", "eventId", "forwardingIndicator"],
   "dependencies": {
      "salt": {
         "$comment": "salt can be present only if hashedIccids is present",
         "required": [ "hashedIccids" ]
      },
      "rootSmdsAddress": {
         "$comment": "smdsAddress can be present only if forwardingIndicator is
true",
         "properties": {
            "forwardingIndicator": { "const": true }
         }
      }
   }
}
```

This function has no <JSON body> part of the <JSON responseMessage>.

6.5.2.14 "ES12.DeleteEvent" Function

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON requestMessage> corresponding to the "ES12.DeleteEvent" function:

```
{
   "type": "object",
   "properties": {
      "eid": {
         "type": "string",
         "pattern": "^[0-9]{32}$",
         "description": "EID as desc in SGP.02"
      },
      "eventId": {
         "type": "string",
         "description": "as defined in section 5.9.2"
      }
   },
   "required": ["eid", "eventId"]
}
```

This function has no <JSON body> part of the <JSON responseMessage>.

6.5.2.15 "ES2+.RpmOrder" Function

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON requestMessage> corresponding to the "ES2+.RpmOrder" function:

```
{
   "type": "object",
   "properties": {
      "eid": {
         "type": "string",
         "pattern": "^[0-9]{32}$",
         "description": "EID as described in SGP.02"
      },
      "rpmScript": {
         "type": "string",
         "format": "base64",
         "description": "RpmPackage defined in section 2.10.1"
      },
    "matchingId": {
      "type": "string",
      "description": "as defined in section 5.3.6"
    },
    "rootSmdsAddress": {
      "type": "string",
      "description": "as defined in section 5.3.6"
    },
    "altSmdsAddress": {
      "type": "string",
      "description": "as defined in section 5.3.6"
    }
  },
  "required": ["eid","rpmScript"],
  "dependencies": { "altSmdsAddress": [ "rootSmdsAddress" ] }
}
```

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON responseMessage> corresponding to the "ES2+.RpmOrder" function:

```
{
  "type": "object",
  "properties": {
    "matchingId": {
      "type": "string",
      "description": "as defined in section {5.3.6}"
    }
  },
  "required": ["matchingId"]
}
```

6.5.2.16 "ES2+.HandleDeviceChangeRequest" Function

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON requestMessage> corresponding to the "ES2+.HandleDeviceChangeRequest" function:

```
{
  "type": "object",
  "properties": {
    "iccid": {
      "type": "string",
      "pattern": "^[0-9]{19}[0-9F]?$",
      "description": "ICCID as described in section 5.2.1"
    },
    "eid": {
      "type": "string",
      "pattern": "^[0-9]{32}$",
      "description": "EID as described in SGP.02"
    },
    "tac": {
      "type": "string",
      "pattern": "^[0-9]{8}$",
      "description": "TAC as described in section 4.2"
    }
  },
  "required": ["iccid"]
}
```

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON responseMessage> corresponding to the "ES2+.HandleDeviceChangeRequest" function:

```
{
   "type": "object",
   "properties": {
      "isNewProfileRequired": {
         "type": "boolean",
         "description": "as defined in section 5.3.7"
      },
      "serviceProviderMessageForDc": {
         "type": "string",
         "format": "base64",
         "description": "Service Provider Message For Device Change defined in 
section 6.6.2.2."
      },
      "cc" : {
         "type" : "string",
         "description" : "as defined in section 5.3.7"
      }
   },
   "required": ["isNewProfileRequired"]
}
```

6.5.2.17 "ES9+.ConfirmDeviceChange" Function

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON requestMessage> corresponding to the "ES9+.ConfirmDeviceChange" function:

```
{
   "type": "object",
   "properties": {
      "transactionId": {
         "type": "string",
         "pattern": "^[0-9,A-F]{2,32}$",
         "description": "TransactionID defined in Section 5.6.5"
      },
      "prepareDeviceChangeResponse": {
         "type": "string",
         "format": "base64",
         "description": "PrepareDeviceCHangeResponse defined in Section 5.7.26"
      }
   },
   "required": ["transactionId","prepareDeviceChangeResponse"]
}
```

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON responseMessage> corresponding to the " ES9+.ConfirmDeviceChange" function:

```
{
   "type": "object",
   "properties": {
      "transactionId": {
         "type": "string",
         "pattern": "^[0-9,A-F]{2,32}$",
         "description": "TransactionID defined in Section 5.6.5"
      },
      "smdpSigned5": {
         "type": "string",
         "format": "base64",
         "description": "SmdpSigned5 encoded data object as defined in
ES10b.VerifyDeviceChangeConfirmation"
      },
      "smdpSignature5": {
         "type": "string",
         "format": "base64",
         "description": "SM-DP+ signature as defined in
"ES10b.VerifyDeviceChangeConfirmation"
      }
   },
   "required": ["transactionId"],
   "$comment": "smdpSigned5 and smdpSignature5 are either both present or both
absent",
   "dependencies": {
      "smdpsigned5" : ["smdpSignature5"],
      "smdpSignature5": ["smdpSigned5"]
   }
}
```

6.5.2.18 "ES11.CheckEvent" Function

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON requestMessage> corresponding to the "ES11.CheckEvent" function:

```
{
   "type": "object",
   "properties": {
      "ecId": {
         "type": "string",
         "pattern": "^[0-9,A-F]{32,64}$",
         "description": "Event Checking Identifier"
      },
      "smdsAddress": {
         "type": "string",
         "description": "SM-DS Address"
      }
   },
   "required": ["ecId", "smdsAddress"]
}
```

Hereunder is the definition of the JSON schema for the <JSON body> part of the <JSON responseMessage> corresponding to the "ES11.CheckEvent" function:

```
{
   "type": "object",
   "properties": {
      "isPendingEvent": {
         "type": "boolean",
         "description": "Indicates if an Event Record corresponding to the received
ECID is pending"
      }
   },
   "required": ["isPendingEvent"]
}
```

6.6 Function Binding in ASN.1

6.6.1 ASN.1 message definition

The Function requester and the Function Provider SHALL exchange the DER encoded ASN.1 objects in HTTP messages as follows.

- HTTP Request SHALL have the following format.

```
HTTP POST gsma/rsp2/asn1 HTTP/1.1
Host: <Server Address>
User-Agent: <User Agent>
X-Admin-Protocol: gsma/rsp/v<x.y.z>
Content-Type: application/x-gsma-rsp-asn1
Content-Length: <Length of the ASN.1 RemoteProfileProvisioningRequest>

<ASN.1 RemoteProfileProvisioningRequest>
```

Any function execution request using ASN.1 binding SHALL be sent to the generic HTTP path '`gsma/rsp2/asn1`'.

The body part of the HTTP POST request SHALL contain one Remote Profile Provisioning Request objects defined as follows:

```
-- ASN1START
RemoteProfileProvisioningRequest ::= [2] CHOICE {  -- Tag 'A2'
   initiateAuthenticationRequest [57] InitiateAuthenticationRequest,  -- Tag 'BF39'
   authenticateClientRequest [59] AuthenticateClientRequest, -- Tag 'BF3B'
   getBoundProfilePackageRequest [58] GetBoundProfilePackageRequest,  -- Tag 'BF3A'
   cancelSessionRequestEs9 [65] CancelSessionRequestEs9, -- Tag 'BF41'
   handleNotification [61] HandleNotification, -- tag 'BF3D'
   confirmDeviceChangeRequest [76] ConfirmDeviceChangeRequest, -- Tag 'BF4C'
   checkEventRequest [70] CheckEventRequest -- Tag 'BF46'
}
-- ASN1STOP
```

HTTP Response SHALL have the following format:

```
HTTP/1.1 <HTTP Status Code>
X-Admin-Protocol: gsma/rsp/v<x.y.z>
Content-Type: application/x-gsma-rsp-asn1
Content-Length: <Length of the ASN.1 RemoteProfileProvisioningResponse>

<ASN.1 RemoteProfileProvisioningResponse>
```

The body part of the HTTP POST response SHALL contain one Remote Profile Provisioning Response object defined as follows:

```
-- ASN1START
RemoteProfileProvisioningResponse ::= [2] CHOICE { -- Tag 'A2'
   initiateAuthenticationResponse [57] InitiateAuthenticationResponse, -- Tag 
'BF39'
   authenticateClientResponseEs9 [59] AuthenticateClientResponseEs9, -- Tag 'BF3B'
   getBoundProfilePackageResponse [58] GetBoundProfilePackageResponse, -- Tag 
'BF3A'
   cancelSessionResponseEs9 [65] CancelSessionResponseEs9, -- Tag 'BF41'
   authenticateClientResponseEs11 [64] AuthenticateClientResponseEs11, -- Tag 
'BF40'
```

```
   confirmDeviceChangeResponse [76] ConfirmDeviceChangeResponse, -- Tag 'BF4C'
   checkEventResponse [70] CheckEventResponse -- Tag 'BF46'
}
-- ASN1STOP
```

6.6.1.1 Common status codes

The following values map the common status codes defined in section 5.2.6.3 that can be returned by any function.

```
invalidInputData(124) -- maps status code "Function-Invalid"
missingInputData(125) -- maps status code "Function - Mandatory Element Missing"
and "Function - Conditional Element Missing"
functionProviderBusy(126) -- maps status code "Function provider - Busy"
undefinedError(127) -- maps status code "Function provider - Execution Error"
```

NOTE: Status codes "Function requester - Unknown (Identification or Authentication)", "Function requester - Not allowed (authorisation)" and "Validity period - Refused" are not relevant for ES9+ and ES11 interfaces.

6.6.2 List of functions

6.6.2.1 "ES9+.InitiateAuthentication" Function

The "ES9+.InitiateAuthentication" request function is defined as follows:

```
-- ASN1START
InitiateAuthenticationRequest ::= [57] SEQUENCE { -- Tag 'BF39'
   euiccChallenge [1] Octet16, -- random eUICC challenge
   smdpAddress [3] UTF8String,
   euiccInfo1 EUICCInfo1,
   lpaRspCapability LpaRspCapability OPTIONAL -- #SupportedFromV3.0.0#
}
-- ASN1STOP
```

The "ES9+.InitiateAuthentication" response function is defined as follows:

```
-- ASN1START
InitiateAuthenticationResponse ::= [57] CHOICE { -- Tag 'BF39'
   initiateAuthenticationOk InitiateAuthenticationOkEs9,
   initiateAuthenticationError INTEGER {
      invalidDpAddress(1),
      euiccVersionNotSupportedByDp(2), -- #SupportedOnlyBeforeV3.0.0#
      ciPKIdNotSupported(3),
      invalidInputData(124), -- #SupportedFromV3.0.0#
      missingInputData(125), -- #SupportedFromV3.0.0#
      functionProviderBusy(126), -- #SupportedFromV3.0.0#
      undefinedError(127) -- #SupportedFromV3.0.0#
   }
}

InitiateAuthenticationOkEs9 ::= SEQUENCE {
   transactionId [0]TransactionId, -- The TransactionID generated by the SM-DP+
   serverSigned1 ServerSigned1, -- Signed information
   serverSignature1 [APPLICATION 55] OCTET STRING, -- Server Sign1, tag '5F37'
   euiccCiPKIdToBeUsed SubjectKeyIdentifier OPTIONAL, -- The CI Public Key to be
used as required by ES10b.AuthenticateServer
   serverCertificate Certificate,
   otherCertsInChain [1] CertificateChain OPTIONAL, -- #SupportedFromV3.0.0#
   crlList [2] SEQUENCE OF CertificateList OPTIONAL -- #SupportedFromV3.0.0# From
RFC 5280
}
-- ASN1STOP
```

6.6.2.2 "ES9+.AuthenticateClient" Function

The "ES9+.AuthenticateClient" request function is defined as follows:

```
-- ASN1START
AuthenticateClientRequest ::= [59] SEQUENCE {  -- Tag 'BF3B'
   transactionId [0] TransactionId,
   authenticateServerResponse [56] AuthenticateServerResponse, -- This is the
response from ES10b.AuthenticateServer
   deleteNotificationForDc [1] DeleteNotificationForDc -- #SupportedForDcV3.0.0#
Delete Notification for Device Change, see section 4.1.3
}
-- ASN1STOP
```

The "ES9+.AuthenticateClient" response function is defined as follows:

```
-- ASN1START
AuthenticateClientResponseEs9 ::= [59] CHOICE {  -- Tag 'BF3B'
   authenticateClientOk AuthenticateClientOk,
   authenticateClientError INTEGER {
      eumCertificateInvalid(1),
      eumCertificateExpired(2),
      euiccCertificateInvalid(3),
      euiccCertificateExpired(4),
      euiccSignatureInvalid(5),
      matchingIdRefused(6),
      eidMismatch(7),
      noEligibleProfile(8),
      ciPKUnknown(9),
      invalidTransactionId(10),
      insufficientMemory(11),
      ciPKMismatch(12), -- #SupportedFromV3.0.0#
      euiccRspCapabilityHasChanged(13), -- #SupportedFromV3.0.0#
      lpaRspCapabilityHasChanged(14), -- #SupportedFromV3.0.0#
      deviceChangeNotSupported(15), -- #SupportedForDcV3.0.0#
      deviceChangeNotAllowed(16), -- #SupportedForDcV3.0.0#
      iccidUnkwon(17), -- #SupportedForDcV3.0.0#
      invalidInputData(124), -- #SupportedFromV3.0.0#
      missingInputData(125), -- #SupportedFromV3.0.0#
      functionProviderBusy(126), -- #SupportedFromV3.0.0#
      undefinedError(127)
   },
   authenticateClientOkRpm AuthenticateClientOkRpm, -- #SupportedForRpmV3.0.0#
   authenticateClientOkDeviceChange AuthenticateClientOkDeviceChange --
#SupportedForDcV3.0.0#
}

AuthenticateClientOk ::= SEQUENCE {
   transactionId [0] TransactionId,
   profileMetadata [37] StoreMetadataRequest OPTIONAL,
   smdpSigned2 [1] SmdpSigned2 OPTIONAL, -- Signed information
   smdpSignature2 [APPLICATION 55] OCTET STRING OPTIONAL,    -- tag '5F37'
   smdpCertificate Certificate -- CERT.DPpb.SIG
}

AuthenticateClientOkRpm ::= SEQUENCE {
   transactionId [0] TransactionId,
   smdpSigned3 [3] SmdpSigned3,
   smdpSignature3 [APPLICATION 55] OCTET STRING
}

AuthenticateClientOkDeviceChange ::= SEQUENCE {
   transactionId [0] TransactionId,
   smdpSigned4 [1] SmdpSigned4, -- Signed information
   smdpSignature4 [APPLICATION 55] OCTET STRING,      -- tag '5F37'
   serviceProviderMessageForDc [1] LocalisedTextMessage OPTIONAL -- Service
Provider Message For Device Change
}
-- ASN1STOP
```

6.6.2.3 "ES9+.GetBoundProfilePackage" Function

The "ES9+.GetBoundProfilePackage" request function is defined as follows:

```
-- ASN1START
GetBoundProfilePackageRequest ::= [58] SEQUENCE {  -- Tag 'BF3A'
   transactionId [0] TransactionId,
   prepareDownloadResponse [33] PrepareDownloadResponse
}
-- ASN1STOP
```

The "ES9+.GetBoundProfilePackage" response function is defined as follows:

```
-- ASN1START
GetBoundProfilePackageResponse ::= [58] CHOICE {  -- Tag 'BF3A'
   getBoundProfilePackageOk GetBoundProfilePackageOk,
   getBoundProfilePackageError INTEGER {
      euiccSignatureInvalid(1),
      confirmationCodeMissing(2),
      confirmationCodeRefused(3),
      confirmationCodeRetriesExceeded(4),
      bppRebindingRefused(5),
      downloadOrderExpired(6),
      invalidTransactionId(95),
      invalidInputData(124), -- #SupportedFromV3.0.0#
      missingInputData(125), -- #SupportedFromV3.0.0#
      functionProviderBusy(126), -- #SupportedFromV3.0.0#
      undefinedError(127)
   }
}

GetBoundProfilePackageOk ::= SEQUENCE {
   transactionId [0] TransactionId,
   boundProfilePackage [54] BoundProfilePackage
}
-- ASN1STOP
```

NOTE: The eUICC MAY start processing of the BPP before having received the full package and having been able to check for a correct TLV structure.

6.6.2.4 "ES9+.HandleNotification" Function

The "ES9+.HandleNotification" request function SHALL consist of the data structure defined for PendingNotification in section 5.7.10. The function is defined as follows:

```
-- ASN1START
HandleNotification ::= [61] SEQUENCE { -- Tag 'BF3D'
   pendingNotification PendingNotification
}
-- ASN1STOP
```

The function has no response.

6.6.2.5 "ES9+.CancelSession" Function

The "ES9+.CancelSession" request function is defined as follows:

```
-- ASN1START
CancelSessionRequestEs9 ::= [65] SEQUENCE { -- Tag 'BF41'
   transactionId TransactionId,
   cancelSessionResponse  CancelSessionResponse -- data structure defined for
ES10b.CancelSession function
}
-- ASN1STOP
```

The "ES9+.CancelSession" response function is defined as follows:

```
-- ASN1START
CancelSessionResponseEs9 ::= [65] CHOICE { -- Tag 'BF41'
   cancelSessionOk CancelSessionOk,
   cancelSessionError INTEGER {
      invalidTransactionId(1),
      euiccSignatureInvalid(2),
      invalidInputData(124), -- #SupportedFromV3.0.0#
      missingInputData(125), -- #SupportedFromV3.0.0#
      functionProviderBusy(126), -- #SupportedFromV3.0.0#
      undefinedError(127)
   }
}

CancelSessionOk ::= SEQUENCE { -- This function has no output data
}
-- ASN1STOP
```

6.6.2.6 "ES11.InitiateAuthentication" Function

The `InitiateAuthenticationRequest` and `InitiateAuthenticationResponse` for the binding of the "ES11.InitiateAuthentication" function are identical to the ones defined for the "ES9+.InitiateAuthentication" function.

6.6.2.7 "ES11.AuthenticateClient" Function

The `AuthenticateClientRequest` for the binding of the "ES11.AuthenticateClient" function is identical to the one defined for the "ES9+.AuthenticateClient" function.

The "ES11.AuthenticateClient" response data object is defined as follows:

```
-- ASN1START
AuthenticateClientResponseEs11 ::= [64] CHOICE {  -- Tag 'BF40'
   authenticateClientOk AuthenticateClientOkEs11V2, -- #SupportedOnlyBeforeV3.0.0#
   authenticateClientError INTEGER {
      eumCertificateInvalid(1),
      eumCertificateExpired(2),
      euiccCertificateInvalid(3),
      euiccCertificateExpired(4),
      euiccSignatureInvalid(5),
      eventIdUnknown(6),
      invalidTransactionId(7),
      ciPKUnknown(8), -- #SupportedFromV3.0.0#
      ciPKMismatch(9), -- #SupportedFromV3.0.0#
      euiccRspCapabilityHasChanged(10), -- #SupportedFromV3.0.0#
      lpaRspCapabilityHasChanged(11), -- #SupportedFromV3.0.0#
      pushServiceNotSupport(12), -- #SupportedForPushServiceV3.0.0#
      pushServiceRegistrationNotSupported(13), -- #SupportedForPushServiceV3.0.0#
      invalidInputData(124), -- #SupportedFromV3.0.0#
      missingInputData(125), -- #SupportedFromV3.0.0#
      functionProviderBusy(126), -- #SupportedFromV3.0.0#
      undefinedError(127)
   },
   authenticateClientOkV3 AuthenticateClientOkEs11V3 -- #SupportedFromV3.0.0#
}

AuthenticateClientOkEs11V2 ::= SEQUENCE { -- #SupportedOnlyBeforeV3.0.0#
   transactionId [0] TransactionId,
   eventEntries [1] SEQUENCE OF EventRecord
}

EventRecord ::= SEQUENCE { -- #SupportedOnlyBeforeV3.0.0#
   eventId UTF8String,
   rspServerAddress UTF8String
}

AuthenticateClientOkEs11V3 ::= SEQUENCE {
   transactionId [0] TransactionId,
   smdsSigned2 [1] SmdsSigned2,
   smdsSignature2 [APPLICATION 55] OCTET STRING -- tag '5F37'
}

-- ASN1STOP
```

6.6.2.8 "ES11.CheckEvent" Function

The "ES11.CheckEvent" request function is defined as follows:

```
-- ASN1START
CheckEventRequest ::= [70] SEQUENCE {  -- #SupportedForEventCheckingV3.0.0# Tag
'BF46'
   ecId [0] OCTET STRING(SIZE(16..32)), -- Event Checking Identifier
   smdsAddress [1] UTF8String
}
-- ASN1STOP
```

The "ES11.CheckEvent" response function is defined as follows:

```
-- ASN1START
CheckEventResponse ::= [70] CHOICE {  -- #SupportedForEventCheckingV3.0.0# Tag
'BF46'
   checkEventOk CheckEventOk,
   checkEventError INTEGER {
      invalidDsAddress(1),
      eventCheckingNotSupported(2), -- #SupportedForEventCheckingV3.0.0#
      expiredEcid(3), -- #SupportedForEventCheckingV3.0.0#
      unknownEcid(4), -- #SupportedForEventCheckingV3.0.0#
      invalidInputData(124), -- #SupportedFromV3.0.0#
      missingInputData(125), -- #SupportedFromV3.0.0#
      functionProviderBusy(126), -- #SupportedFromV3.0.0#
      undefinedError(127) -- #SupportedFromV3.0.0#
   }
}

CheckEventOk ::= SEQUENCE {
   isPendingEvent [0] BOOLEAN -- Indicates if an Event Record corresponding to the
received ECID exists
}
```

```
-- ASN1STOP
```

6.6.2.9 "ES9+.ConfirmDeviceChange" Function

The "ES9+.ConfirmDeviceChange" request function is defined as follows:

```
-- ASN1START
ConfirmDeviceChangeRequest ::= [76] SEQUENCE { -- #SupportedForDcV3.0.0# Tag 'BF4C'
   transactionId TransactionId,
   prepareDownloadResponse [33] PrepareDownloadResponse
}
-- ASN1STOP
```

The "ES9+.ConfirmDeviceChange" response function is defined as follows:

```
-- ASN1START
ConfirmDeviceChangeResponse ::= [76] CHOICE { -- #SupportedForDcV3.0.0# Tag 'BF4C'
   confirmDeviceChangeOk ConfirmDeviceChangeOk,
   confirmDeviceChangeError INTEGER {
      invalidTransactionId(1),
      euiccSignatureInvalid(2),
      confirmationCodeMissing(3),
      confirmationCodeRefused(4),
      confirmationCodeInvalidMatch(5),
      confirmationCodeRetriesExceeded(6),
      invalidInputData(124),
      missingInputData(125),
      functionProviderBusy(126),
      undefinedError(127)
   }
}

ConfirmDeviceChangeOk ::= SEQUENCE {
   transactionId [0] TransactionId,
   smdpSigned5 SmdpSigned5,
   smdpSignature5 OCTET STRING
}
-- ASN1STOP
```

The following ASN.1 definition shall stay at the end of any ASN.1 definitions.

```
-- ASN1START
END
-- ASN1STOP
```

Annex A Use of GlobalPlatform Privileges (Normative)

The eUICC architecture defined in this specification relies on the ISD-R, ISD-P, MNO-SD and ECASD Security Domains defined in SGP.02 [2].

The GlobalPlatform privileges allocation defined in SGP.02 [2] SHALL be applicable for the ISD-R, ISD-P, MNO-SD and ECASD Security Domains as well as Applications inside a Profile.

Annex B Data Definitions (Normative)

- Coding of the IMEI

The value of the IMEI SHALL be coded as defined in section 4.2.

Annex C Device Requirements (Normative)

C.1 Functional Device Requirements

Functional Device Requirements No.	Requirement
DEV1	For connectivity the Device SHALL support at least one of the network access technologies defined by 3GPP or 3GPP2: • UDP over IP as defined in RFC 768 [34] (subject to the right support of access network technology) • TCP over IP as defined in RFC 793 [19].
DEV2	For Network connection control the Device SHALL support: • RPLMN details (LAC/TAC, NMR). • QoS (failures, duration, power, location). • New network selection after SIM/USIM update.
DEV3	The Device SHALL contain a unique IMEI (International Mobile Equipment Identity) value compliant with the format defined in 3GPP TS 23.003 [35] and/or a unique MEID as defined in 3GPP2 S.R0048-A [36].
DEV4	The Device SHALL support, as a minimum, the following set of proactive commands: • PROVIDE LOCAL INFORMATION (location information, IMEI, NMR, date and time, access technology, at least). • POLL INTERVAL, POLLING OFF, TIMER MANAGEMENT [at least one timer], ENVELOPE (TIMER EXPIRATION). • SET UP EVENT LIST and ENVELOPE (EVENT DOWNLOAD). • REFRESH Command (At least mode 4 - "UICC reset")
DEV5	The Device SHALL comply with the IMEI security requirements defined in the GSMA-EICTA document "Security Principles Related to Handset Theft" [22].
DEV6	A Device SHALL be able to handle an eUICC without any installed Profiles.
DEV7	If a Companion Device does not have the capability itself to communicate directly with the SM-DP+, it SHALL use a Primary Device as a conduit, allowing it to communicate with the SM-DP+.
DEV8	At least one of the Primary or Companion Device SHALL have a UI that allows the secure capture of User Intent.
DEV9	At least one of the Primary or Companion Device SHALL have a UI that allows the user to initiate a Profile Download or Local Profile Management.
DEV10	The Device SHALL conform to the terminal requirements within ETSI TS 102 221 [6].
DEV11	A Device implementation of personalisation ("SIM lock") as defined by 3GPP TS 22.022 SHALL operate the same with an enabled eUICC Profile as with a legacy UICC.
DEV12	(Deprecated in this formulation, currently expanded into Annex C.5).
DEV13	Void
DEV14	The Device SHALL operate with an eUICC comprising a default file system (i.e., with no Enabled Profile) as described in section 3.4.1. In particular, neither the modem nor the eUICC SHALL be shut down.

DEV15	It SHALL be possible for the End User to perform a Device Reset without affecting the status of the eUICC. (The Device MAY also allow the End User to perform a combined Device Reset and eUICC Memory Reset.)

Table 58: Device requirements

C.2 Requirements for Companion Device Scenarios

Interface between the Primary Device and the Companion Device

The implementation of the interface between the Primary Device and the Companion Device is optional and Device Manufacturer specific. The OCF Easy Setup specification [i2] provide a potential implementation of this interface, for which the following SHOULD be considered:

- The Primary Device and the Companion Device defined in this document act as the Mediator and the Enrollee, respectively, defined in [i2].
- The JSON binding of the interface defined in [i2] limits the maximum size of the data objects as follows. If a Remote SIM Provisioning procedure over this interface uses a data object that exceeds the defined limits, the interface MAY reject relaying the data object:
  - euiccInfo2 (eUICC Information in [i2])
  - deviceInfo (Device Information for RSP in [i2])
  - Activation Code (Activation Code in [i2])
  - Profile Metadata (eSIM Profile Metadata in [i2])
  - Confirmation Code (Confirmation Code in [i2])

NOTE: OCF provides an open source implementation for the OCF Easy Setup specification [i2]: https://gitlab.iotivity.org/iotivity/iotivity-lite/-/tree/easysetup

Secure interaction between the Primary Device and the Companion Device

The LPAd of the Companion Device SHALL support secure capture of the User Intent for the purpose of RSP operations through the Primary Device when the Companion Device has to rely on the Primary Device for UI function. This SHALL further include a secure pairing and secured communication between the Primary and Companion Device. The End User MAY request RSP operations towards the eUICC in the Companion Device using the Primary Device.

A secure point to point proximity link at transport level between the Primary Device and the Companion Device SHALL either be implemented by the Primary and Companion Devices' Device manufacturer(s), or it SHALL be established as follows:

1. The End User connects the Companion Device to a router (e.g., the Primary Device which shares the network) which will assign an address for the Companion Device.

NOTE: This step MAY be performed at any time prior to step 2.

2. The End User uses the LPAd on the Companion Device to generate a HTTPS URL which includes the Companion Device address (e.g., private local IP address) and security information (i.e., 128-bit random secret key).

   In order to achieve the interoperability between different Device manufacturer Devices, the HTTPS URL SHALL be specified as https://hpath/LPA_access_token, where "hpath" is the Companion Device address and "LPA_access_token" is the security information.

3. The LPAd on the Companion Device indicates the HTTPS URL including the Companion Device address and the security information to the Primary Device using one of following non-exhaustive example means:

   - The Companion Device transfers the HTTPS URL to the Primary Device, e.g., using NFC.
   - The Companion Device displays the HTTPS URL which could be input by the End User to the Primary Device.
   - The Companion Device transforms the HTTPS URL into a QR code or bar code so that the Primary Device can scan the code to obtain the HTTPS URL.
   - The Companion Device transfers the HTTPS URL through a wired connection, such as a USB link to the Primary Device.

4. Using the Companion Device address and the security information obtained from the HTTPS URL, a software component (e.g., LPAd) on the Primary Device establishes a HTTPS session with the LPAd on the Companion Device:

   - Firstly, the software component uses the LPA_access_token in step 2 as the PSK to initiate the PSK-TLS connection as defined in RFC 4279 [47] with the LPAd on the Companion Device. During the TLS handshake, the software component in the Primary Device performs mutual authentication with the LPAd on the Companion Device and negotiates the session key.
   - After the TLS connection is established, the software component on the Primary Device sends an HTTP request over the TLS session to the Companion Device to retrieve the UI presentation of the LUId. Upon receiving the HTTP request, the LPAd on the Companion Device sends the HTTP response containing the UI presentation to the Primary Device.

5. The End User uses the UI provided by the software component on the Primary Device to access the LUI on the Companion Device via the HTTPS session to perform the Local Profile Management Operations towards the eUICC in the Companion Device. The LUI on the Companion Device MAY restrict the actions that can be performed from the Primary Device. For example:

   - It MAY not offer the eUICC Memory Reset.
   - It MAY only expose the 'enable' and 'disable' operations.
   - It MAY expose a Profile for enabling only if no Profile is already enabled on the Companion Device.

C.3 General LPA Requirements

LPA functions

There SHALL be at most one instance of the LPAd per active eUICC.

The LPA SHALL support all the functions related to Profile download and Installation via the LPA’s Local Profile Download (LPD) functions as defined in section 3.1.3.

The LPA SHALL support Notifications as defined in sections 3.1.3, 3.5, 5.6.4, 5.7.9, 5.7.10, 5.7.11.

The LPA SHALL support the following Local Profile Management Operations and Remote Profile Management Operations via the LPA's Local User Interface (LUI) function:

- Initiate an RSP session for Profile Download as defined in section 3.2.5 and 3.2.8.
- Initiate an RSP session for RPM download as defined in section 3.2.7 and section 3.2.8.
- Enable a Disabled Profile locally as defined in section 3.2.1 and remotely as defined in section 3.7.3.1.
- Disable an Enabled Profile locally as defined in section 3.2.2 and remotely as defined in section 3.7.3.2.
- Delete a Profile locally as defined in section 3.2.3 and remotely as defined in section 3.7.3.3.
- Query the Profile Metadata and states of Profiles installed on the eUICC locally as defined in section 3.2.4 and remotely as defined in section 3.7.3.4.
- Update the Profile Metadata of Profiles installed on the eUICC as defined in section 3.7.3.5.
- Perform eUICC Memory Reset, as defined in section 3.3.2.
- Perform eUICC Test Memory Reset, as defined in section 3.3.3, if the Device supports Device Test Mode as described hereunder.

The LPA SHOULD support the following Local Profile Management Operation via the LPA's Local User Interface (LUI) function:

- Set/Edit Profile Nicknames associated with installed Profiles as defined in section 3.2.6. If the LPA does not support Set/Edit Nickname, alternative Device-specific methods to distinguish Profiles on the LUI SHOULD be provided by the LPA.
- Set/Edit Default SM-DP+ Address as defined in section 3.3.4. If the LPA does not support Set/Edit Default SM-DP+ Address, alternative Device-specific methods to edit the Default SM-DP+ address(es) SHALL be provided to the End User.

The LPA SHALL allow the End User to enable or disable the Remote Profile Management Operations of the LPA. This SHOULD be enabled by default.

For an Enterprise Capable Device without any Enterprise Profile with Enterprise Rules installed, the LPA SHALL permit the End User to allow or disallow the installation of Enterprise Profiles with Enterprise Rules. The default setting SHOULD be disallowed.

The LPA SHALL support retrieval of eUICC Information as defined in section 4.3.

The LPA SHALL support retrieval of Event Records as defined in section 3.6.2.

The LPA MAY support implementation-specific storage, retrieval and update (set/edit) of one or more Default SM-DP+ addresses stored on the Device.

The LPAd SHOULD advise the End User when it determines that an RSP operation would fail (or has failed) because network connectivity is not available, or an error occurs. The LPA MAY retry the RSP operation for a period of time as appropriate. The specific means by which the connectivity failure is detected, and the manner in which it is communicated to the End User, are out of scope of this specification.

If the LPAd supports the LPA Proxy, it MAY be possible for the End User to prevent the use of mobile network data for that purpose. The use of mobile network data for this purpose SHALL be allowed by default.

LPAd Functions and Security Protection

The specific mechanisms for securing the operation of the LPAd, ensuring its integrity, and ensuring the privacy and integrity of the data it handles are out of scope of this specification. As appropriate for the class of Device, the proper security level associated with LPAd functions SHOULD be ensured based on industry-proven implementations of:

- A secure boot OS.
- An implementation-dependent software/hardware secure execution environment for capturing, storing and verifying the passcode or biometric input.
- Verification of proper Device manufacturer signature of LPAd related software components.
- Application-level secure pairing and un-pairing methods between Primary and Companion Devices. This MAY be independent of pairing technologies and associated link layer security (e.g., Bluetooth or Wi-Fi).

The Device specific security implementation SHALL:

- Verify the integrity of the LPAd and authorise it to be used.
- Provide access to the trusted LUId user interface only for the authorised LPAd.
- Provide access to the ISD-R of the eUICC only for the authorised LPAd. This restriction to the LPAd SHALL be enforced regardless of any rule stored in the Profile according to GlobalPlatform SEAC [56] which may allow it.
- Restrict access to the LPAd to only those applications and services that are provided by the Device manufacturer to enable the services and functions of the LPAd.
- Protect the LPAd and the data it handles from unauthorised access and modification. Such data includes, but is not limited to, the EID, Activation Code, Confirmation Code, End User credentials for Strong Confirmation, Profile Metadata, Profile Download and Notification payloads, and Event Records.

Depending on the device class, Devices SHALL implement protection mechanisms as shown in the table below.

Device class	Description	Example of Devices
Advanced	Devices with an open operating system where mechanisms such as secure boot and platform signing of applications are available and used to protect the LPA.	Smartphones, Tablets, Laptops, Advanced Wearables
Basic	Devices without possibility to install applications. The attack surface of the LPA is minimal due to the locked	Connected sensors, Simple Wearables, Single use case devices

	down nature of these Devices. Simple mechanisms to ensure that the LPA is not compromised SHALL be taken.	

Table 58a: Device Classes

Where technically feasible, the Device SHALL implement a mechanism allowing the End User to protect the access to the Device and its Profile Management Operations with personal data. Implementation is Device specific. If such a mechanism is implemented:

- The Device SHOULD enforce the mechanism by default, and
- The End User SHOULD be able to enable/disable the mechanism, and
- The End User SHALL be able to configure the personal data.

The Device SHALL provide mechanisms to obtain Strong Confirmation and Simple Confirmation in a Device specific implementation.

As examples, the recommended Strong Confirmation could include:

- Repeating Simple Confirmations, or
- Biometric (e.g., fingerprint) verification, or
- Device passcode verification

Device Test Mode

The Device and LPAd MAY support Device Test Mode. The method of entering Device Test Mode, exiting Device Test Mode, and Device testing functionality that is not related to Remote SIM Provisioning are implementation-specific and out of the scope of this specification.

The LPAd SHALL only provide access to Test Profiles when the Device is operating in Device Test Mode.

When the Device exits Device Test Mode, the LPAd SHALL disable any enabled Test Profile as defined in section 3.2.2 or 3.2.1.

Enterprise

A Device MAY support Enterprise Rules. The support (or non-support) of Enterprise Rules SHALL NOT change during the lifetime of the Device.

The Device SHALL identify itself as an Enterprise-Capable Device in both the TERMINAL CAPABILITY (section 3.4.2) and the DeviceInfo (section 4.2) if and only if it supports the installation and enforcement of Enterprise Rules as described in this specification.

C.4 Support for CAT Mechanisms

Dependent on the deployment, the Devices SHALL support at least the CAT mechanisms (ETSI TS 102 223 [31]) indicated in the table below.

CAT mechanism	LPAd	LPAe with LUIe based on CAT	LPAe with LUIe based on SCWS	LPAe with LUI based on E4E
TERMINAL PROFILE	X	X	X	X
SETUP MENU ENVELOPE (MENU SELECTION) DISPLAY TEXT GET INKEY GET INPUT PLAY TONE SELECT ITEM EVENT DOWNLOAD - User activity EVENT DOWNLOAD - Idle screen available		X		
SET UP EVENT LIST	X	X	X	X
REFRESH with "UICC Reset" or "eUICC Profile State Change"	X	X	X	X
PROVIDE LOCAL INFORMATION (IMEI)		X	X	X
SEND SHORT MESSAGE ENVELOPE (SMS-PP DOWNLOAD)	X	X	X	X
TIMER MANAGEMENT ENVELOPE (TIMER EXPIRATION)		X	X	X
OPEN CHANNEL related to packet data service bearer	X	X	X (1)	X
OPEN CHANNEL related to UICC Server Mode			X (1)	
CLOSE CHANNEL RECEIVE DATA SEND DATA GET CHANNEL STATUS EVENT DOWNLOAD - Data available EVENT DOWNLOAD - Channel status	X	X	X	X
ENVELOPE with tag 'E4'				X
REFRESH with "Application Update" or "File Change Notification"				X
NOTE 1: The Device SHALL support running these 2 BIP channels in parallel.				

Table 59: CAT Mechanisms

NOTE: The table also includes requirements for ES6.

In addition, the following CAT mechanisms SHALL be supported by a Device supporting MEP:

CAT mechanism	LPAd	LPAe with LUIe based on CAT	LPAe with LUIe based on SCWS	LPAe with LUI based on E4E
REFRESH with "eUICC Profile State Change"	X(1)	X	X	X
REFRESH with "Application Update"				X
LSI COMMAND with Proactive session request	X(1) (2)	X	X	X
LSI COMMAND with Platform Reset	X	X	X	X
NOTE 1: This CAT mechanism is optional for MEP-A2 without RPM NOTE 2: This CAT mechanism is optional for MEP-B				

Table 59a: Additional CAT Mechanisms for MEP support

C.5 APDU Access Interface

An NFC Device SHALL support an APDU access interface as described in this section. A Device that is not an NFC Device SHOULD support this interface.

The APDU access interface MAY be provided by a Device component external to the LPA provided that when it is used with an eUICC it satisfies the following requirements. (As an example, the Device may implement the interface using the GlobalPlatform Open Mobile API transport layer [69].)

The APDU access interface SHALL permit authorised Device Applications to send APDUs to the Enabled Profile and receive the responses. Authorised Device Applications SHALL only be able to access the file system, applications, and security domains within the hierarchy of the MNO-SD.

If the APDU access interface is supported in an MEP-Capable Device, the interface of Open Mobile API [69] handling multiple readers SHOULD be implemented to provide simultaneous access to multiple Enabled Profiles.

The Device MAY implement the GlobalPlatform SEAC specification [56]. If so, the Device SHALL authorise Device Applications by retrieving and enforcing Access Rules as specified in the GlobalPlatform SEAC specification [56]. The Access Rules for the Enabled Profile SHALL be stored as part of the Profile. The Device SHALL NOT enforce Access Rules stored in a Disabled Profile.

The APDU access SHALL be implemented using logical channels on the eUICC. When used in an NFC Device, the APDU access interface SHALL NOT provide access to the eUICC basic channel (channel 0).

Annex D Coding of the AIDs for 'Remote SIM Provisioning' (Normative)

The Coding of the AID for ISD-R, ISD-P and ECASD SHALL be as defined in SGP.02 [2].

Annex E List of Identifiers (Informative)

OIDs

The following identifiers for remote provisioning are created under a dedicated OID tree under ISO branch:

- ASN.1 notation: {ISO(1) identified-organization(3) dod(6) internet(1) private(4) enterprise(1)}
- dot notation: 1.3.6.1.4.1
- IOD-IRI notation: /ISO/Identified-Organization/6/1/4/1

The private enterprise numbers may be found under the Internet Assigned Numbers Authority: http://www.iana.org/assignments/enterprise-numbers/enterprise-numbers

EUM Identifiers

Identifier	Uniqueness	Registration Entity
EUM OID	within eSIM the ecosystem	ISO 1.3.6.1.4.1
EIN	within eSIM the ecosystem	GSMA as per SGP.29 [89] or ITU-T as per E.118 [21], depending on the EID assignment scheme (see section 4.3.1)

Table 60: EUM Identifiers

Identifiers on the eUICC

Identifier	Uniqueness	Registration Entity
EID	within the eSIM ecosystem	See Table 60 for EIN EUM for ESIN
ECASD AID	within the eUICC	GSMA ESIM Technical Specification SGP.02 [2]
ISD-R AID	within the eUICC	GSMA ESIM Technical Specification SGP.02 [2]
ISD-P AID	within the eUICC	eUICC within a range defined in GSMA ESIM Technical Specification SGP.02 [2]
ICCID	Global	ITU-T E.118 [21]
ISD-R TAR	within the eUICC	GSMA ESIM Technical Specification SGP.02 [2]
MNO-SD AID	Within the Profile	ETSI TS 101 220 [33]
MNO-SD TAR	Within the Profile	ETSI TS 101 220 (ISD TAR) [33]

Table 61: Identifiers on the eUICC

SM-DP+ Identifier

Identifier	Uniqueness	Registration Entity
SM-DP+ OID	within the eSIM ecosystem	ISO 1.3.6.1.4.1

Table 62: SM-DP+ Identifier

SM-DS Identifier

Identifier	Uniqueness	Registration Entity
SM-DS OID	within the eSIM ecosystem	ISO 1.3.6.1.4.1

Table 63: SM-DS Identifier

MNO Identifiers

Identifier	Uniqueness	Registration Entity
MNO OID	within the eSIM ecosystem	ISO 1.3.6.1.4.1
MCC+MNC (IMSI)	Global	ITU-T for MCC and National Regulators for MNC

Table 64: MNO Identifiers

Annex F Profile Eligibility Check (Informative)

Prior to any Profile download, the Operator or the SM-DP+ verifies if the selected Profile Type is compatible with the targeted Device.

Two types of checking are possible:

- Static eligibility check (SEC): a check based on the static capabilities of the Device and / or the eUICC. These capabilities could be retrieved based on the knowledge of the EID and the TAC. These eUICC capabilities MAY be acquired by various means: information contained in the EID itself, additional tables locally handled by the Operator or communication with an external entity like the EUM. Device capabilities can be retrieved by the Operator based on the TAC. This Static eligibility check is under the responsibility of the Operator; it MAY be done by the SM-DP+ on behalf of the Operator. The means to establish the compatibility of the Profile Type with a Device type and eUICC type is out of scope of this specification.
- Dynamic eligibility check (DEC): a check based on the eUICC Info and / or the Device capabilities signed by the eUICC during Profile Download and Installation procedure. This Dynamic eligibility check is under the responsibility of the SM-DP+ on behalf of the Operator.

The following figure Figure 41: Eligibility Check" describes the global eligibility process depending on the knowledge of the target Device.

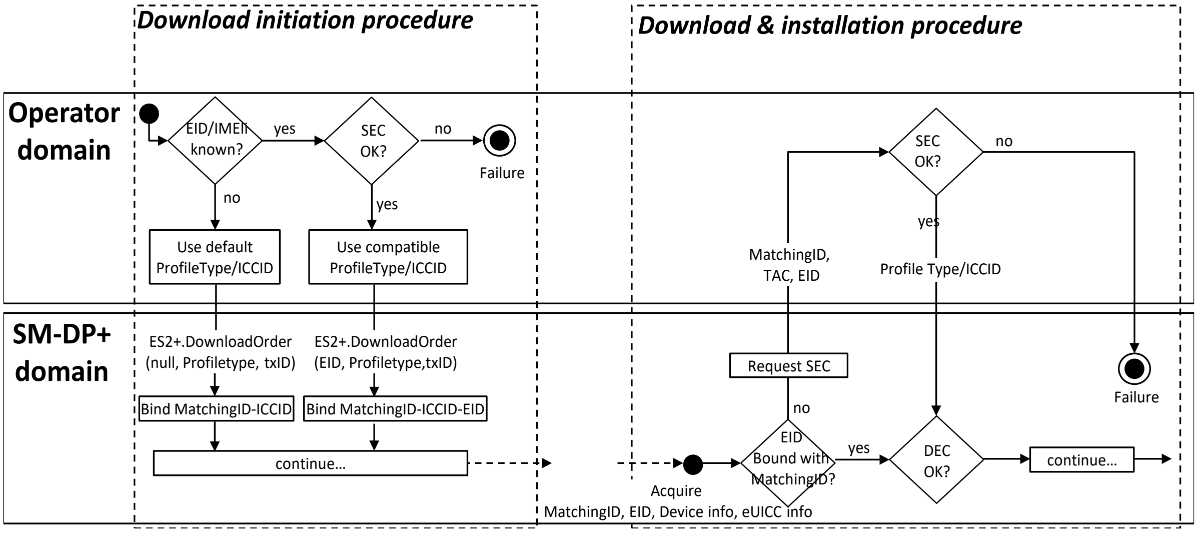

Download initiation procedure
Download & installation procedure
Operator domain
SM-DP+ domain
EID/IMEI known?
yes
SEC OK?
no
Failure
no
yes
Use default ProfileType/ICCID
Use compatible ProfileType/ICCID
ES2+.DownloadOrder (null, Profiletype, txID)
ES2+.DownloadOrder (EID, Profiletype,txID)
Bind MatchingID-ICCID
Bind MatchingID-ICCID-EID
continue…
SEC OK?
no
yes
MatchingID, TAC, EID
Profile Type/ICCID
Request SEC
Failure
no
EID Bound with MatchingID?
yes
DEC OK?
continue…
Acquire MatchingID, EID, Device info, eUICC info

Figure 41: Eligibility Check

Annex G Void

Annex H ASN.1 Definitions (Normative)

```
RSPDefinitions {joint-iso-itu-t(2) international-organizations(23) gsma(146) rsp(1)
spec-version(1) version-three(3)}
DEFINITIONS
AUTOMATIC TAGS
EXTENSIBILITY IMPLIED ::=
BEGIN

IMPORTS Certificate, CertificateList, Time FROM PKIX1Explicit88 {iso(1) identified-
organization(3) dod(6) internet(1) security(5) mechanisms(5) pkix(7) id-mod(0) id-
pkix1-explicit(18)}
SubjectKeyIdentifier FROM PKIX1Implicit88 {iso(1) identified-organization(3) dod(6)
internet(1) security(5) mechanisms(5) pkix(7) id-mod(0) id-pkix1-implicit(19)}
UICCCapability FROM PEDefinitions {joint-iso-itu-t(2) international-
organizations(23) tca(143) euicc-profile(1) spec-version(1) version-three(3)};
-- The UICCCapability import module version is defined in section 4.3

id-rsp OBJECT IDENTIFIER ::= {joint-iso-itu-t(2) international-organizations(23)
gsma(146) rsp(1)}

-- Basic types, for size constraints
Octet1 ::= OCTET STRING(SIZE(1))
Octet4 ::= OCTET STRING (SIZE(4))
Octet8 ::= OCTET STRING (SIZE(8))
Octet16 ::= OCTET STRING (SIZE(16))
OctetTo16 ::= OCTET STRING (SIZE(1..16))
Octet32 ::= OCTET STRING (SIZE(32))

VersionType ::= OCTET STRING(SIZE(3)) -- major/minor/revision version are coded as
binary value on byte 1/2/3, e.g., '02 00 0C' for v2.0.12.
-- If revision is not used (e.g., v2.1), byte 3 SHALL be set to '00'.
Iccid ::= [APPLICATION 26] OCTET STRING (SIZE(10)) -- ICCID as coded in EFiccid,
corresponding tag is '5A'
RemoteOpId ::= [2] INTEGER {installBoundProfilePackage(1)}
TransactionId ::= OCTET STRING (SIZE(1..16))

-- Definition of OIDs
id-rsp-cert-objects OBJECT IDENTIFIER ::= {id-rsp 2}
   -- value 0 in id-rsp-cert-objects was assigned in SGP.22 v2.x
   -- #SupportedOnlyBeforeV3.0.0#

id-rspRole OBJECT IDENTIFIER ::= {id-rsp-cert-objects 1}

-- Definition of OIDs for role identification in certificates
id-rspRole-ci OBJECT IDENTIFIER ::= {id-rspRole 0}
id-rspRole-ciSubCa OBJECT IDENTIFIER ::= {id-rspRole-ci 0}

id-rspRole-eum OBJECT IDENTIFIER ::= {id-rspRole-ciSubCa 0}
id-rspRole-eumSubCa OBJECT IDENTIFIER ::= {id-rspRole-eum 0}
id-rspRole-euicc OBJECT IDENTIFIER ::= {id-rspRole-eumSubCa 0}

id-rspRole-dpSubCa OBJECT IDENTIFIER ::= {id-rspRole-ciSubCa 1}
id-rspRole-dp-tls OBJECT IDENTIFIER ::= {id-rspRole-dpSubCa 0}
id-rspRole-dp-auth OBJECT IDENTIFIER ::= {id-rspRole-dpSubCa 1}
id-rspRole-dp-pb OBJECT IDENTIFIER ::= {id-rspRole-dpSubCa 2}

id-rspRole-dsSubCa OBJECT IDENTIFIER ::= {id-rspRole-ciSubCa 2}
id-rspRole-ds-tls OBJECT IDENTIFIER ::= {id-rspRole-dsSubCa 0}
id-rspRole-ds-auth OBJECT IDENTIFIER ::= {id-rspRole-dsSubCa 1}

-- The following OIDs are used in Variant O and OO Certificates
id-rspRole-euicc-v2 OBJECT IDENTIFIER ::= {id-rspRole 1}
id-rspRole-eum-v2 OBJECT IDENTIFIER ::= {id-rspRole 2}
id-rspRole-dp-tls-v2 OBJECT IDENTIFIER ::= {id-rspRole 3}
id-rspRole-dp-auth-v2 OBJECT IDENTIFIER ::= {id-rspRole 4}
id-rspRole-dp-pb-v2 OBJECT IDENTIFIER ::= {id-rspRole 5}
id-rspRole-ds-tls-v2 OBJECT IDENTIFIER ::= {id-rspRole 6}
```

```
id-rspRole-ds-auth-v2 OBJECT IDENTIFIER ::= {id-rspRole 7}

-- Definition of OIDs for RSP-specific extensions in Certificates
-- #SupportedFromV3.0.0#
id-rsp-extensions OBJECT IDENTIFIER ::= {id-rsp-cert-objects 2}
id-rsp-extension-permitted-eins OBJECT IDENTIFIER ::= { id-rsp-extensions 0}

-- The extnValue field of the id-rsp-extension-permitted-eins extension SHALL be of
type PermittedEins:
PermittedEins ::= SEQUENCE OF PrintableString


PprIds ::= BIT STRING {-- Definition of Profile Policy Rules identifiers
  pprUpdateControl(0), -- defines how to update PPRs via ES6
  ppr1(1), -- Indicator for PPR1 'Disabling of this Profile is not allowed'
  ppr2(2) -- Indicator for PPR2 'Deletion of this Profile is not allowed'
}

OperatorId ::= SEQUENCE {
  mccMnc OCTET STRING (SIZE(3)), -- MCC&MNC coded as 3GPP TS 24.008
  gid1 OCTET STRING OPTIONAL, -- referring to content of EF GID1 (file identifier
'6F3E') in 3GPP TS 31.102 [54]
  gid2 OCTET STRING OPTIONAL  -- referring to content of EF GID2 (file identifier
'6F3F') in 3GPP TS 31.102 [54]
}

RpmConfiguration ::= SEQUENCE { -- #SupportedForRpmV3.0.0#
  managingDpList [0] SEQUENCE OF SEQUENCE {
    managingDpOid [0] OBJECT IDENTIFIER, -- Managing SM-DP+ OID
    rpmType [1] RpmType OPTIONAL,
    tagList [APPLICATION 28] OCTET STRING OPTIONAL
  },
  pollingAddress [1] UTF8String OPTIONAL, -- Tag '81'
  allowedCiPKId [2] SubjectKeyIdentifier OPTIONAL, -- eSIM CA RootCA PKID that is
allowed for managing SM-DP+s
  profileOwnerOid [3] OBJECT IDENTIFIER
}

RpmType ::= BIT STRING{
  enable(0), disable(1), delete(2), listProfileInfo(3), contactPcmp(4)
}

LocalisedTextMessage ::= SEQUENCE { -- #SupportedFromV3.0.0#
  languageTag UTF8String DEFAULT "en", -- language tag as defined by RFC 5646
  message UTF8String
}

LprConfiguration ::= SEQUENCE { -- #SupportedForLpaProxyV3.0.0#
  pcmpAddress [0] UTF8String,
  dpiEnable [1] UTF8String OPTIONAL,
  triggerLprOnEnableProfile [2] NULL OPTIONAL
}

CertificateChain ::= SEQUENCE OF Certificate -- #SupportedFromV3.0.0#

EnterpriseConfiguration ::= SEQUENCE { -- #SupportedForEnterpriseV3.0.0#
  enterpriseOid [0] OBJECT IDENTIFIER,
  enterpriseName [1] UTF8String (SIZE(0..64)),
  enterpriseRules [2] SEQUENCE {
    enterpriseRuleBits [0] BIT STRING {
      referenceEnterpriseRule (0),
      priorityEnterpriseProfile (1),
      onlyEnterpriseProfilesCanBeInstalled (2)
    },
    numberOfNonEnterpriseProfiles [1] INTEGER -- that can be Enabled
  } OPTIONAL
}
```

```
OPENTYPE ::= CLASS {
   &typeId OBJECT IDENTIFIER,
   &Type
}

VendorSpecificExtension ::= SEQUENCE OF SEQUENCE { -- #SupportedFromV2.4.0#
   vendorOid [0] OPENTYPE.&typeId, -- OID of the vendor who defined this specific
extension
   vendorSpecificData [1] OPENTYPE.&Type
}

DeviceChangeConfiguration ::= CHOICE { -- #SupportedForDcV3.0.0#
   requestToDp [0] SEQUENCE {
      smdpAddressForDc UTF8String, -- SM-DP+ address that processes the Device
Change
      allowedCiPKId SubjectKeyIdentifier OPTIONAL, -- PKID allowed for the SM-DP+
address that processes the Device Change
      eidRequired NULL OPTIONAL, -- the EID of the new Device is required
      tacRequired NULL OPTIONAL -- the TAC of the new Device is required
   },
   usingStoredAc [1] SEQUENCE {
      activationCodeForDc UTF8String (SIZE(0..255)), -- Activation Code for Device
Change of this Profile
      deleteOldProfile NULL OPTIONAL -- deletion of this Profile is required before
providing the Activation code to the new Device
   }
}

BoundProfilePackage ::= [54] SEQUENCE { -- Tag 'BF36'
   initialiseSecureChannelRequest [35] InitialiseSecureChannelRequest, -- Tag
'BF23'
   firstSequenceOf87 [0] SEQUENCE OF [7] OCTET STRING, -- sequence of '87' TLVs
   sequenceOf88 [1] SEQUENCE OF [8] OCTET STRING, -- sequence of '88' TLVs
   secondSequenceOf87 [2] SEQUENCE OF [7] OCTET STRING OPTIONAL, -- sequence of
'87' TLVs
   sequenceOf86 [3] SEQUENCE OF [6] OCTET STRING -- sequence of '86' TLVs
}

-- Definition of Profile Installation Result
ProfileInstallationResult ::= [55] SEQUENCE { -- Tag 'BF37'
   profileInstallationResultData [39] ProfileInstallationResultData,
   euiccSignPIR EuiccSign
}

ProfileInstallationResultData ::= [39] SEQUENCE { -- Tag 'BF27'
   transactionId[0] TransactionId, -- The TransactionID generated by the SM-DP+
   notificationMetadata[47] NotificationMetadata,
   smdpOid OBJECT IDENTIFIER, -- SM-DP+ OID (value from CERT.DPpb.SIG)
   finalResult [2] CHOICE {
      successResult SuccessResult,
      errorResult ErrorResult
   }
}

EuiccSign ::= [APPLICATION 55] OCTET STRING -- Tag '5F37', eUICCs signature

SuccessResult ::= SEQUENCE {
   aid [APPLICATION 15] OCTET STRING (SIZE (5..16)), -- AID of ISD-P
   ppiResponse OCTET STRING -- contains (multiple) 'EUICCResponse' of the Profile
Package Interpreter as defined in [5]
}

ErrorResult ::= SEQUENCE {
   bppCommandId BppCommandId,
   errorReason ErrorReason,
   ppiResponse OCTET STRING OPTIONAL -- contains (multiple) 'EUICCResponse' of the
Profile Package Interpreter as defined in [5]
}
```

```
BppCommandId ::= INTEGER {
   initialiseSecureChannel(0),
   configureISDP(1),
   storeMetadata(2),
   storeMetadata2(3),
   replaceSessionKeys(4),
   loadProfileElements(5)
}

ErrorReason ::= INTEGER {
   incorrectInputValues(1),
   invalidSignature(2),
   invalidTransactionId(3),
   unsupportedCrtValues(4),
   unsupportedRemoteOperationType(5),
   unsupportedProfileClass(6),
   bspStructureError(7),
   bspSecurityError(8),
   installFailedDueToIccidAlreadyExistsOnEuicc(9),
   installFailedDueToInsufficientMemoryForProfile(10),
   installFailedDueToInterruption(11),
   installFailedDueToPEProcessingError (12),
   installFailedDueToDataMismatch(13),
   testProfileInstallFailedDueToInvalidNaaKey(14),
   pprNotAllowed(15),
   enterpriseProfilesNotSupported(17), -- #SupportedForEnterpriseV3.0.0#
   enterpriseRulesNotAllowed(18), -- #SupportedForEnterpriseV3.0.0#
   enterpriseProfileNotAllowed(19), -- #SupportedForEnterpriseV3.0.0#
   enterpriseOidMismatch(20), -- #SupportedForEnterpriseV3.0.0#
   enterpriseRulesError(21), -- #SupportedForEnterpriseV3.0.0#
   enterpriseProfilesOnly(22), -- #SupportedForEnterpriseV3.0.0#
   lprNotSupported(23), -- #SupportedForLpaProxyV3.0.0#
   unknownTlvInMetadata(26), -- #SupportedFromV3.0.0#
   installFailedDueToUnknownError(127)
}

RpmPackage ::= SEQUENCE OF RpmCommand -- #SupportedForRpmV3.0.0#

RpmCommand ::= SEQUENCE {
   continueOnFailure [0] NULL OPTIONAL,
   rpmCommandDetails CHOICE {
      enable [1] SEQUENCE {iccid [APPLICATION 26] Iccid},
      disable [2] SEQUENCE {iccid [APPLICATION 26] Iccid},
      delete [3] SEQUENCE {iccid [APPLICATION 26] Iccid},
      listProfileInfo [4] ListProfileInfo,
      updateMetadata [5] SEQUENCE {
         iccid [APPLICATION 26] Iccid,
         updateMetadataRequest UpdateMetadataRequest
      },
      contactPcmp [6] SEQUENCE {
         iccid [APPLICATION 26] Iccid,
         dpiRpm UTF8String OPTIONAL
      }
   }
}

LoadRpmPackageResult ::= [68] CHOICE { -- Tag 'BF44' #SupportedForRpmV3.0.0#
   loadRpmPackageResultSigned LoadRpmPackageResultSigned,
   loadRpmPackageResultNotSigned LoadRpmPackageResultNotSigned
}

LoadRpmPackageResultSigned ::= SEQUENCE {
   loadRpmPackageResultDataSigned LoadRpmPackageResultDataSigned,
   euiccSignRPR EuiccSign
}

LoadRpmPackageResultDataSigned ::= SEQUENCE {
   transactionId [0] TransactionId,
   notificationMetadata[47] NotificationMetadata,
   smdpOid OBJECT IDENTIFIER, -- SM-DP+ OID (value from CERT.DPauth.SIG)
   finalResult [2] CHOICE {
      rpmPackageExecutionResult SEQUENCE OF RpmCommandResult,
      loadRpmPackageErrorCodeSigned LoadRpmPackageErrorCodeSigned
   }
}

RpmCommandResult ::= SEQUENCE { -- #SupportedForRpmV3.0.0#
   iccid [APPLICATION 26] Iccid OPTIONAL, -- SHALL be present, except for 
listProfileInfoResult and rpmProcessingTerminated
   rpmCommandResultData CHOICE {
      enableResult [49] EnableProfileResponse, -- ES10c.EnableProfile
      disableResult [50] DisableProfileResponse, -- ES10c.DisableProfile
      deleteResult [51] DeleteProfileResponse, -- ES10c.DeleteProfile
      listProfileInfoResult [45] ProfileInfoListResponse, -- ES10c.GetProfilesInfo
      updateMetadataResult [42] UpdateMetadataResponse, -- ES6.UpdateMetadata
      contactPcmpResult [0] ContactPcmpResponse,
      rpmProcessingTerminated INTEGER {
         resultSizeOverflow (1),
         unknownOrDamagedCommand (2),
         interruption (3),
         commandsWithRefreshExceeded (4),
         commandAfterContactPcmp (5),
         commandPackageTooLarge (6)
      }
   }
}

ContactPcmpResponse ::= CHOICE {
   contactPcmpResponseOk SEQUENCE {
      pcmpAddress UTF8String
   },
   contactPcmpResponseError INTEGER {
      profileNotEnabled(2),
      commandError(7),
      noLprConfiguration(13),
      undefinedError(127)}
}

LoadRpmPackageResultNotSigned ::= SEQUENCE {
   transactionId [0] TransactionId,
   loadRpmPackageErrorCodeNotSigned LoadRpmPackageErrorCodeNotSigned
}

LoadRpmPackageErrorCodeSigned ::= INTEGER { invalidSignature(2), 
invalidTransactionId(5), undefinedError(127)}

LoadRpmPackageErrorCodeNotSigned ::= INTEGER { noSession(4), undefinedError(127)}

DeleteNotificationForDc ::= SEQUENCE {
   notificationMetadata NotificationMetadata,
   euiccNotificationSignature EuiccSign
}

DeviceInfo ::= SEQUENCE {
   tac Octet4,
   deviceCapabilities DeviceCapabilities,
   imei Octet8 OPTIONAL,
   preferredLanguages SEQUENCE OF UTF8String OPTIONAL, -- 
#DeviceInfoExtensibilitySupported#
   deviceTestMode NULL OPTIONAL, -- #DeviceInfoExtensibilitySupported# if present 
the Device is operating in Device Test Mode
   lpaRspCapability LpaRspCapability OPTIONAL -- #DeviceInfoExtensibilitySupported#
}
```

```
DeviceCapabilities ::= SEQUENCE { -- Highest fully supported release for each
definition
  -- The device SHALL set all the capabilities it supports
   gsmSupportedRelease VersionType OPTIONAL,
   utranSupportedRelease VersionType OPTIONAL,
   cdma2000onexSupportedRelease VersionType OPTIONAL,
   cdma2000hrpdSupportedRelease VersionType OPTIONAL,
   cdma2000ehrpdSupportedRelease VersionType OPTIONAL,
   eutranEpcSupportedRelease VersionType OPTIONAL,
   contactlessSupportedRelease VersionType OPTIONAL,
   rspCrlSupportedVersion VersionType OPTIONAL, -- #SupportedOnlyBeforeV3.0.0#
   nrEpcSupportedRelease VersionType OPTIONAL, --
#DeviceInfoExtensibilitySupported#
   nr5gcSupportedRelease VersionType OPTIONAL, --
#DeviceInfoExtensibilitySupported#
   eutran5gcSupportedRelease VersionType OPTIONAL, --
#DeviceInfoExtensibilitySupported#
   lpaSvn VersionType OPTIONAL, -- #DeviceInfoExtensibilitySupported# provided for
information only
   catSupportedClasses CatSupportedClasses OPTIONAL, --
#DeviceInfoExtensibilitySupported#
   euiccFormFactorType EuiccFormFactorType OPTIONAL --
#DeviceInfoExtensibilitySupported#
}

CatSupportedClasses ::= BIT STRING {
    a(0), b(1), c(2), d(3), e(4), f(5), g(6), h(7), i(8), j(9),
    k(10), l(11), m(12), n(13), o(14), p(15), q(16), r(17), s(18), t(19),
    u(20), v(21), w(22), x(23), y(24), z(25), aa(26), ab(27), ac(28), ad(29)
}

-- Definition of EuiccFormFactorType
EuiccFormFactorType ::= INTEGER {
   removableEuicc (0), -- eUICC can be removed
   nonRemovableEuicc (1) -- eUICC cannot be removed
}

-- Definition of LpaRspCapability
LpaRspCapability ::= BIT STRING {
   crlStaplingV3Support (0),
   certChainV3Support (1),
   apduApiSupport (2),
   enterpriseCapableDevice (3),
   lpaProxySupport (4),
   signedSmdsResponseV3Support (5),
   euiccCiUpdateSupport (6),
   eventCheckingSupport (7),
   pushServiceSupport (8),
   pendingOperationAlertingSupport (9)
}

EUICCInfo1 ::= [32] SEQUENCE { -- Tag 'BF20'
   lowestSvn [2] VersionType,
   euiccCiPKIdListForVerification [9] SEQUENCE OF SubjectKeyIdentifier, -- List of
eSIM CA RootCA Public Key Identifiers supported on the eUICC for signature
verification
   euiccCiPKIdListForSigning [10] SEQUENCE OF SubjectKeyIdentifier, -- List of eSIM
CA RootCA Public Key Identifier supported on the eUICC for signature creation that
can be verified by a certificate chain Variant O
   euiccCiPKIdListForSigningV3 [17] SEQUENCE OF SubjectKeyIdentifier OPTIONAL, --
#SupportedFromV3.0.0# List of eSIM CA RootCA Public Key Identifiers supported on
the eUICC for signature creation that can be verified by a certificate chain
different from Variant O.
   euiccRspCapability [8] EuiccRspCapability OPTIONAL, -- #MandatoryFromV3.0.0#
   highestSvn [19] VersionType OPTIONAL -- #SupportedFromV3.0.0#
}

EUICCInfo2 ::= [34] SEQUENCE { -- Tag 'BF22'
```

```
   baseProfilePackageVersion [1] VersionType,  -- Base eUICC Profile package
version supported
   lowestSvn [2] VersionType,
   euiccFirmwareVersion [3] VersionType,       -- eUICC Firmware version
   extCardResource [4] OCTET STRING,     -- Extended Card Resource Information
according to ETSI TS 102 226
   uiccCapability [5] UICCCapability,
   ts102241Version [6] VersionType OPTIONAL,
   globalplatformVersion [7] VersionType OPTIONAL, -- #MandatoryFromV3.0.0#
   euiccRspCapability [8] EuiccRspCapability,
   euiccCiPKIdListForVerification [9] SEQUENCE OF SubjectKeyIdentifier, -- List of
eSIM CA RootCA Public Key Identifiers supported on the eUICC for signature
verification
   euiccCiPKIdListForSigning [10] SEQUENCE OF SubjectKeyIdentifier, -- List of eSIM
CA RootCA Public Key Identifier supported on the eUICC for signature creation that
can be verified by a certificate chain Variant O
   euiccCategory [11] INTEGER {
      other(0),
      basicEuicc(1),
      mediumEuicc(2),
      contactlessEuicc(3)
   } OPTIONAL, -- #MandatoryFromV3.0.0#
   forbiddenProfilePolicyRules [25] PprIds OPTIONAL, -- Tag '99'
   ppVersion VersionType, -- Protection Profile version
   sasAcreditationNumber UTF8String (SIZE(0..64)),
   certificationDataObject [12] CertificationDataObject OPTIONAL, --
#MandatoryFromV3.0.0#
   treProperties [13] BIT STRING {
      isDiscrete(0),
      isIntegrated(1),
      usesRemoteMemory(2) -- refers to the usage of remote memory protected by
                  -- the Remote Memory Protection Function described in SGP.21 [4]
   } OPTIONAL, -- #MandatoryFromV3.0.0#
   treProductReference [14] UTF8String OPTIONAL, -- #SupportedFromV3.0.0#
Platform_Label as defined in GlobalPlatform DLOA specification [57]
   additionalProfilePackageVersions [15] SEQUENCE OF VersionType OPTIONAL, --
#SupportedFromV3.0.0#
   lpaMode [16] LpaMode OPTIONAL, -- #MandatoryFromV3.0.0# active LPA
   euiccCiPKIdListForSigningV3 [17] SEQUENCE OF SubjectKeyIdentifier OPTIONAL, --
#SupportedFromV3.0.0# List of eSIM CA RootCA Public Key Identifiers supported on
the eUICC for signature creation that can be verified by a certificate chain
different from Variant O.
   additionalEuiccInfo [18] OCTET STRING (SIZE(0..32)) OPTIONAL,   --
#SupportedFromV3.0.0# EUM specific eUICC information
   highestSvn [19] VersionType OPTIONAL -- #SupportedFromV3.0.0#
}

-- Definition of EuiccRspCapability
EuiccRspCapability ::= BIT STRING {
   additionalProfile(0), -- at least one more Profile can be installed
   loadCrlSupport(1), -- #SupportedOnlyBeforeV3.0.0# Support for ES10b.LoadCRL
   rpmSupport(2), -- Remote Profile Management
   testProfileSupport (3), -- support for test profile
   deviceInfoExtensibilitySupport (4),  -- #SupportedFromV2.2.2# support for ASN.1
extensibility in the Device Info
   serviceSpecificDataSupport (5), -- #SupportedFromV2.4.0# support for Service
Specific Data in the Profile Metadata
   hriServerAddressSupport (6), -- #SupportedFromV3.0.0# support for storing HRI
server address
   serviceProviderMessageSupport (7), -- #SupportedFromV3.0.0# Service Provider
message is allowed within Profile metadata
   lpaProxySupport (8), -- #SupportedForLpaProxyV3.0.0# support for LPA Proxy
   enterpriseProfilesSupport (9), -- #SupportedForEnterpriseV3.0.0# support for
enterprise profiles
   serviceDescriptionSupport (10), -- #SupportedFromV3.0.0# support for storing
Service Description
   deviceChangeSupport (11), -- #SupportedFromV3.0.0# support for Device change
   encryptedDeviceChangeDataSupport (12), -- #SupportedFromV3.0.0# support for
encrypted Device Change data in Device Change response
   estimatedProfileSizeIndicationSupport (13), -- #SupportedFromV3.0.0# support for
including estimated profile size
   profileSizeInProfilesInfoSupport (14), -- #SupportedFromV3.0.0# support for
profile size in GetProfilesInfo
   crlStaplingV3Support (15), -- #SupportedFromV3.0.0# support for CRL stapling
   certChainV3VerificationSupport (16), -- #SupportedFromV3.0.0# support for
certificate chain verification Variant A, B and C
   signedSmdsResponseV3Support (17), -- #SupportedFromV3.0.0# support for SM-DS
signed response
   euiccRspCapInInfo1 (18), -- #SupportedFromV3.0.0# EUICCInfo1 includes
euiccRspCapability
   osUpdateSupport (19), -- #SupportedFromV3.0.0# support for eUICC OS Update
   cancelForEmptySpnPnSupport (20), -- #SupportedFromV3.0.0# support for cancel
session reasons empty SPN and empty Profile Name
   updateNotifConfigInfoSupport (21), -- #SupportedFromV3.0.0# support for updating
NotificationConfigurationInfo as defined in section 5.4.1
   updateMetadataV3Support (22) -- #SupportedFromV3.0.0# support for the modified
update metadata mechanism defined in section 5.4.1
}

-- Definition of CertificationDataObject
CertificationDataObject ::= SEQUENCE {
   platformLabel UTF8String,      -- Platform_Label as defined in GlobalPlatform
DLOA specification [57]
   discoveryBaseURL UTF8String    -- Discovery Base URL of the SE default DLOA
Registrar as defined in GlobalPlatform DLOA specification [57]
}

-- Definition of LpaMode
LpaMode ::= INTEGER {
   lpad (0), -- LPAd is active
   lpae (1) -- LPAe is active
}

UpdateMetadataRequest ::= [42] SEQUENCE {  -- Tag 'BF2A'
   serviceProviderName [17] UTF8String (SIZE(0..32)) OPTIONAL, -- Tag '91'
   profileName [18] UTF8String (SIZE(0..64)) OPTIONAL, -- Tag '92'
   iconType [19] IconType OPTIONAL, -- Tag '93'
   icon [20] OCTET STRING (SIZE(0..1024)) OPTIONAL, -- Tag '94'
   profilePolicyRules [25] PprIds OPTIONAL, -- Tag '99'
   serviceSpecificDataStoredInEuicc [34] VendorSpecificExtension OPTIONAL, --
#SupportedFromV2.4.0# Tag 'BF22'
   notificationConfigurationInfo [22] SEQUENCE OF
NotificationConfigurationInformation OPTIONAL, -- #SupportedFromV3.0.0# Tag 'B6'
   tagsForDeletion [APPLICATION 28] OCTET STRING OPTIONAL, -- for tagList
#SupportedFromV3.0.0# tag '5C'
   rpmConfiguration [26] RpmConfiguration OPTIONAL, -- #SupportedForRpmV3.0.0# Tag
'BA'
   hriServerAddress [27] UTF8String OPTIONAL, -- #SupportedFromV3.0.0# Tag '9B'
   lprConfiguration [28] LprConfiguration OPTIONAL, -- #SupportedForLpaProxyV3.0.0#
Tag 'BC'
   enterpriseConfiguration [29] EnterpriseConfiguration OPTIONAL, --
#SupportedForEnterpriseV3.0.0# Tag 'BD'
   deviceChangeConfiguration [32] DeviceChangeConfiguration OPTIONAL --
#SupportedForDcV3.0.0# Tag 'BF20'
}

UpdateMetadataResponse ::= [42] INTEGER { -- #SupportedForRpmV3.0.0# Tag 'BF2A'
   ok (0),
   enterpriseConfigurationNotAllowed (6), -- #SupportedForEnterpriseV3.0.0#
   commandError (7),
   pprUpdateInvalidSetting (12),
   invalidRpmConfiguration (14),
   deleteNotAllowed (15),
   undefinedError(127)
}
```

```
--Definition of data objects for InitialiseSecureChannel Request
InitialiseSecureChannelRequest ::= [35] SEQUENCE { -- Tag 'BF23'
   remoteOpId RemoteOpId, -- Remote Operation Type Identifier (value SHALL be set
to installBoundProfilePackage)
   transactionId [0] TransactionId, -- The TransactionID generated by the SM-DP+
   controlRefTemplate[6] IMPLICIT ControlRefTemplate, -- Control Reference Template
(Key Agreement). Current specification considers a subset of CRT specified in
GlobalPlatform Card Specification Amendment F [13] section 6.5.2.3 for the Mutual
Authentication Data Field
   smdpOtpk [APPLICATION 73] OCTET STRING, ---otPK.DP.KA in accordance with
GlobalPlatform Card Specification Amendment F [13] section 6.5.2.3 for ePK.OCE.KA,
tag '5F49'
   smdpSign [APPLICATION 55] OCTET STRING -- SM-DP's signature, tag '5F37'
}

ControlRefTemplate ::= SEQUENCE {
   keyType[0] Octet1, -- Key type according to GlobalPlatform Card Specification
[8] Table 11-16, Tag '80'
   keyLen[1] Octet1, --Key length in number of bytes. Tag '81'
   hostId[4] OctetTo16 -- Host ID value , Tag '84'
}

--Definition of data objects for ConfigureISDPRequest
ConfigureISDPRequest ::= [36] SEQUENCE { -- Tag 'BF24'
   dpProprietaryData [24] DpProprietaryData OPTIONAL -- Tag 'B8'
}

DpProprietaryData ::= SEQUENCE { -- maximum size including tag and length field:
128 bytes
   dpOid OBJECT IDENTIFIER -- OID in the tree of the SM-DP+ that created the
Profile
   -- additional data objects defined by the SM-DP+ MAY follow
}

StoreMetadataRequest ::= [37] SEQUENCE { -- Tag 'BF25'
   iccid Iccid,
   serviceProviderName [17] UTF8String (SIZE(0..32)), -- Tag '91'
   profileName [18] UTF8String (SIZE(0..64)), -- Tag '92' (corresponds to 'Short
Description' defined in SGP.21 [2])
   iconType [19] IconType OPTIONAL, -- Tag '93' (JPG or PNG)
   icon [20] OCTET STRING (SIZE(0..1024)) OPTIONAL, -- Tag '94' (Data of the icon.
Size 64 x 64 pixel. This field SHALL only be present if iconType is present)
   profileClass [21] ProfileClass DEFAULT operational, -- Tag '95'
   notificationConfigurationInfo [22] SEQUENCE OF
NotificationConfigurationInformation OPTIONAL,
   profileOwner [23] OperatorId OPTIONAL, -- Tag 'B7'
   profilePolicyRules [25] PprIds OPTIONAL, -- Tag '99'
   serviceSpecificDataStoredInEuicc [34] VendorSpecificExtension OPTIONAL, --
#SupportedFromV2.4.0# Tag 'BF22'
   serviceSpecificDataNotStoredInEuicc [35] VendorSpecificExtension OPTIONAL, --
#SupportedFromV2.4.0# Tag 'BF23'
   rpmConfiguration [26] RpmConfiguration OPTIONAL, -- #SupportedForRpmV3.0.0# Tag
'BA'
   hriServerAddress [27] UTF8String OPTIONAL, -- #SupportedFromV3.0.0# Tag '9B'
   serviceProviderMessage [30] LocalisedTextMessage OPTIONAL, --
#SupportedFromV3.0.0# Tag 'BE'
   lprConfiguration [28] LprConfiguration OPTIONAL, -- #SupportedForLpaProxyV3.0.0#
Tag 'BC'
   enterpriseConfiguration [29] EnterpriseConfiguration OPTIONAL, --
#SupportedForEnterpriseV3.0.0# Tag 'BD'
   serviceDescription [31] ServiceDescription OPTIONAL, -- #SupportedFromV3.0.0#
Tag '9F1F'
   deviceChangeConfiguration [32] DeviceChangeConfiguration OPTIONAL, --
#SupportedForDcV3.0.0# Tag 'BF20'
   estimatedProfileSize [33] INTEGER OPTIONAL -- #SupportedFromV3.0.0# Tag '9F21'
}
```

```
NotificationEvent ::= BIT STRING {
   notificationInstall(0),
   notificationLocalEnable(1),
   notificationLocalDisable(2),
   notificationLocalDelete(3),
   notificationRpmEnable(4), -- #SupportedForRpmV3.0.0#
   notificationRpmDisable(5), -- #SupportedForRpmV3.0.0#
   notificationRpmDelete(6), -- #SupportedForRpmV3.0.0#
   loadRpmPackageResult(7) -- #SupportedForRpmV3.0.0#
}

NotificationConfigurationInformation ::= SEQUENCE {
   profileManagementOperation NotificationEvent,
   notificationAddress UTF8String -- FQDN to forward the notification
}

ServiceDescription ::= BIT STRING { -- 1: service is on, 0: service is off
#SupportedFromV3.0.0#
   voice (0), -- Operator-provided voice service
   data (1) -- Operator-provided data service
}

-- Definition of request message for command ReplaceSessionKeys
ReplaceSessionKeysRequest ::= [38] SEQUENCE { -- tag 'BF26'
-- The new initial MAC chaining value
   initialMacChainingValue OCTET STRING,
-- New session key value for encryption/decryption (PPK-ENC)
   ppkEnc OCTET STRING,
-- New session key value of the session key C-MAC computation/verification (PPK-
MAC)
   ppkCmac OCTET STRING
}

ISDRProprietaryApplicationTemplate ::= [PRIVATE 0] SEQUENCE { -- Tag 'E0'
   lowestSvn [2] VersionType,
   euiccConfiguration BIT STRING {
      lpaeUsingCatSupported(0), -- LPA in the eUICC using Card Application Toolkit
      lpaeUsingScwsSupported(1), -- LPA in the eUICC using Smartcard Web Server
      enabledProfile(2), -- eUICC contains an Enabled Profile
      lpaeUsingE4ESupported(3) -- LPA in the eUICC using 'E4' ENVELOPEs
   } OPTIONAL -- #MandatoryFromV3.0.0#
}

LpaeActivationRequest ::= [66] SEQUENCE { -- Tag 'BF42'
   lpaeOption BIT STRING {
      activateCatBasedLpae(0), -- LPAe with LUIe based on CAT
      activateScwsBasedLpae(1) -- LPAe with LUIe based on SCWS
   }
}

LpaeActivationResponse ::= [66] SEQUENCE { -- Tag 'BF42'
   lpaeActivationResult INTEGER {ok(0), notSupported(1)}
}

EuiccConfiguredDataRequest ::= [60] SEQUENCE {  -- Tag 'BF3C'
}

EuiccConfiguredDataResponse ::= [60] SEQUENCE {  -- Tag 'BF3C'
   defaultDpAddress UTF8String OPTIONAL,  -- Default SM-DP+ address
   rootDsAddress UTF8String,  -- Root SM-DS address
   additionalRootDsAddresses SEQUENCE OF UTF8String OPTIONAL, --
#SupportedFromV3.0.0#
   allowedCiPKId SubjectKeyIdentifier OPTIONAL, -- #SupportedFromV3.0.0# PKID
allowed for the Default SM-DP+
   ciList SEQUENCE OF SEQUENCE {     -- #SupportedFromV3.0.0#
      ciPKId SubjectKeyIdentifier,  -- List of eSIM CA RootCA public key identifiers
supported
      ciName UTF8String             -- on the eUICC together with a readable name
   } OPTIONAL
}

SetDefaultDpAddressRequest ::= [63] SEQUENCE { -- Tag 'BF3F'
   defaultDpAddress UTF8String, -- Default SM-DP+ address as an FQDN
   allowedCiPKId SubjectKeyIdentifier OPTIONAL -- #SupportedFromV3.0.0# PKID
allowed for the Default SM-DP+
}

SetDefaultDpAddressResponse ::= [63] SEQUENCE { -- Tag 'BF3F'
   setDefaultDpAddressResult INTEGER {
   ok (0),
   unsupportedCiPKId(8), -- #SupportedFromV3.0.0#
   undefinedError (127)}
}

PrepareDownloadRequest ::= [33] SEQUENCE { -- Tag 'BF21'
   smdpSigned2 SmdpSigned2,                    -- Signed information
   smdpSignature2 [APPLICATION 55] OCTET STRING,      -- tag '5F37'
   hashCc Octet32 OPTIONAL, -- Hash of confirmation code
   smdpCertificate Certificate    -- CERT.DPpb.SIG
}

SmdpSigned2 ::= SEQUENCE {
   transactionId [0] TransactionId,      -- The TransactionID generated by the SMDP+
   ccRequiredFlag BOOLEAN, -- Indicates if the Confirmation Code is required
   bppEuiccOtpk [APPLICATION 73] OCTET STRING OPTIONAL,      -- otPK.EUICC.KA
already used for binding the BPP, tag '5F49'
   rpmPending NULL OPTIONAL -- #SupportedForRpmV3.0.0#
}

PrepareDownloadResponse ::= [33] CHOICE { -- Tag 'BF21'
   downloadResponseOk PrepareDownloadResponseOk,
   downloadResponseError PrepareDownloadResponseError
}

PrepareDownloadResponseOk ::= SEQUENCE {
   euiccSigned2 EUICCSigned2,           -- Signed information
   euiccSignature2 [APPLICATION 55] OCTET STRING      -- tag '5F37'
}

EUICCSigned2 ::= SEQUENCE {
   transactionId [0] TransactionId,
   euiccOtpk [APPLICATION 73] OCTET STRING,           -- otPK.EUICC.KA, tag '5F49'
   hashCc Octet32 OPTIONAL,              -- Hash of confirmation code
   additionalInformation VendorSpecificExtension OPTIONAL -- #SupportedFromV3.0.0#
}

PrepareDownloadResponseError ::= SEQUENCE {
   transactionId [0] TransactionId,
   downloadErrorCode DownloadErrorCode
}

DownloadErrorCode ::= INTEGER {invalidCertificate(1), invalidSignature(2),
noSession(4), invalidTransactionId(5), undefinedError(127)}

GetEuiccChallengeRequest ::= [46] SEQUENCE { -- Tag 'BF2E'
}

GetEuiccChallengeResponse ::= [46] SEQUENCE { -- Tag 'BF2E'
   euiccChallenge Octet16  -- random eUICC challenge
}

GetEuiccInfo1Request ::= [32] SEQUENCE { -- Tag 'BF20'
}

GetEuiccInfo2Request ::= [34] SEQUENCE { -- Tag 'BF22'
}
```

```
ListNotificationRequest ::= [40] SEQUENCE { -- Tag 'BF28'
   profileManagementOperation [1] NotificationEvent OPTIONAL
}

ListNotificationResponse ::= [40] CHOICE { -- Tag 'BF28'
   notificationMetadataList SEQUENCE OF NotificationMetadata,
   listNotificationsResultError INTEGER {undefinedError(127)}
}

NotificationMetadata ::= [47] SEQUENCE { -- Tag 'BF2F'
   seqNumber [0] INTEGER,
   profileManagementOperation [1] NotificationEvent, -- Only one bit SHALL be set
to 1
   notificationAddress UTF8String, -- FQDN to forward the notification
   iccid Iccid OPTIONAL
}

RetrieveNotificationsListRequest ::= [43] SEQUENCE { -- Tag 'BF2B'
   searchCriteria CHOICE {
      seqNumber [0] INTEGER,
      profileManagementOperation [1] NotificationEvent
   } OPTIONAL
}

RetrieveNotificationsListResponse ::= [43] CHOICE { -- Tag 'BF2B'
   notificationList SEQUENCE OF PendingNotification,
   notificationsListResultError INTEGER { undefinedError(127)}
}

PendingNotification ::= CHOICE {
   profileInstallationResult [55] ProfileInstallationResult, -- tag 'BF37'
   otherSignedNotification OtherSignedNotification,
   loadRpmPackageResultSigned [1] LoadRpmPackageResultSigned
}

OtherSignedNotification ::= SEQUENCE {
   tbsOtherNotification NotificationMetadata,
   euiccNotificationSignature EuiccSign,
   euiccCertificate Certificate, -- eUICC Certificate (CERT.EUICC.SIG)
   nextCertInChain Certificate, -- The certificate certifying the eUICC Certificate
   otherCertsInChain [1] CertificateChain OPTIONAL -- #SupportedFromV3.0.0# Other
Certificates in the eUICC certificate chain, if any
}

NotificationSentRequest ::= [48] SEQUENCE { -- Tag 'BF30'
   seqNumber [0] INTEGER
}

NotificationSentResponse ::= [48] SEQUENCE { -- Tag 'BF30'
   deleteNotificationStatus INTEGER {ok(0), nothingToDelete(1),
undefinedError(127)}
}

AuthenticateServerRequest ::= [56] SEQUENCE { -- Tag 'BF38'
   serverSigned1 ServerSigned1,                   -- Signed information
   serverSignature1 [APPLICATION 55] OCTET STRING,    -- tag '5F37'
   euiccCiPKIdToBeUsed SubjectKeyIdentifier OPTIONAL, -- eSIM CA RootCA Public Key
Identifier to be used; MAY also have zero length
   serverCertificate Certificate, -- RSP Server Certificate CERT.XXauth.SIG
   ctxParams1 CtxParams1,
   otherCertsInChain [1] CertificateChain OPTIONAL, -- #SupportedFromV3.0.0# The
remaining part of the CERT.XXauth.SIG certificate chain (if any)
   crlList [2] SEQUENCE OF CertificateList OPTIONAL -- #SupportedFromV3.0.0# as
specified in RFC 5280
}

ServerSigned1 ::= SEQUENCE {
   transactionId [0] TransactionId,            -- The Transaction ID generated by
the RSP Server
   euiccChallenge [1] Octet16,           -- The eUICC Challenge
   serverAddress [3] UTF8String,  -- The RSP Server address as an FQDN
   serverChallenge [4] Octet16,          -- The RSP Server Challenge
   sessionContext [5] SessionContext OPTIONAL, -- #SupportedFromV3.0.0#
   serverRspCapability [6] ServerRspCapability OPTIONAL -- #SupportedFromV3.0.0#
}

CtxParams1 ::= CHOICE {
   ctxParamsForCommonAuthentication[0] CtxParamsForCommonAuthentication,
   ctxParamsForDeviceChange [1] CtxParamsForDeviceChange,
   ctxParamsForProfileRecovery [2] CtxParamsForProfileRecovery,
   ctxParamsForPushServiceRegistration [3] CtxParamsForPushServiceRegistration
-- New contextual data objects MAY be defined for extensibility.
}

CtxParamsForCommonAuthentication ::= SEQUENCE {
   matchingId [0] UTF8String OPTIONAL, -- The MatchingId could be the Activation
code token or EventID or empty
   deviceInfo [1] DeviceInfo, -- The Device information
   operationType [2] OperationType DEFAULT {profileDownload}, --
#SupportedFromV3.0.0#
   iccid Iccid OPTIONAL, -- ICCID, tag '5A' #SupportedFromV3.0.0#
   matchingIdSource [3] MatchingIdSource OPTIONAL, -- #SupportedFromV3.0.0#
   vendorSpecificExtension [4] VendorSpecificExtension OPTIONAL --
#SupportedFromV3.0.0#
}

CtxParamsForDeviceChange ::= SEQUENCE { -- #SupportedForDcV3.0.0#
   iccid Iccid,
   deviceInfo [1] DeviceInfo,
   targetEidValue [APPLICATION 26] Octet16 OPTIONAL,
   targetTacValue [2] Octet4 OPTIONAL,
   vendorSpecificExtension [3] VendorSpecificExtension OPTIONAL
}

CtxParamsForProfileRecovery ::= SEQUENCE { -- #SupportedForDcV3.0.0#
   iccid Iccid,
   deviceInfo [1] DeviceInfo,
   vendorSpecificExtension [2] VendorSpecificExtension OPTIONAL
}

CtxParamsForPushServiceRegistration ::= SEQUENCE { --
#SupportedForPushServiceV3.0.0#
   selectedPushService [0] OBJECT IDENTIFIER,
   pushToken [1] UTF8String
}

MatchingIdSource ::= CHOICE {
   none [0] NULL,
   activationCode [1] NULL,
   smdsOid [2] OBJECT IDENTIFIER
}

OperationType ::= BIT STRING {
   profileDownload(0),
   rpm(1)
}

-- Records information agreed along the session
SessionContext ::= SEQUENCE {
   serverSvn [0] VersionType, -- RSP Server SVN (provided for information only)
   crlStaplingV3Used [1] BOOLEAN, -- Indicates CRLs were attached to the RSP Server
response
   euiccCiPKIdToBeUsedV3 [2] SubjectKeyIdentifier OPTIONAL,
   supportedPushServices [3] SEQUENCE OF OBJECT IDENTIFIER OPTIONAL
}
```

```
-- Definition of ServerRspCapability
ServerRspCapability ::= BIT STRING {
   crlStaplingV3Support (0), -- support for CRL stapling
   eventListSigningV3Support (1), -- support for Event Record signing
   pushServiceV3Support (2), -- support for Push Service
   cancelForEmptySpnPnSupport (3)
}

AuthenticateServerResponse ::= [56] CHOICE { -- Tag 'BF38'
   authenticateResponseOk [0] AuthenticateResponseOk,
   authenticateResponseError [1] AuthenticateResponseError
}

AuthenticateResponseOk ::= SEQUENCE {
   euiccSigned1 EuiccSigned1,           -- Signed information
   euiccSignature1 [APPLICATION 55] OCTET STRING,     --EUICC_Sign1, tag 5F37
   euiccCertificate Certificate,  -- eUICC Certificate (CERT.EUICC.SIG)
   nextCertInChain Certificate,   -- The Certificate certifying the eUICC
Certificate
   otherCertsInChain [0] CertificateChain OPTIONAL -- #SupportedFromV3.0.0# Other
Certificates in the eUICC certificate chain, if any
}

EuiccSigned1 ::= SEQUENCE {
   transactionId [0] TransactionId,
   serverAddress [3] UTF8String, -- The RSP Server address as an FQDN
   serverChallenge [4] Octet16,   -- The RSP Server Challenge
   euiccInfo2 [34] EUICCInfo2,
   ctxParams1 CtxParams1
}

AuthenticateResponseError ::= SEQUENCE {
   transactionId [0] TransactionId,
   authenticateErrorCode AuthenticateErrorCode
}

AuthenticateErrorCode ::= INTEGER {invalidCertificate(1), invalidSignature(2),
unsupportedCurve(3), noSession(4), invalidOid(5), euiccChallengeMismatch(6),
ciPKUnknown(7),
transactionIdError (8), -- #SupportedFromV3.0.0#
missingCrl(9), -- #SupportedFromV3.0.0#
invalidCrlSignature(10), -- #SupportedFromV3.0.0#
revokedCert(11), -- #SupportedFromV3.0.0#
invalidCertOrCrlTime(12), -- #SupportedFromV3.0.0#
invalidCertOrCrlConfiguration(13), -- #SupportedFromV3.0.0#
invalidIccid(14), -- #SupportedForDcV3.0.0#
undefinedError(127)}

CancelSessionRequest ::= [65] SEQUENCE { -- Tag 'BF41'
   transactionId TransactionId,    -- The TransactionID generated by the RSP Server
   reason CancelSessionReason
}

CancelSessionReason ::= INTEGER {
   endUserRejection(0),
   postponed(1),
   timeout(2),
   pprNotAllowed(3),
   metadataMismatch(4),
   loadBppExecutionError(5),
   sessionAborted(16), -- #SupportedFromV3.0.0#
   enterpriseProfilesNotSupported(17), -- #SupportedFromV3.0.0#
   enterpriseRulesNotAllowed(18), -- #SupportedForEnterpriseV3.0.0#
   enterpriseProfileNotAllowed(19), -- #SupportedForEnterpriseV3.0.0#
   enterpriseOidMismatch(20), -- #SupportedForEnterpriseV3.0.0#
   enterpriseRulesError(21), -- #SupportedForEnterpriseV3.0.0#
   enterpriseProfilesOnly(22), -- #SupportedForEnterpriseV3.0.0#
   lprNotSupported(23), -- #SupportedForLpaProxyV3.0.0#
   lprNetworkDataNotAllowed(24), -- #SupportedForLpaProxyV3.0.0#
   emptyProfileOrSpName(25), -- #SupportedFromV3.0.0#
   rpmDisabled(27), -- #SupportedForRpmV3.0.0#
   invalidRpmPackage(28), -- #SupportedFromV3.0.0#
   loadRpmPackageError(29), -- #SupportedForRpmV3.0.0#
   undefinedReason(127)
}

CancelSessionResponse ::= [65] CHOICE { -- Tag 'BF41'
   cancelSessionResponseOk CancelSessionResponseOk,
   cancelSessionResponseError INTEGER {invalidTransactionId(5),
undefinedError(127)}
}

CancelSessionResponseOk ::= SEQUENCE {
   euiccCancelSessionSigned EuiccCancelSessionSigned,        -- Signed information
   euiccCancelSessionSignature [APPLICATION 55] OCTET STRING -- tag '5F37'
}

EuiccCancelSessionSigned ::= SEQUENCE {
   transactionId TransactionId,
   smdpOid OBJECT IDENTIFIER, -- SM-DP+ OID as contained in CERT.DPauth.SIG
   reason CancelSessionReason
}

ProfileInfoListRequest ::= [45] SEQUENCE { -- Tag 'BF2D'
   searchCriteria [0] CHOICE {
      isdpAid [APPLICATION 15] OctetTo16, -- AID of the ISD-P, tag '4F'
      iccid Iccid, -- ICCID, tag '5A'
      profileClass [21] ProfileClass -- Tag '95'
   } OPTIONAL,
   tagList [APPLICATION 28] OCTET STRING OPTIONAL -- tag '5C'
}

ListProfileInfo ::= [5] SEQUENCE {
   searchCriteria [0] CHOICE {
      iccid Iccid,
      profileOwnerOid [0] OBJECT IDENTIFIER
   },
   tagList [APPLICATION 28] OCTET STRING OPTIONAL
}

-- Definition of ProfileInfoListResponse
ProfileInfoListResponse ::= [45] CHOICE { -- Tag 'BF2D'
   profileInfoListOk SEQUENCE OF ProfileInfo,
   profileInfoListError ProfileInfoListError
}

ProfileInfo ::= [PRIVATE 3] SEQUENCE { -- Tag 'E3'
   iccid Iccid OPTIONAL,
   isdpAid [APPLICATION 15] OctetTo16 OPTIONAL, -- AID of the ISD-P containing the
Profile, tag '4F'
   profileState [112] ProfileState OPTIONAL, -- Tag '9F70'
   profileNickname [16] UTF8String (SIZE(0..64)) OPTIONAL, -- Tag '90'
   serviceProviderName [17] UTF8String (SIZE(0..32)) OPTIONAL, -- Tag '91'
   profileName [18] UTF8String (SIZE(0..64)) OPTIONAL, -- Tag '92'
   iconType [19] IconType OPTIONAL, -- Tag '93'
   icon [20] OCTET STRING (SIZE(0..1024)) OPTIONAL, -- Tag '94',
   profileClass [21] ProfileClass OPTIONAL, -- Tag '95'
   notificationConfigurationInfo [22] SEQUENCE OF
NotificationConfigurationInformation OPTIONAL, -- Tag 'B6'
   profileOwner [23] OperatorId OPTIONAL, -- Tag 'B7'
   dpProprietaryData [24] DpProprietaryData OPTIONAL, -- Tag 'B8'
   profilePolicyRules [25] PprIds OPTIONAL, -- Tag '99'
   serviceSpecificDataStoredInEuicc [34] VendorSpecificExtension OPTIONAL, --
#SupportedFromV2.4.0# Tag 'BF22'
```

```
   rpmConfiguration [26] RpmConfiguration OPTIONAL, -- #SupportedForRpmV3.0.0# Tag
'BA'
   hriServerAddress [27] UTF8String OPTIONAL, -- #SupportedFromV3.0.0# Tag '9B'
   lprConfiguration [28] LprConfiguration OPTIONAL, -- #SupportedForLpaProxyV3.0.0#
Tag 'BC'
   enterpriseConfiguration [29] EnterpriseConfiguration OPTIONAL,
-- #SupportedForEnterpriseV3.0.0# Tag 'BD'
   serviceDescription [31] ServiceDescription OPTIONAL, -- #SupportedFromV3.0.0#
Tag '9F1F'
   deviceChangeConfiguration [32] DeviceChangeConfiguration OPTIONAL, --
#SupportedForDcV3.0.0# Tag 'BF20'
   enabledOnEsimPort [36] INTEGER OPTIONAL, -- #SupportedForMEPV3.0.0# Tag '9F24'
   profileSize [37] INTEGER OPTIONAL -- #SupportedFromV3.0.0# Tag '9F25'
}

IconType ::= INTEGER {jpg(0), png(1)}
ProfileState ::= INTEGER {disabled(0), enabled(1)}
ProfileClass ::= INTEGER {test(0), provisioning(1), operational(2)}
ProfileInfoListError ::= INTEGER {
   incorrectInputValues(1),
   profileChangeOngoing (11), -- #SupportedForRpmV3.0.0#
   undefinedError(127)
}

EnableProfileRequest ::= [49] SEQUENCE { -- Tag 'BF31'
   profileIdentifier CHOICE {
      isdpAid [APPLICATION 15] OctetTo16, -- AID, tag '4F'
      iccid Iccid -- ICCID, tag '5A'
   },
   refreshFlag BOOLEAN, -- indicating whether REFRESH is required
   targetEsimPort  INTEGER OPTIONAL-- #SupportedForMEPV3.0.0#
}

EnableProfileResponse ::= [49] SEQUENCE { -- Tag 'BF31'
   enableResult INTEGER {
      ok(0),
      iccidOrAidNotFound(1),
      profileNotInDisabledState(2),
      disallowedByPolicy(3),
      wrongProfileReenabling(4),
      catBusy(5),
      disallowedByEnterpriseRule(6), -- #SupportedForEnterpriseV3.0.0#
      commandError(7),  -- #SupportedFromV3.0.0#
      disallowedForRpm(9),    -- #SupportedForRpmV3.0.0#
      noEsimPortAvailable(10), -- #SupportedForMEPV3.0.0# and
                               -- #SupportedForRpmV3.0.0#
      undefinedError(127)
   },
   targetEsimPort INTEGER OPTIONAL -- #SupportedForMEPV3.0.0#
}

DisableProfileRequest ::= [50] SEQUENCE { -- Tag 'BF32'
   profileIdentifier CHOICE {
      isdpAid [APPLICATION 15] OctetTo16, -- AID, tag '4F'
      iccid Iccid -- ICCID, tag '5A'
   },
   refreshFlag BOOLEAN -- indicating whether REFRESH is required
}

DisableProfileResponse ::= [50] SEQUENCE { -- Tag 'BF32'
   disableResult INTEGER {
      ok(0),
      iccidOrAidNotFound(1),
      profileNotInEnabledState(2),
      disallowedByPolicy(3),
      catBusy(5),
      commandError(7),  -- #SupportedFromV3.0.0#
      disallowedForRpm(10),    -- #SupportedForRpmV3.0.0#
      undefinedError(127)
   }
}

DeleteProfileRequest ::= [51] CHOICE { -- Tag 'BF33'
   isdpAid [APPLICATION 15] OctetTo16, -- AID, tag '4F'
   iccid Iccid -- ICCID, tag '5A'
}

DeleteProfileResponse ::= [51] SEQUENCE { -- Tag 'BF33'
   deleteResult INTEGER {
      ok(0),
      iccidOrAidNotFound(1),
      profileNotInDisabledState(2),
      disallowedByPolicy(3),
      disallowedInTestMode(4),  -- #SupportedFromV3.0.0#
      commandError(7),  -- #SupportedFromV3.0.0#
      undefinedError(127)
   }
}

EuiccMemoryResetRequest ::= [52] SEQUENCE { -- Tag 'BF34'
   resetOptions [2] BIT STRING {
      deleteOperationalProfiles(0),
      deleteFieldLoadedTestProfiles(1),
      resetDefaultSmdpAddress(2),
      deletePreLoadedTestProfiles(3), -- #SupportedFromV3.0.0#
      deleteProvisioningProfiles(4)} -- #SupportedFromV3.0.0#
      -- setting bits 0, 1, 3 and 4 wipes all Profiles
}

EuiccMemoryResetResponse ::= [52] SEQUENCE { -- Tag 'BF34'
   resetResult INTEGER {ok(0), nothingToDelete(1), catBusy(5), undefinedError(127)}
}

GetEuiccDataRequest ::= [62] SEQUENCE { -- Tag 'BF3E'
   tagList [APPLICATION 28] Octet1  -- tag '5C', the value SHALL be set to '5A'
}

GetEuiccDataResponse ::= [62] SEQUENCE { -- Tag 'BF3E'
   eidValue [APPLICATION 26] Octet16  -- tag '5A'
}

-- Definition of Profile Nickname Information
SetNicknameRequest ::= [41] SEQUENCE { -- Tag 'BF29'
   iccid Iccid,
   profileNickname [16] UTF8String (SIZE(0..64))
}

SetNicknameResponse ::= [41] SEQUENCE { -- Tag 'BF29'
   setNicknameResult INTEGER {ok(0), iccidNotFound (1), undefinedError(127)}
}

GetRatRequest ::= [67] SEQUENCE { -- Tag 'BF43'
   -- No input data
}

GetRatResponse ::= [67] SEQUENCE { -- Tag 'BF43'
   rat RulesAuthorisationTable
}

RulesAuthorisationTable ::= SEQUENCE OF ProfilePolicyAuthorisationRule
ProfilePolicyAuthorisationRule ::= SEQUENCE {
   pprIds PprIds,
   allowedOperators SEQUENCE OF OperatorId,
   pprFlags BIT STRING {consentRequired(0)}
}
```

```
AlertData ::= [74] CHOICE { -- Tag 'BF4A' #SupportedFromV3.0.0#
   metadataUpdateEnabledProfile [0] MetadataUpdateEnabledProfile,
   pendingOperationAlert [1] ServerWithPendingOperation
}

MetadataUpdateEnabledProfile ::= SEQUENCE {
   iccid Iccid OPTIONAL,
   tagList [APPLICATION 28] OCTET STRING -- tag '5C'
}

ServerWithPendingOperation ::= CHOICE {
      pollingAddress [0] NULL,
      rootSmds [1] NULL,
      defaultSmdp [2] NULL,
      explicitAddress [3] UTF8String
}

VerifySmdsResponseRequest ::= [69] SEQUENCE { -- Tag 'BF45' #SupportedFromV3.0.0#
   smdsSigned2 [1] SmdsSigned2, -- must have tag [1]
   smdsSignature2 [APPLICATION 55] OCTET STRING
}

SmdsSigned2 ::= SEQUENCE {
   transactionId [0] TransactionId,
   requestSpecificData CHOICE {
      eventList [0] SEQUENCE {
         eventEntries [1] SEQUENCE OF EventRecordV3,
         ecId [2] OCTET STRING(SIZE(16..32)) OPTIONAL, --
#SupportedForEventCheckingV3.0.0# Event Checking ID
         pushServiceRefreshTime [3] GeneralizedTime OPTIONAL --
#SupportedForPushServiceV3.0.0# date and time to re-register a Push Token to the
SM-DS
      },
      pushServiceRegistrationResult [1] SEQUENCE {
         pushServiceRefreshTime [3] GeneralizedTime OPTIONAL --
#SupportedForPushServiceV3.0.0# date and time to re-register a Push Token to the
SM-DS
      }
   }
}

EventRecordV3 ::= SEQUENCE { -- #SupportedFromV3.0.0#
   eventId UTF8String,
   rspServerAddress UTF8String,
   eventType INTEGER, -- either 1 (for Profile Download) or 2 (for RPM)
   hashedIccids SEQUENCE OF OCTET STRING (SIZE(32)) OPTIONAL, -- hashed ICCID(s)
calculated as either SHA256(ICCID) or SHA256(ICCID|Salt)
   salt OCTET STRING (SIZE(8..16)) OPTIONAL, -- optional salt to be concatenated
with ICCID(s) for hashing
   serviceProviderName [17] UTF8String (SIZE(0..32)) OPTIONAL,
   operatorId [23] OperatorId OPTIONAL
}

VerifySmdsResponseResponse ::= [69] CHOICE {  -- Tag 'BF45' #SupportedFromV3.0.0#
   verifySmdsResponseOk NULL,
   verifySmdsResponseError INTEGER {
      invalidSignature(2),
      noSession(4),
      invalidTransactionId(5),
      undefinedError(127)
   }
}

LoadRpmPackageRequest ::= [68] SEQUENCE { -- #SupportedForRpmV3.0.0# Tag 'BF44'
   smdpSigned3 SmdpSigned3,
   smdpSignature3 [APPLICATION 55] OCTET STRING, -- tag '5F37'
   targetEsimPort INTEGER OPTIONAL
}
```

```
SmdpSigned3 ::= SEQUENCE {
   transactionId [0] TransactionId, -- The TransactionID generated by the SM-DP+
   rpmPackage [1] RpmPackage,
   rpmPending [2] NULL OPTIONAL
}

PrepareDeviceChangeRequest ::= [77] SEQUENCE { -- #SupportedForDcV3.0.0# Tag 'BF4D'
   smdpSigned4 SmdpSigned4, -- Signed information
   smdpSignature4 [APPLICATION 55] OCTET STRING, -- tag '5F37'
   hashCc Octet32 OPTIONAL -- Hash of confirmation code
}

SmdpSigned4 ::= SEQUENCE { -- #SupportedForDcV3.0.0#
   transactionId [0] TransactionId,      -- The TransactionID generated by the SMDP+
   ccRequiredFlag BOOLEAN, -- Indicates if the Confirmation Code is required
   activationCodeForProfileRecovery [1] UTF8String (SIZE(0..255)) OPTIONAL --
presents only in ES9+.AuthenticateClient response for a profileRecoveryRequest
}

PrepareDeviceChangeResponse ::= [77] CHOICE { -- #SupportedForDcV3.0.0# Tag 'BF4D'
   downloadResponseOk PrepareDeviceChangeResponseOk,
   downloadResponseError PrepareDeviceChangeResponseError
}

PrepareDeviceChangeResponseOk ::= SEQUENCE { -- #SupportedForDcV3.0.0#
   euiccSigned3 EUICCSigned3, -- Signed information
   euiccSignature3 [APPLICATION 55] OCTET STRING -- tag '5F37'
}

EUICCSigned3 ::= SEQUENCE { -- #SupportedForDcV3.0.0#
   transactionId [0] TransactionId,
   euiccOtpk [APPLICATION 73] OCTET STRING OPTIONAL, -- otPK.EUICC.KA, tag '5F49'
   hashCc Octet32 OPTIONAL, -- Hash of confirmation code
   additionalInformation VendorSpecificExtension OPTIONAL
}

PrepareDeviceChangeResponseError ::= SEQUENCE { -- #SupportedForDcV3.0.0#
   transactionId [0] TransactionId,
   downloadErrorCode DownloadErrorCode
}

VerifyDeviceChangeRequest ::= [75] CHOICE { -- Tag 'BF4B' #SupportedForDcV3.0.0#
   deviceChangeConfirmation [0] SEQUENCE {
      smdpSigned5 SmdpSigned5, -- Signed information
      smdpSignature5 [APPLICATION 55] OCTET STRING
   },
   profileRecoveryResponse [1] SEQUENCE {
      smdpSigned4 SmdpSigned4, -- Signed information
      smdpSignature4 [APPLICATION 55] OCTET STRING -- tag '5F37'
   }
}

SmdpSigned5 ::= SEQUENCE { -- #SupportedForDcV3.0.0#
   transactionId [0] TransactionId,
   deviceChangeResponse [1] DeviceChangeResponse OPTIONAL
}

DeviceChangeResponse ::= CHOICE {
   deviceChangeData [0] DeviceChangeData,
   encryptedDeviceChangeData [1] EncryptedDeviceChangeData
}

DeviceChangeData ::= SEQUENCE { -- #SupportedForDcV3.0.0#
   iccid Iccid,
   activationCodeForDc [0] UTF8String (SIZE(0..255)),
   deleteOldProfile [1] NULL OPTIONAL, -- Deletion of the installed Profile
required
```

```
   deleteNotificationForDcSupport [2] NULL OPTIONAL, -- Delete Notification for
Device Change supported
   notificationAddress [3] UTF8String OPTIONAL, -- FQDN that processes the Delete
Notification for Device Change
   profileRecoverySupport [4] NULL OPTIONAL,
   profileRecoveryValidityPeriod [5] GeneralizedTime OPTIONAL -- Absolute date and
time for Profile Recovery
}

EncryptedDeviceChangeData ::= SEQUENCE { -- #SupportedForDcV3.0.0#
   controlRefTemplate [6] IMPLICIT ControlRefTemplate,
   smdpOtpk [APPLICATION 73] OCTET STRING, -- okPK.DP.KA
   sequenceOf87 [1] SEQUENCE OF [7] OCTET STRING -- sequence of '87' TLVs
}

VerifyDeviceChangeResponse ::= [75] CHOICE { -- Tag 'BF4B' #SupportedForDcV3.0.0#
   verifyDeviceChangeOk CHOICE {
      verifyDeviceChangeNull [0] NULL,
      verifyDeviceChangeData [1] DeviceChangeData
   },
   verifyDeviceChangeError INTEGER {
      invalidSignature(2),
      noSession(4),
      invalidTransactionId(5),
      unsupportedCrtValues(6),
      invalidData(7),
      profileNotInDisabledState(8),
      undefinedError(127)
   }
}

E4ERequest ::= [PRIVATE 4] CHOICE { -- Tag 'E4'
   startDownload [0] SEQUENCE {
      activationCode [0] UTF8String (SIZE(0..255))
   }, -- Start Download
   confirmDownload [1] SEQUENCE {
      enable [0] NULL OPTIONAL, -- enable Profile after download
      confirmationCode [1] UTF8String OPTIONAL, -- confirmation code
      pinCode [2] UTF8String (SIZE(4..8)) OPTIONAL -- LPAe PIN if used
   }, -- Confirm Download
   listProfiles [2] NULL, -- List Profiles
   enableProfile [3] SEQUENCE {iccid [APPLICATION 26] Iccid}, -- Enable Profile
   disableProfile [4] SEQUENCE {iccid [APPLICATION 26] Iccid},  -- Disable Profile
   deleteProfile [5] SEQUENCE {
      iccid [APPLICATION 26] Iccid,
      pinCode [1] UTF8String (SIZE(4..8)) OPTIONAL -- LPAe PIN if used
   }, -- Delete Profile
   euiccMemReset [6] SEQUENCE {
      pinCode [0] UTF8String (SIZE(4..8)) OPTIONAL -- LPAe PIN if used
   }, -- eUICC Memory Reset
   changeConfirmationPin [7] UTF8String (SIZE(9..17)), -- Change confirmation PIN
   setRpmAllow [8] BOOLEAN, -- Turn on/off Remote Profile Management
   pollRpmPackage [9] SEQUENCE {iccid [APPLICATION 26] Iccid OPTIONAL},
   -- Check for RPM packages for profile with iccid. No iccid means 'Update All'
   confirmRpmPackage [10] SEQUENCE {
      pinCode [0] UTF8String (SIZE(4..8)) OPTIONAL
      -- LPAe PIN, if used, with Strong Confirmation
   }, -- Confirms the pending RpmPackage
   cancelSession[11] NULL
   -- Cancels the pending profile download or RpmPackage execution
}

E4EResponse ::= [PRIVATE 4] SEQUENCE { -- Tag 'E4'
   resultCode [0] E4EResultCode,
   resultData [1] CHOICE {
      startDownloadResponse [0] SEQUENCE {
         serviceProviderName [17] UTF8String (SIZE(0..32)), -- Tag '91'
         profileName [18] UTF8String (SIZE(0..64)), -- Tag '92'
         ccRequired [0] NULL OPTIONAL -- confirmation code required
      },
      listProfilesResponse [3] SEQUENCE OF SEQUENCE {
         iccid [APPLICATION 26] Iccid, -- Profile ICCID
         profileState [112] ProfileState, -- Tag '9F70'
         serviceProviderName [17] UTF8String (SIZE(0..32)), -- Tag '91'
         profileName [18] UTF8String (SIZE(0..64)) -- Tag '92'
         -- the eUICC MAY truncate these names so that the response fits
         -- into one APDU
      },
      pollRpmPackageResponse [4] SEQUENCE {
         rpmPackage [0] RpmPackage, -- RPM Package to be confirmed by user
         rpmPending [1] NULL OPTIONAL -- There are pending RPM Packages after this
      },
      confirmDownloadResponse [5] SEQUENCE {
         iccid [APPLICATION 26] Iccid -- Profile ICCID
      }
   } OPTIONAL
}

E4EResultCode ::= INTEGER {
   success (0),
   errorBusy (1), -- CAT not available due to another operation
   errorComm (2), -- Communication error with server
   errorAuth (3), -- Mutual Authentication Error
   errorNoProfile (4), -- No Profile available for download at SM-DP+
   errorEligibility (5), -- SM-DP+ rejected download due to Eligibility Check
   errorInstall (6), -- Error during Profile installation
   errorPin (7), -- Invalid PIN
   errorProfileRef (8), -- Referenced Profile does not exist
   errorAlreadyEnabled (9), -- Referenced Profile is already enabled
   errorAlreadyDisabled (10), -- Referenced Profile is already disabled
   errorConfirmationCode (11), -- Invalid Confirmation Code,
   errorRpmDisabled (12), -- Cannot pollRpmPackage, RPM is disabled
   errorProfileDoesNotExist (13), -- There is no profile with provided ICCID
   undefinedError (127)
}


RemoteProfileProvisioningRequest ::= [2] CHOICE {  -- Tag 'A2'
   initiateAuthenticationRequest [57] InitiateAuthenticationRequest,  -- Tag 'BF39'
   authenticateClientRequest [59] AuthenticateClientRequest, -- Tag 'BF3B'
   getBoundProfilePackageRequest [58] GetBoundProfilePackageRequest,  -- Tag 'BF3A'
   cancelSessionRequestEs9 [65] CancelSessionRequestEs9, -- Tag 'BF41'
   handleNotification [61] HandleNotification, -- tag 'BF3D'
   confirmDeviceChangeRequest [76] ConfirmDeviceChangeRequest, -- Tag 'BF4C'
   checkEventRequest [70] CheckEventRequest -- Tag 'BF46'
}

RemoteProfileProvisioningResponse ::= [2] CHOICE { -- Tag 'A2'
   initiateAuthenticationResponse [57] InitiateAuthenticationResponse, -- Tag
'BF39'
   authenticateClientResponseEs9 [59] AuthenticateClientResponseEs9, -- Tag 'BF3B'
   getBoundProfilePackageResponse [58] GetBoundProfilePackageResponse, -- Tag
'BF3A'
   cancelSessionResponseEs9 [65] CancelSessionResponseEs9, -- Tag 'BF41'
   authenticateClientResponseEs11 [64] AuthenticateClientResponseEs11, -- Tag
'BF40'
   confirmDeviceChangeResponse [76] ConfirmDeviceChangeResponse, -- Tag 'BF4C'
   checkEventResponse [70] CheckEventResponse -- Tag 'BF46'
}

InitiateAuthenticationRequest ::= [57] SEQUENCE { -- Tag 'BF39'
   euiccChallenge [1] Octet16, -- random eUICC challenge
   smdpAddress [3] UTF8String,
   euiccInfo1 EUICCInfo1,
   lpaRspCapability LpaRspCapability OPTIONAL -- #SupportedFromV3.0.0#
}
```

```
InitiateAuthenticationResponse ::= [57] CHOICE { -- Tag 'BF39'
   initiateAuthenticationOk InitiateAuthenticationOkEs9,
   initiateAuthenticationError INTEGER {
      invalidDpAddress(1),
      euiccVersionNotSupportedByDp(2), -- #SupportedOnlyBeforeV3.0.0#
      ciPKIdNotSupported(3),
      invalidInputData(124), -- #SupportedFromV3.0.0#
      missingInputData(125), -- #SupportedFromV3.0.0#
      functionProviderBusy(126), -- #SupportedFromV3.0.0#
      undefinedError(127) -- #SupportedFromV3.0.0#
   }
}

InitiateAuthenticationOkEs9 ::= SEQUENCE {
   transactionId [0]TransactionId, -- The TransactionID generated by the SM-DP+
   serverSigned1 ServerSigned1, -- Signed information
   serverSignature1 [APPLICATION 55] OCTET STRING, -- Server Sign1, tag '5F37'
   euiccCiPKIdToBeUsed SubjectKeyIdentifier OPTIONAL, -- The CI Public Key to be
used as required by ES10b.AuthenticateServer
   serverCertificate Certificate,
   otherCertsInChain [1] CertificateChain OPTIONAL, -- #SupportedFromV3.0.0#
   crlList [2] SEQUENCE OF CertificateList OPTIONAL -- #SupportedFromV3.0.0# From
RFC 5280
}

AuthenticateClientRequest ::= [59] SEQUENCE {  -- Tag 'BF3B'
   transactionId [0] TransactionId,
   authenticateServerResponse [56] AuthenticateServerResponse, -- This is the
response from ES10b.AuthenticateServer
   deleteNotificationForDc [1] DeleteNotificationForDc -- #SupportedForDcV3.0.0#
Delete Notification for Device Change, see section 4.1.3
}

AuthenticateClientResponseEs9 ::= [59] CHOICE {  -- Tag 'BF3B'
   authenticateClientOk AuthenticateClientOk,
   authenticateClientError INTEGER {
      eumCertificateInvalid(1),
      eumCertificateExpired(2),
      euiccCertificateInvalid(3),
      euiccCertificateExpired(4),
      euiccSignatureInvalid(5),
      matchingIdRefused(6),
      eidMismatch(7),
      noEligibleProfile(8),
      ciPKUnknown(9),
      invalidTransactionId(10),
      insufficientMemory(11),
      ciPKMismatch(12), -- #SupportedFromV3.0.0#
      euiccRspCapabilityHasChanged(13), -- #SupportedFromV3.0.0#
      lpaRspCapabilityHasChanged(14), -- #SupportedFromV3.0.0#
      deviceChangeNotSupported(15), -- #SupportedForDcV3.0.0#
      deviceChangeNotAllowed(16), -- #SupportedForDcV3.0.0#
      iccidUnkwon(17), -- #SupportedForDcV3.0.0#
      invalidInputData(124), -- #SupportedFromV3.0.0#
      missingInputData(125), -- #SupportedFromV3.0.0#
      functionProviderBusy(126), -- #SupportedFromV3.0.0#
      undefinedError(127)
   },
   authenticateClientOkRpm AuthenticateClientOkRpm, -- #SupportedForRpmV3.0.0#
   authenticateClientOkDeviceChange AuthenticateClientOkDeviceChange --
#SupportedForDcV3.0.0#
}

AuthenticateClientOk ::= SEQUENCE {
   transactionId [0] TransactionId,
   profileMetadata [37] StoreMetadataRequest OPTIONAL,
   smdpSigned2 [1] SmdpSigned2 OPTIONAL, -- Signed information
   smdpSignature2 [APPLICATION 55] OCTET STRING OPTIONAL,    -- tag '5F37'
   smdpCertificate Certificate -- CERT.DPpb.SIG
}

AuthenticateClientOkRpm ::= SEQUENCE {
   transactionId [0] TransactionId,
   smdpSigned3 [3] SmdpSigned3,
   smdpSignature3 [APPLICATION 55] OCTET STRING
}

AuthenticateClientOkDeviceChange ::= SEQUENCE {
   transactionId [0] TransactionId,
   smdpSigned4 [1] SmdpSigned4, -- Signed information
   smdpSignature4 [APPLICATION 55] OCTET STRING,      -- tag '5F37'
   serviceProviderMessageForDc [1] LocalisedTextMessage OPTIONAL -- Service
Provider Message For Device Change
}

GetBoundProfilePackageRequest ::= [58] SEQUENCE {  -- Tag 'BF3A'
   transactionId [0] TransactionId,
   prepareDownloadResponse [33] PrepareDownloadResponse
}

GetBoundProfilePackageResponse ::= [58] CHOICE {  -- Tag 'BF3A'
   getBoundProfilePackageOk GetBoundProfilePackageOk,
   getBoundProfilePackageError INTEGER {
      euiccSignatureInvalid(1),
      confirmationCodeMissing(2),
      confirmationCodeRefused(3),
      confirmationCodeRetriesExceeded(4),
      bppRebindingRefused(5),
      downloadOrderExpired(6),
      invalidTransactionId(95),
      invalidInputData(124), -- #SupportedFromV3.0.0#
      missingInputData(125), -- #SupportedFromV3.0.0#
      functionProviderBusy(126), -- #SupportedFromV3.0.0#
      undefinedError(127)
   }
}

GetBoundProfilePackageOk ::= SEQUENCE {
   transactionId [0] TransactionId,
   boundProfilePackage [54] BoundProfilePackage
}

HandleNotification ::= [61] SEQUENCE { -- Tag 'BF3D'
   pendingNotification PendingNotification
}

CancelSessionRequestEs9 ::= [65] SEQUENCE { -- Tag 'BF41'
   transactionId TransactionId,
   cancelSessionResponse  CancelSessionResponse -- data structure defined for
ES10b.CancelSession function
}

CancelSessionResponseEs9 ::= [65] CHOICE { -- Tag 'BF41'
   cancelSessionOk CancelSessionOk,
   cancelSessionError INTEGER {
      invalidTransactionId(1),
      euiccSignatureInvalid(2),
      invalidInputData(124), -- #SupportedFromV3.0.0#
      missingInputData(125), -- #SupportedFromV3.0.0#
      functionProviderBusy(126), -- #SupportedFromV3.0.0#
      undefinedError(127)
   }
}

CancelSessionOk ::= SEQUENCE { -- This function has no output data
```

```
}

AuthenticateClientResponseEs11 ::= [64] CHOICE {  -- Tag 'BF40'
   authenticateClientOk AuthenticateClientOkEs11V2, -- #SupportedOnlyBeforeV3.0.0#
   authenticateClientError INTEGER {
      eumCertificateInvalid(1),
      eumCertificateExpired(2),
      euiccCertificateInvalid(3),
      euiccCertificateExpired(4),
      euiccSignatureInvalid(5),
      eventIdUnknown(6),
      invalidTransactionId(7),
      ciPKUnknown(8), -- #SupportedFromV3.0.0#
      ciPKMismatch(9), -- #SupportedFromV3.0.0#
      euiccRspCapabilityHasChanged(10), -- #SupportedFromV3.0.0#
      lpaRspCapabilityHasChanged(11), -- #SupportedFromV3.0.0#
      pushServiceNotSupport(12), -- #SupportedForPushServiceV3.0.0#
      pushServiceRegistrationNotSupported(13), -- #SupportedForPushServiceV3.0.0#
      invalidInputData(124), -- #SupportedFromV3.0.0#
      missingInputData(125), -- #SupportedFromV3.0.0#
      functionProviderBusy(126), -- #SupportedFromV3.0.0#
      undefinedError(127)
   },
   authenticateClientOkV3 AuthenticateClientOkEs11V3 -- #SupportedFromV3.0.0#
}

AuthenticateClientOkEs11V2 ::= SEQUENCE { -- #SupportedOnlyBeforeV3.0.0#
   transactionId [0] TransactionId,
   eventEntries [1] SEQUENCE OF EventRecord
}

EventRecord ::= SEQUENCE { -- #SupportedOnlyBeforeV3.0.0#
   eventId UTF8String,
   rspServerAddress UTF8String
}

AuthenticateClientOkEs11V3 ::= SEQUENCE {
   transactionId [0] TransactionId,
   smdsSigned2 [1] SmdsSigned2,
   smdsSignature2 [APPLICATION 55] OCTET STRING -- tag '5F37'
}


CheckEventRequest ::= [70] SEQUENCE {  -- #SupportedForEventCheckingV3.0.0# Tag 
'BF46'
   ecId [0] OCTET STRING(SIZE(16..32)), -- Event Checking Identifier
   smdsAddress [1] UTF8String
}

CheckEventResponse ::= [70] CHOICE {  -- #SupportedForEventCheckingV3.0.0# Tag 
'BF46'
   checkEventOk CheckEventOk,
   checkEventError INTEGER {
      invalidDsAddress(1),
      eventCheckingNotSupported(2), -- #SupportedForEventCheckingV3.0.0#
      expiredEcid(3), -- #SupportedForEventCheckingV3.0.0#
      unknownEcid(4), -- #SupportedForEventCheckingV3.0.0#
      invalidInputData(124), -- #SupportedFromV3.0.0#
      missingInputData(125), -- #SupportedFromV3.0.0#
      functionProviderBusy(126), -- #SupportedFromV3.0.0#
      undefinedError(127) -- #SupportedFromV3.0.0#
   }
}

CheckEventOk ::= SEQUENCE {
   isPendingEvent [0] BOOLEAN -- Indicates if an Event Record corresponding to the 
received ECID exists
}
```

```
ConfirmDeviceChangeRequest ::= [76] SEQUENCE { -- #SupportedForDcV3.0.0# Tag 'BF4C'
   transactionId TransactionId,
   prepareDownloadResponse [33] PrepareDownloadResponse
}

ConfirmDeviceChangeResponse ::= [76] CHOICE { -- #SupportedForDcV3.0.0# Tag 'BF4C'
   confirmDeviceChangeOk ConfirmDeviceChangeOk,
   confirmDeviceChangeError INTEGER {
      invalidTransactionId(1),
      euiccSignatureInvalid(2),
      confirmationCodeMissing(3),
      confirmationCodeRefused(4),
      confirmationCodeInvalidMatch(5),
      confirmationCodeRetriesExceeded(6),
      invalidInputData(124),
      missingInputData(125),
      functionProviderBusy(126),
      undefinedError(127)
   }
}

ConfirmDeviceChangeOk ::= SEQUENCE {
   transactionId [0] TransactionId,
   smdpSigned5 SmdpSigned5,
   smdpSignature5 OCTET STRING
}

END
```

Annex I JSON Request Response Examples (Informative)

An example for the "ES9+.InitiateAuthentication" function is shown below:

- HTTP Request (from LPA to SM-DP+):

The following example is in the case where the SM-DP+ supports the v3-specific FQDN as described in section 2.6.6.2.

```
HTTP POST /gsma/rsp2/es9plus/initiateAuthentication HTTP/1.1
Host: rsp3-smdp.example.com
User-Agent: gsma-rsp-lpad
X-Admin-Protocol: gsma/rsp/v2.1.0
Content-Type: application/json;charset=UTF-8
Content-Length: XXX

{
   "euiccChallenge": "ZVVpY2NDaGFsbGVuZ2VFeGFtcGxlQmFzZTY0oUFZuQnNZVE5D",
   "euiccInfo1": "RmVHRnRjR3hsUW1GelpUWTBvVUZadVFuTlpWRTU",
   "smdpAddress": "smdp.example.com",
   "lpaRspCapability": "ODAwMjAzRjg="
}
```

- HTTP Response

```
HTTP/1.1 200 OK
X-Admin-Protocol: gsma/rsp/v2.1.0
Content-Type: application/json;charset=UTF-8
Content-Length: XXX

{
   "header": {
      "functionExecutionStatus": {
         "status": "Executed-Success"
      }
   },
   "transactionId": "0123456789ABCDEF",
   "serverSigned1": "VGhpcyBpcyBub3QgYSByZWFsIHZhbHVl",
   "serverSignature1": "RKNFZsbFVUa05qUm14e",
   "euiccCiPKIdToBeUsed": " BBQAAQIDBAUGBwgJCgsMDQ4PEBESEw==",
   "serverCertificate": "RUU2NTQ0ODQ5NDA0RlpSRUZERA==...",
   "otherCertsInChain": ["q83vASM..."]
}
```

An example for the "ES2+.DownloadOrder" function is shown as follows.

- HTTP Request (from Operator to SM-DP+):

```
HTTP POST /gsma/rsp2/es2plus/downloadOrder HTTP/1.1
Host: smdp.example.com
X-Admin-Protocol: gsma/rsp/v3.0.0
Content-Type: application/json;charset=UTF-8
Content-Length: XXX

{
   "header": {
      "functionRequesterIdentifier": "RequesterID",
      "functionCallIdentifier": "TX-567"
   }
   "eid": "89001567010203040506070809101152",
   "iccid": "8947010000123456784F",
   "profileType": "myProfileType"
}
```

- HTTP Response for a successful execution:

```
HTTP/1.1 200 OK
X-Admin-Protocol: gsma/rsp/v3.0.0
Content-Type: application/json;charset=UTF-8
Content-Length: XXX

{
   "header": {
      "functionExecutionStatus": {
         "status": "Executed-Success"
      }
   },
   "iccid": "8947010000123456784F"
}
```

- HTTP Response for a failed execution:

```
HTTP/1.1 200 OK
X-Admin-Protocol: gsma/rsp/v3.0.0
Content-Type: application/json;charset=UTF-8
Content-Length: XXX

{
   "header": {
      "functionExecutionStatus": {
         "status": "Failed",
         "statusCodeData": {
            "subjectCode": "8.2.5",
            "reasonCode": "3.7",
            "message": "No more Profile"
         }
      }
   }
}
```

An example for the "ES2+.HandleNotification" function is shown as follows:

- HTTP Request (from SM-DP+ to Operator):

```
HTTP POST /gsma/rsp3/es2plus/handleNotification HTTP/1.1
Host: smdp.example.com
X-Admin-Protocol: gsma/rsp/v3.0.0
Content-Type: application/json;charset=UTF-8
Content-Length: XXX

{
   "header": {
      "functionRequesterIdentifier": "RequesterID",
   },
   "eid": "89001567010203040506070809101152",
   "iccid": "8947010000123456784F",
   "profileType": "myProfileType",
   "timeStamp": "2015-12-16T09:30:47Z",
   "notificationEvent": 4,
   "notificationEventStatus": {
      "status": "Executed-Success"
   }
}
```

- HTTP Response for a successful execution:

```
HTTP/1.1 204 No Content
X-Admin-Protocol: gsma/rsp/v3.0.0
```

Annex J Tag allocation (Normative)

This annex lists the tags allocated to data objects that SHALL be used for the definition of the eUICC functions.

Tag	Data name	Value
'BF20'	GetEuiccInfo1Request or EUICCInfo1	32
'BF21'	PrepareDownloadRequest or PrepareDownloadResponse	33
'BF22'	GetEuiccInfo2Request or EUICCInfo2	34
'BF23'	InitialiseSecureChannelRequest	35
'BF24'	ConfigureISDPRequest	36
'BF25'	StoreMetadataRequest	37
'BF26'	ReplaceSessionKeysRequest	38
'BF27'	Reserved	39
'BF28'	ListNotificationRequest or ListNotificationResponse	40
'BF29'	SetNicknameRequest or SetNicknameResponse	41
'BF2A'	UpdateMetadataRequest or UpdateMetadataResponse	42
'BF2B'	RetrieveNotificationsListRequest or RetrieveNotificationsListResponse	43
'BF2D'	ProfileInfoListRequest or ProfileInfoListResponse	45
'BF2E'	GetEuiccChallengeRequest or GetEuiccChallengeResponse	46
'BF2F'	NotificationMetadata	47
'BF30'	NotificationSentRequest or NotificationSentResponse	48
'BF31'	EnableProfileRequest or EnableProfileResponse	49
'BF32'	DisableProfileRequest or DisableProfileResponse	50
'BF33'	DeleteProfileRequest or DeleteProfileResponse	51
'BF34'	EuiccMemoryResetRequest or EuiccMemoryResetResponse	52
'BF35'	Reserved	53
'BF36'	BoundProfilePackage	54
'BF37'	ProfileInstallationResult	55
'BF38'	AuthenticateServerRequest or AuthenticateServerResponse	56
'BF39'	InitiateAuthenticationRequest or InitiateAuthenticationResponse	57
'BF3A'	GetBoundProfilePackageRequest or GetBoundProfilePackageResponse	58
'BF3B'	AuthenticateClientRequest or AuthenticateClientResponseES9	59
'BF3C'	EuiccConfiguredDataRequest or EuiccConfiguredDataResponse	60
'BF3D'	handleNotification	61
'BF3E'	GetEuiccDataRequest or GetEuiccDataResponse	62
'BF3F'	SetDefaultDpAddressRequest or SetDefaultDpAddressResponse	63
'BF40'	AuthenticateClientResponseEs11	64
'BF41'	CancelSessionRequest or CancelSessionResponse or cancelSessionRequestEs9 or cancelSessionResponseEs9	65
'BF42'	LpaeActivationRequest or LpaeActivationResponse	66

Tag	Data name	Value
'BF43'	GetRatRequest or GetRatResponse	67
'BF44'	LoadRpmPackageRequest or LoadRpmPackageResult	68
'BF45'	VerifySmdsRespnseRequest or VerifySmdsResponseResponse	69
'BF46'	CheckEventRequest or CheckEventResponse	70
'BF4A'	AlertData	74
'BF4B'	VerifyDeviceChangeRequest or VerifyDeviceChangeResponse	75
'BF4C'	ConfirmDeviceChangeRequest or ConfirmDeviceChangeResponse	76
'BF4D'	PrepareDeviceChangeRequest or PrepareDeviceChangeResponse	77

Table 66: Tag Allocation

Annex K OID allocation (Informative)

This annex provides some background on the schema of the OID allocation used in this document.

For the purpose of assigning OIDs, a root OID for GSMA was registered within the RSP project.

The value of this root OID is:

joint-iso-itu-t(2) international-organizations(23) gsma(146)

For the purpose of this project, a first node was allocated under this node:

rsp(1)

All OIDs allocated in this version and in version 1.X of this specification belong to the rsp node. OIDs not defined in the ASN.1 (Annex H) are out of scope of this specification, and allocated by their respective owners.

At the time of writing of this specification, Mr. Gary Waite (GSMA) was responsible for the allocation of additional OIDs under the gsma node, i.e., acting as Registration Authority.

Other GSMA projects should use a similar approach: register a project specific node under the gsma node and then define sub-nodes in the project specific documentation.

Within the rsp node, the following schema is used:

rsp(1) – root for the RSP project

spec-version(1) – root for identifying the ASN.1 module of the different versions

version-one(1) – ASN.1 module of version 1.X

version-two(2) – ASN.1 module of version 2.X

version-three(3) – ASN.1 module of version 3.X

… - future ASN.1 modules SHOULD use additional sub-nodes here

cert-objects(2) – root for nodes identifying objects and roles
used in certificates

id-rspExt(0) – root for certificate extensions defined in version 2.x of this specification

id-rspRole(1) – root for roles used in certificates
(see section 2.4a.1.0 for further details)

Annex L DLOA document (Normative)

The DLOA is an XML document as defined in GlobalPlatform DLOA [57].

The following table describes the specific coding and rules of the `Platform_DLOA` used in this specification to represent the certification, evaluation, approval, qualification, or validation granted to an eUICC platform.

Field	Description
`Authority_Label`	This field SHALL contain the DLOA Authority OID in doted notation: '2.23.146.1' (GSMA RSP node).
`LOA_Identifier`	This field SHALL contain the identifier of the LOA assigned by the DLOA Authority.
`LOA_Scope`	This field SHALL contain a string indicating the scope of the certification, evaluation, approval, qualification, or validation covered by the LOA. The string SHALL contain the following: "RSP SGP.22 v<version>/<compliance level>" - <version> SHALL identify the version of specification against which the eUICC platform has been certified. Value SHALL be coded as a string of major/minor/revision values (each on possibly several digits) separated by '. '. If revision is not used, it SHALL be omitted. E.g., "RSP SGP.22 v2.0.12", "RSP SGP.22 v1.0". - <compliance level> value SHALL be "Basic" for this version of specification.
`Platform_Label`	This field SHALL contain a string as the concatenation of the OID of the platform manufacturer and the Unique Identifier of the platform, separated by a '/': "<OID platform manufacturer>/<Unique Identifier of the platform>" - <OID platform manufacturer > = OID in doted notation of the entity being the manufacturer of the eUICC platform. E.g., "1.2.3.4" - <Unique Identifier of the platform> = a value defined by the platform manufacturer (out of scope), but SHALL follow rules defined in GlobalPlatform DLOA [57].
`Issuance_Date`	This field SHALL contain the date of issuance of the related LOA.
`Expiration_Date`	The Expiration_Date SHALL be set with the value defined by the applied compliance process.
`LOA_Url`	This field SHALL contain the URL where the original LOA as issued by the DLOA Authority can be retrieved.
`Signature`	The signature SHALL be done as defined in GlobalPlatform DLOA [57]. In order to limit the cryptographic requirements on Management System side, this specification limits to the signature algorithm 'http://www.w3.org/2001/04/xmldsig-more#ecdsa-sha256', with one of the curves defined in 2.6.5.
NOTE: The OID of the platform manufacturer and the EIN part of the EID represent the same entity. Both values could be used interchangeably to identify the entity.	

Table 67: Platform_DLOA description

NOTE: The implementation details for `Platform_DLOA` will be indicated in SGP.24 [64] when the DLOA service becomes active. The `Application_DLOA` is not used in this specification.

Annex M Configuration for RSP Server, LPA and EUICC (Normative)

EUICC

An eUICC stating conformance to this version of this specification SHALL:

- Set the `EuiccRspCapability.loadCrlSupport` bit to '0'.
- Set the `EuiccRspCapability.deviceInfoExtensibilitySupport` bit to '1'.
- Set the `EuiccRspCapability.serviceProviderMessageSupport` bit to '1'.
- Set the `EuiccRspCapability.crlStaplingV3Support` bit to '1'.
- Set the `EuiccRspCapability.certChainV3VerificationSupport` bit to '1'.
- Set the `EuiccRspCapability.signedSmdsResponseV3Support` bit to '1'.
- Set the `EuiccRspCapability.euiccRspCapInInfo1` bit to '1'.
- Set the `EuiccRspCapability.cancelForEmptySpnPnSupport` bit to '1'.
- Set the `EuiccRspCapability.updateNotifConfigInfoSupport` bit to '1'.
- Set the `EuiccRspCapability.updateMetadataV3Support` bit to '1'.
- Include `euiccCategory` in `EuiccInfo2`.
- Set the `lpaMode` in `EUICCInfo2` corresponding to the active LPA (LPAd or LPAe).
- Set the `EuiccInfo1.lowestSvn`, `EuiccInfo2.lowestSvn`, and `ISDRProprietaryApplicationTemplate.lowestSvn` to v2.1.0.

NOTE: this value is required for interoperability with version 2 RSP servers.

- Set the `EuiccInfo1.highestSvn` and `EuiccInfo2.highestSvn` to v3.0.0.

NOTE: this value is provided for information only.

- Set the `EuiccInfo2.baseProfilePackageVersion` to lowest major and highest minor version of the eUICC Profile Package Specification [5] supported by the eUICC.

NOTE: this value is required for interoperability with version 2 RSP servers.

- Set the `EuiccInfo2.additionalProfilePackageVersions` with the list of additional eUICC Profile Package Specification [5] supported by the eUICC, if any (e.g., 3.2.0, 4.1.0).

An integrated eUICC stating conformance to this version of this specification SHALL:

- Include `treProperties` in `EuiccInfo2`.
- Include `treProductReference` in `EuiccInfo2`.

LPA

An LPA stating conformance to this version of this specification SHALL:

- Set the `LpaRspCapability.crlStaplingV3Support` bit to '1'.
- Set the `LpaRspCapability.certChainV3Support` bit to '1'.
- Set the `LpaRspCapability.signedSmdsResponseV3Support` bit to '1' if there is any SM-DS address configured in the Device or eUICC.
- Set the `DeviceInfo.lpaSvn` to v3.0.0.
- Set the `euiccFormFactorType` in `DeviceInfo`.
- Indicate the followings in Terminal Capability:
  - Metadata update alerting support
  - Enterprise Capable Device

SM-DP+

An SM-DP+ stating conformance to this version of this specification SHALL:

- Set the `SessionContext.serverSvn` to v3.0.0.
- Set the `ServerRspCapability.crlStaplingV3Support` bit to '1'.
- Set the `ServerRspCapability.cancelForEmptySpnPnSupport` bit to '1'.

SM-DS

An SM-DS stating conformance to this version of this specification SHALL:

- Set the `SessionContext.serverSvn` to v3.0.0.
- Set the `ServerRspCapability.crlStaplingV3Support` bit to '1'.
- Set the `ServerRspCapability.eventListSigningV3Support` bit to '1'.
- Support Event Registration and Event Retrieval with HashedIccid(s), Salt, ServiceProviderName, and OperatorId.

Annex N Version Interoperability (Informative)

There is no need to indicate that an eUICC no longer supports version 2 certificate verification because such eUICC would be configured with only a version 3 certificate itself. The version 2 server will see an empty list of euiccCiPKIdListForSigning that will cause ESXX.InitiateAuthentication to fail.

Assumption that should be verified on LPA version 2:

- When an eUICC version 3 is configured with version 3 certificate chain and is inserted in a device with LPA version 2, an RSP Server version 3 will generate ESXX.InitiateAuthentication response with an empty euiccCiPKIdToBeUsed. The LPA version 2 should accept that empty value.

An SM-DP+ (and the Operator) may see IMEIs with differently coded last octets coming from LPAs implementing version 2 or version 3 of this specification.

For HTTP layer:

- Version 3 RSP Server or LPA should be ready to handle an HTTP request or response with a Content-type having no character set encoding information coming from a version 2 peer. In turn a version 2 RSP server or LPA is expected to accept a character set encoding being present.

Annex O Device Change (Informative)

The Service Provider is expected to provide to the SM-DP+ a configuration for the Device Change procedure.

This configuration informs about behaviour requested by the Service Provider upon Device Change requests.

This configuration includes, but is not limited to, the following indications:

- If the SM-DP+ is requested, upon reception of a Common Mutual Authentication with a Device Change request (i.e., ES9+.AuthenticateClient with ctxParamsForDeviceChange), to:
  - call the ES2+.HandleDeviceChangeRequest function, including the ICCID of the Profile with the EID and/or TAC of the new Device (if requested by the `DeviceChangeConfiguration` of the profile),
  - notify the Service Provider with the ES2+.HandleNotification functionafter Common Mutual Authentication if the Device Change request is accepted,
- If the SM-DP+ is requested, upon reception of ES9+.ConfirmDeviceChange with indication of the End User’s confirmation result (e.g., accepted, rejected, no response) to notify the Service Provider with the ES2+.HandleNotification function.

If the SM-DP+ is not requested to call the ES2+.HandleDeviceChangeRequest function, the configuration could include, but is not limited to, information to be provided to the LPAd in the different steps of the Device Change procedure:

- For ES9+.AuthenticateClient response:
  - The Service Provider Message for Device Change, if it has to be delivered by the SM-DP+ to the old Device (i.e., `serviceProviderMessageForDc` in ES9+.AuthenticateClient response),
  - If the SM-DP+ is requested to ask to the End User to enter a Confirmation Code, (i.e., `ccRequiredFlag` set to TRUE in `smdpSigned2`) and the related Confirmation Code to allow the SM-DP+ to verify it,
- For ES9+.ConfirmDeviceChange response:
  - If SM-DP+ has to deliver to the new Device a new Profile or the same Profile as the one currently installed in the old Device (i.e., `deleteOldProfile` in `DeviceChangeResponse`).

The means for the Service Provider to provide a configuration for the Device Change procedure to the SM-DP+ is out of scope of this specification.

Annex P Use of VendorSpecificExtension (informative)

This annex provides examples of ASN.1 type definitions that can be referenced by `VendorSpecificExtension`.

A data object with the following type can be included in `VendorSpecificExtension` that is carried within `euiccSigned3` for Device Change:

```
SecureUserIntent ::= SEQUENCE {
    secureUserIntentCaptured BOOLEAN,
    additionalData OCTET STRING OPTIONAL
}
```

Annex Q Document Management (Informative)

Q.1 Document History

Version	Date	CR N°	Brief Description of Change	Approval Authority	Editor / Company
V1.0	13 January 2016		New PRD Publication	PSMC	Duncan Macadam GSMA
V.1.1	14 April 2016		Minor Change to fix bugs Phase one	PSMC	Yolanda Sanz GSMA
V2.0	July 2016		Major Change to include new features	PSMC	Yolanda Sanz GSMA
V2.1	Jan 2017		Minor Change to fix bugs and maintenance changes	PSM	Yolanda Sanz GSMA
V2.2	31 Aug 2017	CR2455R02	Clarification about mutual authentication	RSPLEN	Yolanda Sanz GSMA
		CR2456R01	Align ASN.1 for BPP not available for new binding		
		CR2457R01	Clarification on the OID allocation		
		CR2458R00	Definition of certificates validity period		
		CR2460R03	Update on CRL following RSPTFTEC#23		
		CR2461R00	Fix EID format in eUICC certificate		
		CR2462R01	DLOA introduction		
		CR2463R01	Clarify ECC support requirements		
		CR2464R01	Fix version on JASON Binding		
		CR2465R00	Clarification on Get Default SM-DP+ address		
		CR2466R01	Bug fix on ES11.InitiateAuthentication		
		CR2467R02	Clarify removal of icon via ES6.UpdateMetadata		
		CR2468R01	Remove Le for STORE DATA on ES6		
		CR2469R00	ESci alignement with SGP.14		
		CR2470R01	Align initial device setup to SGP.21 v2.2		
		CR2471R00	Adding UI to Abbreviations		
		CR2472R01	Change Symantec by GSMA in Certificates description		

		CR2473R03	Verifying Profile state for download retry		
		CR2475R01	SM-XX Verification		
		CR2476R02	Fix JSON pattern for timestamp field		
		CR2477R00	Fix error code in ES12.DeleteEvent		
		CR2479R01	Access to the ISD-R vs. SEAC		
		CR2480R02	Provide more indication on Multi CI support for TLS		
		CR2481R02	ECC reference in TLS		
		CR2482R00	Align ES9+ and ES11.AuthenticateClient errors		
		CR2483R00	Clarification of SIMalliance error code mapping		
		CR2484R01	Clarification of UpdateMetadata operation for PPR		
		CR2487R01	Clarification on ES8+ interface		
		CR2488R00	Fix profileClass in GetProfilesInfo		
		CR2451R02	Update of Profile Package specification reference		
		CR2459R05	Update of Certificates following RSPTFTEC#23		
		CR2485R00	Editorial correction in Sub-procedure Profile Installation		
		CR2489R00	Clarification on ES9+ interface		
		CR2490R01	Clarification on the ES9+ AuthenticateClient		
		CR2491R01	JSON string escaping clarification		
		CR2493R01	DLOA: LOA_Scope and Expiration date definition		
		NA	Various improvements following TF review		
		CR2494R01	Clarification of Profile Owner checking		
		CR2495R00	Editorial correction to terminal capability		
		CR2496R00	TF comments		
		CR2497R01	Gender neutrality		
V2.2.1	29th Nov 2018		Minor minor change to fix minor changes	RSPPLEN	Yolanda Sanz GSMA
V2.2.2	05th June 2020		Minor minor change to fix minor changes	ISAG	Yolanda Sanz GSMA
SGP.22 v2.3	30 June 2021	CR2701R01	Simplified End User confirmation	ISAG	Yolanda Sanz GSMA
		CR2702R00	Removal of mandatory RAT configuration		
		CR2705R00	Clarifications on v2.2.2 Application Notes		

		CR2703R01	Optional Edit Default SM-DP+ Address funtions		
		CR2706R01	SGP.22 Mirror CR for integrated eUICC		
		CR2704R00	Allowing LPA to accept TLS Certificate chain to Public CA		
		CR2710R00	Update for changes in LPA45		
		CR2709R00	Alignment of Discrete eUICC definition with SGP.21 v2.3		
		CR2711R01	treProperties mandatory for integrated		
		CR2708R03	More changes related to SGP.29		
		CR2712R03	References updated		
		CR2714R01	Profile download retry upon temporary errors (mirror to CR3486)		
		CR2715R02	Optional LPAd support for Profile Download with PPRs on Removable eUICCs		
		CR2717R05	Editorial Clarification on disabling a Test Profile		
		CR2718R00	Update version numbers in HTTP headers		
		CR2719R02	References cleanup		
		CR2722R00	More SIMAlliance clean up		
		CR2721R00	Optional support of Brainpool and FRP for TLS		
		CR2724R00	ASN.1 corrections		
		CR2725R00	Allow multiple eUICC Profile Package versions		
V2.4	28 October 2021	CR2801R02	Field-test eUICC	ISAG	Yolanda Sanz GSMA
		CR2802R02	Clarification on outdate ppVersion during oudate interm period		
		CR2803R02	Add Profile Metadata Extensibility		
SGP.22 v3.0	19 October 2022	CR3001R01	eUICC Memory Reset for Provisioning and pre-loaded Test Profiles	ISAG	Denis Praca, Thales
		CR3003R02	eUICC initialization without enabled profile		
		CR3004R03	Alignment of ISD-R selection		
		CR3005R02	Optimization during eUICC initialization		
		CR3006R04	High Resolution Icon		
		CR3007R06	RPM Procedures		
		CR3009R02	SMXX check for incorrect signing CI		
		CR3012R00	High Resolution Icon on ES10c		

		CR3014R02	Unsetting of Policy Rules via Update Metadata		
		CR3015R03	Adding OperationType in ES9+.AuthenticateClient Request		
		CR3016R06	RPM functions		
		CR3017R01	RPM Package		
		CR3018R01	Setting OperationType and ICCID in ctxParams1		
		CR3011R01	Removal of icon via ES6.UpdateMetadata		
		CR3020R01	Version negotiation during device initialisation		
		CR3022R01	APDU access API		
		CR3023R01	Service provider message during Profile download		
		CR3019R01	Introduce server version number in common mutual authentication procedure		
		CR3025R01	Clean some yellow highlighted content in RPM related sections		
		CR3013R07	LPA alerting of Metadata updates		
		CR3026R07	Multi-CI Security		
		CR3027R01	ES2+ and SMDP new Requirements		
		CR3028R02	OS Update Requirements		
		CR3029R01	OS Update capability ELG16		
		CR3030R01	Metadata Update procedure		
		CR3033R00	Update reference for application update refresh		
		CR3038R01	General compliance checking rules		
		CR3042R00	Network attachment in Local Profile Management		
		CR3045R00	Shall be stopped fixed		
		CR3052R01	Add minor comments from Plenary in SGP.22 v2.2		
		CR3044R00	Add missing error codes in ES2+.releaseProfile function		
		CR3031R02	ES2+.RpmOrder		
		CR3032R04	RPM clean up		
		CR3034R01	RSP version numbers (editorial)		
		CR3035R06	RPM Command 'Contact PCMP'		
		CR3039R05	ES10b.CancelSession		
		CR3010R13	Add SubCAs		
		CR3041R04	Enterprise		
		CR3043R02	Add missing error codes binding in asn1		
		CR3046R00	CancelSession reason codes serverAddressMismatch and serverOidMismatch		

		CR3047R01	Clarify calculation of confirmation code hash		
		CR3049R01	Clarify download attempt limits		
		CR3050R03	Clarification of the usage of the Function Requester Identifier field		
		CR3051R01	Clarification of HTTP client retry and reconnection management		
		CR3053R06	Common Cancel Session		
		CR3055R05	Clarification on common mutual authentication, profile download procedure, subject key identifier and minor reference correction		
		CR3056R01	STORE DATA block number count		
		CR3057R01	CancelOrder JSON fix		
		CR3058R04	Unified Notifications		
		CR3060R02	Make catBusy optional		
		CR3062R01	Editorial corrections		
		CR3063R00	Apply CERTPK12 restriction only to server PKIds		
		CR3064R01	Align ES9+ and ES11 AuthenticateClient		
		CR3065R01	TF Comments on SGP.22 v3.0		
		CR3066R04	LPA API		
		CR3067R01	finalProfileStatusIndicator rewording		
		CR3074R00	Fix ASN.1 in RpmPackageResult		
		CR3075R01	Remove Cert in RPM Package Download		
		CR3080R01	No use case to store serviceProviderMessage on eUICC		
		CR3070R02	LPA API support for companion devices		
		CR3061R01	Change RPM type to CHOICE		
		CR3088R01	RPM type twice in Annex H and other fixes		
			Allocation of missing tag values		
		CR3048R02	Download Test Profile only in Device Test Mode		
		CR3059R03	Reference to Sub-procedure Cancel Session		
		CR3071R03	Adding notificationReason in Phase 3 Notifications		
		CR3082R00	Clarify Service Provider vs. Operator		
		CR3083R00	Clarify that RPM always uses refresh		
		CR3084R02	Event alerting mechanism		
		CR3085R00	LPA API PlantUML update		
		CR3086R02	Adding matchingIdSource in ctxParams1		

		CR3087R01	SM-DS operation when no pending Event during CMA		
		CR3089R01	Remove Cert in ES9+.AuthenticateClient for RPM		
		CR3090R02	Restructure RPM package and add RPM notifications		
		CR3091R02	Rename rpmPackageResult to loadRpmPackageResult		
		CR3095R01	Allow option to postpone download with PPRs		
		CR3097R00	Fix references to non-existent section		
		CR3102R02	Correction to GlobalPlatform reference		
		CR3069R04	Integrity protection of SM-DS response		
		CR3076R02	Limit RpmPackage size		
		CR3077R04	Limit RpmPackageResult size		
		CR3081R03	Clear cached data upon RPM Update Metadata		
		CR3092R02	rpmConfiguration in UpdateMetadata, StoreMetadata, and GetProfilesInfo		
		CR3096R01	JSON and ASN.1 binding of ES9+.AuthenticateClient		
		CR3099R02	LPA PCMP Address configuration		
		CR3106R00	End User access to ICCID of Operational Profiles		
		CR3068R05	Advanced activation code		
		CR3094R03	CI public key and credential update for eUICC		
		CR3098R02	Verifying MatchingID of Command Code		
		CR3100R05	Validation of Device Application		
		CR3104R03	Public CA for TLS		
		CR3107R02	Simplify description of RPM package processing		
		CR3108R02	Limit RPM to one command with REFRESH		
		CR3109R03	End User consent on enabling a Profile after its installation		
		CR3110R00	ES2+ HandleNotification renaming		
		CR3111R02	Alignment of SGP22 with SGP21 on eUICC category which is a mandatory information to be shared		
		CR3112R01	Cleanup of description of UpdateMetadata		
		CR3113R01	Removal of implicit enabling of Enterprise Profiles		

		CR3114R01	Editorial cleanup of function binding for ES22		
		CR3117R01	LPA API procedure update with ES2+ interworking		
		CR3119R01	Rewording of notificationPointId and notificationPointStatus		
		CR3120R01	OperationType for Profile Download and Installation		
		CR3121R01	Local Profile Management operation Add-Update All		
		CR3125R00	TLS certificate for HRI server		
		CR3127R00	Addition of missing error codes for RPM		
		CR3101R07	Removing unknown Event from SM-DS		
		CR3122R01	eUICC capability for v3		
		CR3126R04	RPM and ReM clarification as per TF comment		
		CR3128R01	Notification alignment between ES9+ and ES2+		
		CR3129R01	Removal of implicit enabling of Enterprise Profiles in section 3.1		
		CR3132R01	Resolve three Verizon review issues		
		CR3103R09	Rework version negotiation		
		CR3130R03	Start conditions for RPM regarding v2 interoperability		
		CR3131R03	LPA API error codes regarding v2 interoperability		
		CR3133R01	Forward compatibility of activation code		
		CR3134R01	Backward compatibility with v2 certificates		
		CR3138R01	Optional Operation Type for Command Code		
		CR3139R02	Various backward+forward compatibility fixes		
		CR3140R01	RPM Execution reference fix		
		CR3141R01	Editorial clean-up of StoreMetadata		
		CR3142R02	ES10b.CancelSession reason codes		
		CR3143R01	REFRESH correction in RPM procedure		
		CR3144R01	Align UICCCapability with SIMa v2.1		
		CR3145R01	Clarify RSP session		
		CR3146R00	Editorial corrections		
		CR3147R01	Revising Function Input Header of ES2+.HandleNotification		
		CR3150R01	RSP session definition		
		CR3151R01	Remove overlap in 3.1.5		

		CR3152R01	Clean up VendorSpecificExtension		
		CR3154R01	Protected Profile clarification		
		CR3157R01	JSON binding of ES2+.SignCommandCode		
		CR3158R01	HRI and LPAe		
		CR3159R01	Clarification in DS event deletion		
		CR3163R00	Cancel session related procedures		
		CR3036R04	LPA Proxy based on GP SERAM – 1st CR		
		CR3135R02	Backward compatibility with v2 certificates		
		CR3136R04	Server Certificate revocation		
		CR3148R02	Adding MatchingID in ES2+.HandleNotification regarding TF input		
		CR3149R02	Returning a rpmPending flag in ES9+.AuthenticateClient response		
		CR3153R01	ELG15 Implementation		
		CR3155R05	Certificate chain support negotiation		
		CR3160R02	No notifications when switching with test profiles		
		CR3162R01	Enterprise profiles cleanup		
		CR3165R00	Fixing a reference in LPA API procedure		
		CR3168R01	otherSignedNotification clarification		
		CR3169R05	Test mode restrictions		
		CR3170R03	Clarify Enterprise Owned Device		
		CR3172R01	Add LPA API to architecture summary		
		CR3173R02	Remove reference to ASCII		
		CR3174R01	security note regarding euiccCiPKIdToBeUsed		
		CR3176R01	Grouping of Enterprise Features		
		CR3177R01	LPAe and BIP		
		CR3178R01	Deprecating X-Admin-Protocol in ES9+ and ES11		
		CR3181R00	Add missing User Agent indication in ES11		
		CR3182R01	Fix compilation issue in ES9+.AuthenticateClient binding		
		CR3183R00	Fix issues in ES11.AuthenticateClient binding		
		CR3184R00	Fix in OtherNotification definition		
		CR3185R00	Simplify RpmConfiguration		
		CR3187R00	Remove unused reference		
		CR3190R01	Revising wrong section numberings		

		CR3192R00	Remove Context from noSessionContext		
		CR3193R01	General description on RPM procedures		
		CR3194R01	Change Profile Download and Local Profile Management to RSP Operation in Annex C.2		
		CR3199R00	Removing two Editors Notes on combining Profile Download and RPM		
		CR3164R01	Time checks on the eUICC		
		CR3166R02	Review description of all functions (RSPTEC343.AP02)		
		CR3180R01	Simplify eUICC implementation of Update Metadata		
		CR3191R02	Adding description on Command Code for RPM and Event Retrieval		
		CR3196R01	Clarifying contextual operation		
		CR3197R05	Improve AC coding		
		CR3198R02	Fix in VerifyCommandCommand error codes		
		CR3200R01	Missing No columns		
		CR3201R01	Clarify updateMetata in RPM order		
		CR3202R00	Enterprise Profile and Enterprise Configuration		
		CR3204R01	eUICC handling of CtxParamsForCommonAuthentication		
		CR3209R00	Clarify LPA handling of failed Profile installation		
		CR3212R01	Add CAT letter classes to device information		
		CR3207R03	Clarify RPM errors		
		NA	To remove the CR3148R02 MatchingID in ES2+.HandleNotification regarding TF input		
		CR3118R04	Replace RSP by eSIM		
		CR3167R04	Load RPM Package Result clarification		
		CR3179R02	ES10x interoperability description in section 2.4.14 (overview on SVN negotiation)		
		CR3189R05	PCM procedure		
		CR3195R07	Framework for RspCapability exchange		
		CR3203R00	SAS-UP Reference		
		CR3206R01	Clarification on ECKA and euiccCiPKIdListForVerification		

		CR3208R04	Allow empty MatchingID in download procedure		
		CR3211R01	Pending operation alerting procedure		
		CR3215R01	CR3136 cleanup		
		CR3216R02	Remove superfluous profile owner checks for RPM		
		CR3217R00	Remove variant Ov3 for server		
		CR3218R00	Truncated CI Public Key Identifier		
		CR3219R01	Add certificate chains variant for flexibility		
		CR3220R01	eUICC session termination		
		CR3221R01	New functions defined in V3		
		CR3222R02	Security Overview: adding paragraph regarding Notifications		
		CR3223R01	Switching RPM Enabled and Disabled in LPA by End User		
		CR3224R02	Clarification for keys profile binding and random key clarification		
		CR3225R01	OSUpdate Clarifications		
		CR3226R01	SGP.22 Phase 1 backward compatibility		
		CR3227R01	Cleanup of wording in 5.7.14a		
		CR3228R01	Invalid Transaction ID error cases		
		CR3229R06	Make SVN negotiation obsolete		
		CR3230R01	ES2+ cover SMDP43-44-45-46-47-48		
		CR3231R02	Remove CERT.DPauth.ECDSA from ES10b.LoadRpmPackage		
		CR3232R01	End user control over LPA proxy network usage		
		CR3234R01	Remove cancelSessionEmptyResponse		
		CR3235R02	Service Indication in Profile Metadata		
		CR3236R02	Simplifying procedure descriptions in 3.1.3.3		
		CR3237R02	Editorial changes from TF on Draft 9		
		CR3238R00	Consecutive user confirmations		
		CR3240R01	Limit the number of ot key re-use (RSPTEC243.AP08)		
		CR3241R01	Explicitly state the rebinding error (RSPTEC35_AP07)		
		CR3242R02	User friendly way for setting the allowedCiPKId for the default DP (RSPTEC31.4_AP01)		
		CR3243R04	LPA Proxy API		
		CR3244R01	Device capabilities for LPA API, LPA proxy, and APDU API		
		CR3245R01	TLS certificate revocation		

		CR3246R01	Changing Root CI to GSMA Root CI (RSPTEC36.1_AP01)		
		CR3247R00	Correcting attemps to attempts		
		CR3248R01	SetNickname is optional for the LPA		
		CR3250R03	Fix loss of backward compatibility around euiccCiPKIdToBeUsed		
		CR3253R04	Clearly define notificationEvent values		
		CR3254R01	Clean up description of enable, disable and eUICC memory reset		
		CR3255R03	Eliminate references to enterprise owned device		
		CR3188R04	Fix error handling in RPM execution		
		CR3251R02	Simplifying procedure descriptions in section 3.1		
		CR3256R02	New error code rejecting Reference Enterprise Rules		
		CR3257R02	Clean up enterprise checks on the eUICC		
		CR3259R01	Update description of LPA Services		
		CR3260R00	Clarify AAC procedure and AAC Browser		
		CR3261R00	Align wording for root certificates		
		CR3262R01	Clarify LPA API usage as per Telenor comment		
		CR3263R02	eUICC capability for service provider message		
		CR3264R00	Add optional AAC to initial device setup		
		CR3265R03	Encode AAC as a version 2 AC		
		CR3267R01	OperationType in Add Profile and Update Profile		
		CR3268R01	Clarify definition of ES12		
		CR3269R01	Removing Editor's Notes on RPM Package and LPA API		
		CR3270R01	ASN.1 Restructuring		
		CR3271R06	Description on the elements in euiccInfo		
		CR3273R02	TLS optional in ES2+		
		CR3274R01	Intro comments from TF on Draft 11		
		CR3275R01	Clarification about options numbering in RPM		
		CR3276R01	Clarification about the 'allowedCiPKId'		
		CR3277R01	Additional clean-up around svn		
		CR3278R02	Remove dependency on svn for EventList signing		

		CR3279R01	Add a capability on eUICC for VerifyCommandCode		
		CR3280R00	LoadRpmPackageREsultData missing references		
		CR3281R01	Processing for an eUICC not supporting enterprise		
		CR3283R03	LPAe always uses REFRESH		
		CR3284R01	Move ASN.1 for update metadata response to 5.4.1		
		CR3286R00	Clarify LPA behaviour for enabling a Provisioning Profile		
		CR3290R01	ICCID and EID clarifications		
		CR3291R01	Editorial changes in Device Info		
		CR3292R01	New error codes for ASN.1 binding of ESXX.AuthenticateClient		
		CR3293R01	Certificates comments from TF on Draft 11		
		CR3205R09	CI public key and credential update for eUICC		
		CR3239R01	Definitions and Abbreviations comments from TF on Draft 9		
		CR3252R01	Simplifying procedure descriptions in section 3.7		
		CR3258R01	Clarify catBusy in local profile management procedures		
		CR3285R01	Extend error handling in 3.1.5 to include RPM		
		CR3288R03	AAC Comments		
		CR3289R04	eUICC removable or not		
		CR3294R01	Restrict Profile Owner update via RPM		
		CR3297R00	Error code alignment		
		CR3298R04	ES12 TLS requirements		
		CR3299R02	Interoperability between v2.x LPA and v3 eUICC		
		CR3300R03	Clarify CommandCode signing		
		CR3301R00	Update GP CS Amd. F to v1.1		
		CR3302R01	Add event alerting in the starting conditions of affected procedures		
		CR3303R01	Removal of one Editor's Note on RPM		
		CR3304R02	User Confirmation for RPM Update Metadata		
		CR3305R01	Removal of one Editor's Note on RPM (RSPTEC38_AP12)		
		CR3307R01	Add AES keys description (RSPTEC38_AP16)		
		CR3309R01	Remove limitation on validity period for SubCA issued Certificate		

		CR3310R03	Mark a Profile 'to be deleted' on RPM Delete Profile (RSPTEC38_AP13)		
		CR3313R00	Consistently refer to CI PK indicator		
		CR3315R02	TF comment (TFTEC38_Doc019) on RPM		
		CR3319R00	Definition of Enterprise Rule		
		CR3320R01	LPA behaviour on ASN.1 size limit of UTF-8 string		
		CR3306R03	Add cancelSession in Event Retrieval procedure		
		CR3308R00	Fix in ES11.AuthenticateClient binding		
		CR3312R00	Remove constraint on certs to have same subject name		
		CR3314R01	Add missing data objects for ASN1 generation		
		CR3316R02	Additional eUICC info		
		CR3317R01	Cleanup description of RPM ListProfileInfo		
		CR3318R02	Extension of the sequence of 88 TLVs for Profile metadata		
		CR3322R00	Clarifying RPM Contact PCMP error codes		
		CR3323R01	Modify EF UMPC handling		
		CR3324R04	Revising implicit disable in RPM Enable Profile		
		CR3325R00	Remove UI specifics from SetNickname procedure		
		CR3326R00	Bug fix in ES2+.HandleNotification example		
		CR3327R01	Editorial corrections to section 2		
		CR3328R00	Fix specification of LPA46 condition		
		CR3329R02	Align User-Agent header to RFC7231		
		CR3330R00	Action RSPTEC392_AP02&03		
		CR3332R03	Mandatory from version 3.0.0 acronym for feature support		
		CR3334R01	Clarify ES2+.SignCommandCode error		
		CR3335R00	Clarify use of server_name extension for TLS		
		CR3336R00	Clarify use of subCAs in common mutual authentication start conditions		
		CR3337R00	No CI public key restriction for TLS		
		CR3338R00	Allow Variant C in ES9+.InitiateAuthentication response		
		CR3339R03	Fix IMEI encoding definition		
		CR3340R00	Discovery of non-released Profiles		

		CR3341R01	Rework section for Cert verification (RSPTEC40.2_AP02)		
		CR3344R01	Enterprise Capable Devices optionally rejecting Enterprise Rules		
		CR3345R01	Revise ReM operation		
		CR3347R01	GetProfilesInfo example retrieving all tags		
		CR3350R01	Support of minimal file system eSIMs		
		CR3354R01	lpaMode in EUICCInfo2 should be OPTIONAL		
		CR3357R00	Revert to v2 serialNumber description (RSPTEC41_AP09)		
		CR3359R01	Editorial correction to Annex P		
		CR3362R01	DLOA support mandatory		
		CR3311R05	Version interoperability in Annex O and eUICC SVN		
		CR3321R04	Deprecating LPA SVN and Server SVN in the specification		
		CR3333R00	edit Default SM-DP+ is optional for the LPA		
		CR3355R08	Providing new HTTP status codes for HandleNotification response		
		CR3356R05	Clarify MatchingID usage on ES2+		
		CR3358R01	Clarifying mention of default file system		
		CR3360R01	Clarifying Managing SM-DP+ check in ES10c function calls via RPM		
		CR3363R02	Consistent use of GSMA CI RootCA and GSMA CI terms		
		CR3364R01	Rewording 'Phase 3'		
		CR3365R00	Abbreviation of API		
		CR3366R01	euiccRspCapability and LPA API		
		CR3367R02	Revising implicit disable in RPM Enable Profile		
		CR3368R01	Clarifying ES22 functions		
		CR3369R01	Clarification about NotificationEvent coding		
		CR3371R00	Profile Installation Result upon interruption		
		CR3374R01	Clarify GetProfilesInfo response		
		CR3375R00	Store only signed Load RPM Package Results in eUICC		
		CR3376R00	Fix UML in Event Retrieval		
		CR3377R01	Clarifications around Metadata update		
		CR3379R01	Make DP certs subject name unique		
		CR3380R00	ASN.1 comment style		

		CC3381R01	Revise description on RPM Execution procedures		
		CR3382R07	Cryptographic negotiation description		
		CR3383R01	Cleanup related to TF draft 15 comments		
		CR3384R01	Align ES21 to GP SERAM		
		CR3385R01	Editorial in PrepareDownload for curve designation		
		CR3387R01	Activation Code example		
		CR3388R02	Editorial guidelines for capitalisation and ASN.1 comments		
		CR3389R03	HTTP header for JSON		
		CR3390R02	ASN.1 tag fixes		
		CR3391R01	Editorial change in triggering PCM via RPM		
		CR3392R01	Remove obsoleted HTTP RFC		
		CR3394R01	Correct interface description in section 5.1		
		CR3397R01	Remove duplication of GP SERAM details		
		CR3398R00	Remove ed note on RSA		
		CR3400R00	Version reference in section 2.6.6.2		
		CR3401R00	Fix eUICC hardware characteristics		
		CR3393R07	eUICC support for LPA Proxy		
		CR3399R00	Remove ed note on rebind error		
		CR3404R00	Change Auditing eUICC to Audit eUICC		
		CR3405R00	Maximum size of RPM packages		
		CR3406R02	PCM Script definition (RSPTEC44_AP08)		
		CR3407R01	Correction of Open Mobile API Specification		
		CR3409R01	Cleanup of tags for RPM packages		
		CR3410R01	Cleanup of OIDs		
		CR3412R01	Informative reference to Generic Test Profile		
		CR3413R02	Clarify meaning of LPA Proxy		
		CR3414R00	Fix references to ETSI TS 102 221		
		CR3416R01	Clarify ICCID in notification metadata		
		CR3418R01	RPM rejection by LPAd during RPM download		
		CR3422R00	EnterpriseProfilesNotSupported error code revised		
		CR3403R03	Revising FunctionExecutionStatus and NotificationEventStatus		
		CR3402R09	BPP Security Protocol		
		CR3408R04	Privacy of RPM Commands		

		CR3411R08	Adding Status Code to ES9+ Authentication Client		
		CR3415R04	Adding Status Code to ES2+ Cancel Order		
		CR3419R00	Removing "Normal Case" from Procedures		
		CR3420R01	Fulfilling action RSPTEC392_AP05		
		CR3421R01	Fulfilling action RSPTEC392_AP04		
		CR3423R02	Add 5G RAT to DeviceInfo		
		CR3424R01	Deactivating LPAe		
		CR3425R01	ES2+ Download Order clarification		
		CR3426R00	Clean up naming of Event Records		
		CR3428R00	Remove trusted-ca-keys		
		CR3430R01	Update EID definition		
		CR3429R06	Editor’s note for backwards compatibility		
		CR3433R01	Clarify (former) TF comments on Draft 15		
		CR3436R01	TLS handshake with a non-empty session_id		
		CR3437R00	X-Admin-Protocol values in Annex I example		
		CR3438R00	Multiple editorial changes		
		CR3440R01	Fix HTTP path for ES9 ASN.1 binding		
		CR3443R00	Fulfilling voided AC6 requirement		
		CR3444R01	Fulfilling revised Device requirements on EID		
		CR3446R01	Remove discovery ordering between SM-DS and Default SM-DP+		
		CR3447R00	Device reset without eUICC Memory Reset		
		CR3451R01	Clarify that JavaCard is mandatory		
		CR3417R03	Addition of 5G related UICC capability		
		CR3445R05	LPAe using E4 ENVELOPEs		
		CR3448R09	Clean nameConstraints definition in EUM Certificate		
		CR3450R06	Add integrated euicc in eligibility check		
		CR3455R03	Simplified End User confirmation		
		CR3456R00	Clarification in section 5.1		
		CR3457R00	simaResponse is conditional		
		CR3458R01	Allow ES2 resends		
		CR3459R02	SAIP supported versions indication		
		CR3460R00	Clarify LPAe activation		

		CR3463R02	Case-insensitive SM-DP+ FQDN verification		
		CR3466R04	Update the ordering of v3 capabilities in deviceInfo		
		CR3470R01	Reference to GSMA TSG TS.43 as example protocol specification for ESapp Ineterface		
		CR3471R00	Change gsma.com by example.com		
		CR3472R01	HTTP header correction		
		CR3473R00	AC Format clarification		
		CR3476R01	Allow cancel order with ICCID only		
		CR3477R01	Allow more than one Root SM-DS		
		CR3478R00	Cleanup of CR3450		
		CR3479R01	Integrated eUICC REQ ID01		
		CR3480R01	Reference to OCF Easy Setup		
		CR3481R00	Align CR3479 with M2M CR4104R01		
		CR3482R02	Reference SGP.29 EID assignment scheme		
		CR3487R01	Adding a section for Device Change		
		CR3486R00	Profile download retry upon temporary errors		
		CR3488R11	Device Change terms and definitions, and procedure details		
		CR3489R08	Device Change procedure to support v2 eUICC		
		CR3490R07	Function description and function bindings for Device Change		
		CR3491R06	Use of an elementary file for Device Change to support v2 eUICC		
		CR3504R02	MEP - Reserve MEP abbreviation for Multiple Enabled Profiles		
		CR3505R06	Adding estimated profile size in Profile Metadata		
		CR3507R01	Updated Device requirements		
		CR3508R00	Update for removal of LPA45		
		CR3509R00	DS stands for Discovery Service		
		CR3510R07	ES12 enhancement		
		CR3511R00	Rewording "Device-specific" in Annex C		
		CR3512R00	Removal of mandatory RAT configuration		
		CR3513R01	Policy rule updates for WG1 CR0080r2		
		CR3523R02	Reference update to OCF Easy Setup		
		CR3148R06	Add identifiers in ES2+.HandleNotification		

		CR3442R07	Clarify profile metadata field extensibility		
		CR3492R06	Profile Recovery for Device Change		
		CR3518R17	Add MEP support in Procedures section of SGP.22		
		CR3519R11	Add MEP support in Functions section of SGP.22		
		CR3525R01	Presence of OperationType parameter in ES9+.AuthenticateClient		
		CR3526R01	Mirror to V3.0 of CR2711R01 (treProperties mandatory for integrated)		
		CR3528R01	Add informative reference section		
		CR3531R00	Replace deprecated ASN.1 ANY type by an open type		
		CR3533R01	Removing Command Code		
		CR3534R02	Get Profile Size from GetProfilesInfo		
		CR3536R01	Correction of explanation of EstimatedProfileSize		
		CR3543R01	Field-test eUICC		
		CR3544R01	Clarification on outdated ppVersion during interim period		
		CR3494R06	Delivery of the Delete Notification for Device Change		
		CR3506R09	Device Information Code		
		CR3515R13	TLS Security requirement update proposal		
		CR3527R03	Clarification on ICCID in Enhanced ES12		
		CR3529R03	Editorial Clarification on disabling a Test Profile		
		CR3530R03	Device Change Configuration description annex R		
		CR3535R05	Enhanced Discovery Service – Event Checking		
		CR3538R06	MEP: General Description		
		CR3540R07	Enhanced Discovery Service - Push Service		
		CR3541R01	MEP - Fix enable response for eUICC assigned port		
		CR3542R00	MEP - Fix disable response in procedure		
		CR3546R00	Editorial fix for DPpb keys		
		CR3547R03	MEP: Case 3 commands out of scope		
		CR3548R02	Fix profile size description		
		CR3549R01	Update Annex O - RPM optional		
		CR3550R01	Fix ICCID and EID examples		

		CR3555R01	MEP – Clean-up related to Command/Target Port/Profile		
		CR3556R00	Command Code clean up		
		CR3557R00	DEV-IC_fix EID example		
		CR3558R01	Adding DeviceChangeConfig in UpdateMetadata		
		CR3559R00	MEP - Forbidden Policy Rules		
		CR3564R03	MEP – Clarify Command/Target Port in Enable/Disable Procedures		
		CR3565R01	Java Card section update		
		CR3566R00	Remove advanced activation code		
		CR3568R02	Device Change using a stored Activation Code		
		CR3569R02	Function description and function bindings of ES2+ functions for Device Change		
		CR3571R02	Clarifying smdsSigned3		
		CR3573R03	MEP - refresh handling for enabling a Profile		
		CR3575R03	MEP – Clean-up of port selection for Enable procedure		
		CR3577R00	Resolving MEP EN in Clause 3.4		
		CR3578R00	Resolving EN in Clause 3.4.1		
		CR3579R01	Resolving EN in Clause 5.7.1		
		CR3570R01	Clarification on the length of Activation Code for Device Change		
		CR3574R02	MEP – Mode setup		
		CR3580R05	Clause 7.5.16 – MEP-A2 error code addition		
		CR3581R03	MEP – CAT mechanism in C.4		
		CR3583R00	Clean up pushServiceRegistrationResult in Push Service Registration procedure		
		CR3584R00	MEP – Remove isdrIsMepSelectable bit		
		CR3585R04	MEP – enterprise requirements		
		CR3586R03	MEP - Mode setup examples		
		CR3587R02	MEP - Mode setup error response		
		CR3588R00	Add a reference to SGP.25		
		CR3590R02	Rename to priorityEnterpriseProfile		
		CR3592R01	Clause 5.7.16 – MEP-A2 with (no) refreshFlag		
		CR3469R07	RPM support in LPAe using E4 ENVELOPEs		
		CR3517R09	Add MEP Definitions in SGP.22		
		CR3567R08	Server support for multiple Root SM-DSs		

		CR3589R07	Clause 2.12 – Clarify profile assignment to eSIM Port.		
		CR3593R03	MEP – refresh handling for disabling a Profile		
		CR3596R04	Update of TCA Profile Package specification version		
		CR3597R05	RPM with MEP - procedures		
		CR3599R01	New Error code for expired ECID		
		CR3600R02	MEP - simplify memory reset procedure		
		CR3602R00	MEP - align test memory reset procedure		
		CR3603R02	MEP - align memory reset function		
		CR3604R02	Clarify handling of euiccRspCapability in ESXX.AuthenticateClient		
		CR3605R02	Align ASN.1 tags with v2		
		CR3606R06	Common JSON and ASN.1 request binding for ESXX.AuthenticateClient		
		CR3607R06	UICC Capability in TCA Profile Package specification		
		CR3609R00	Clause 2.12 – Add Requirements for profile assignment to eSIM Port - Alt to CR3608		
		CR3612R01	Format of lpaRspCapabilities in JSON		
		CR3614R02	MEP - align test memory reset procedure		
		CR3615R02	Fix reference to section 4.5.2.1.2		
		CR3620R02	APDU Access Interface comments		
		CR3622R05	Algorithm related comments		
		CR3624R01	Removing solved ED in Enable Profile Procedure		
		CR3626R01	JSON base64 + hex cleanup		
		CR3627R01	MEP - outdated ed note on mem reset		
		CR3629R03	MEP - Cleanup of port selection for Enable function		
		CR3468R05	Independent CI – Definition and clarification		
		CR3532R09	Ensure SPN and Profile Name metadata are never empty		
		CR3595R04	Device Change – Notification and profile lifecycle clarification		
		CR3617R05	RPM with MEP - eSIM Port indication by LPA		
		CR3625R01	MEP - reconfiguration of number of ePorts out of scope		

		CR3628R01	Other Notifications Generation Clarification		
		CR3630R01	RPM with MEP - eSIM Port selection during RPM download		
		CR3631R03	Some push service cleanup		
		CR3632R00	Removal of obsolete ed notes in enable procedure		
		CR3633R05	eUICC initialisation section clean-up		
		CR3634R05	RSP sessions and MEP		
		CR3635R00	Fixing RSP Device Capabilities		
		CR3636R02	Update of operation alerting		
		CR3637R00	Removal of ed note in ES2.CancelOrder		
		CR3639R03	MEP - Clarify content of profileState		
		CR3641R01	RPM – Clean-up Get Profiles Info Procedure		
		CR3647R01	EN resolution on Event Checking		
		CR3648R04	ES25 E4E API download result message		
		CR3649R02	EN resolution on Device Change procedure		
		CR3654R00	RPM with MEP - no implicit disabling		
		CR3655R03	MEP - update of enable and disable functions		
		CR3660R00	Fixing step 7 in the notification procedure		
		CR3661R02	Editorial: FQDN extended type		
		CR3662R01	TIMESTAMP correction		
		CR3663R03	Editorial: MEP Removal EN in Disable procedure		
		CR3665R02	MEP – Remove ENs regarding to list APDU multiplexing in the 2.12		
		CR3666R04	MEP – Proposed way-forward for EP on an UP		
		CR3667R01	Clarify location of euiccCiPKIdToBeUsedV3		
		CR3485R08	eUICC OS update security		
		CR3640R02	Signed Event Record is optional for LPA without SM-DS Address		
		CR3643R03	PKI – Support of Variant O		
		CR3658R00	Reorder misplaced sections		
		CR3659R00	Fix optionality in common mutual authentication procedure		
		CR3670R05	Generalization of VerifySmdsResponse		
		CR3672R00	Removal of last EN in RPM execution		
		CR3673R00	Cleanup of delete function		

		CR3674R01	Clarify CRL distribution point		
		CR3675R03	Fixing if statements in ES9+.AuthenticateClient		
		CR3676R00	Cleanup of section 5.7.1		
		CR3677R01	Fix wording on disable in section 2.12		
		CR3678R00	Remove ed note on renaming MEP modes		
		CR3679R02	Combinations of MEP options		
		CR3681R03	PPR2 handling for Device Change		
		CR3684R00	Remove reference to ITU-T X.509		
		CR3685R01	Update reference to GP SERAM		
		CR3686R00	Remove obsolete editor's notes		
		CR3689R00	Clarify disabling with CAT not initialised		
		CR3611R09	eUICC capability on Device Change		
		CR3671R01	Removal of last EN in enable function		
		CR3680R06	Confirmation Code for Device Change		
		CR3687R04	Fix RPM object deletion		
		CR3688R04	MEP and test profiles		
		CR3692R01	Cleanup related to metadata extensibility		
		CR3693R02	Consistently apply euiccRspCapability		
		CR3694R03	Add notification configuration to RPM		
		CR3695R02	Fix signature wording		
		CR3696R00	Fix errors in JSON bindings		
		CR3697R02	Recommend certificate variant usage		
		CR3698R00	Remove ed notes on SMx crpyto		
		CR3699R00	Remove ed note on double errors		
		CR3700R00	Add EID example		
		CR3701R00	Remove ed note on TLS security level		
		CR3702R00	Fix format of hashedIccid		
		CR3704R01	Remove ed note on filtered event record		
		CR3708R00	Fix missing change from CR3148R06		
		CR3709R00	Remove yellow for CC definition		
		CR3710R00	Remove note on ETSI TS 102 223		
		CR3711R00	Resolve yellow in section 3.4.2		
		CR3638R01	Some Device Change clarifications		
		CR3644R01	Clarify content of ES2+.handleNotification/resultData		
		CR3657R06	Reworking ES9+.AuthClient		

		CR3683R06	Clarifying pre-configured Activation Code for Device Change		
		CR3703R04	Resolve ed note in cancel session		
		CR3705R01	Replace SIMalliance wording		
		CR3706R01	Fixes related to enterprise		
		CR3707R01	SM-DS support for hashedIccid		
		CR3712R01	Event retrieval at Device setup		
		CR3713R02	TLS - Removing ED about deprecating AES CBC cipher		
		CR3714R00	Update ETSI references for MEP		
		CR3715R01	Resolve 2 ed notes in GetProfilesInfo		
		CR3716R00	Resolve ed note on SM4 key type		
		CR3717R00	Resolve ed note on unsigned RPM errors		
		CR3718R02	Resolve ed note on Notification within Device Change		
		CR3719R00	Resolve miscellaneous ed notes		
		CR3725R00	Fix references to ES10b.VerifyDeviceChange		
		CR3726R01	Missing functions in RemoteProfileProvisioningRequest		
		CR3727R00	Fix PCM section reference		
		CR3728R02	Annex R Removal of Editor's note		
		CR3732R00	Sort new sub-sections in function binding section		
		CR3664R10	Device Change – Multiple Profiles transfer		
		CR3721R03	Remove LPA API sections		
		CR3724R00	Consistency statement for LPR		
		CR3582R01	Support Device storage of Default SM-DP+ addresses		
		CR3645R04	Clarify content of ES2+.handleNotification/notificationEventStatus for RPM		
		CR3722R04	Clarify tags for base64 JSON fields		
		CR3738R01	Clarify ES9+.AuthClient for Profile download related with Device Change		
		CR3720R04	Always return smdsSigned2 to a v3 LPA discovery request		
		CR3729R04	Device Change Clean up		
		CR3733R00	Correct ES10b.GetProfilesInfo		
		CR3735R00	Fix updateNotificationConfigurationInfoSupport		
		CR3737R00	Fix bullet list in Device Info		
		CR3739R01	Remove customised rpmScript format		

		CR3740R03	Remove dependency on asynchronous Download Preparation Process during Device Change Procedure		
		CR3741R02	Clarify ES12 deletion of stale Event Records		
		CR3743R02	Fix clause numbering		
		CR3744R02	Editorial clean up		
		CR3746R00	Enterprise clean up		
		CR3749R03	Changing DPpb to DPauth in Device Change and Profile Recovery		
		CR3753R00	Fix pending operations alerting		
		CR3730R04	Encrypted Activation Code for Device Change		
		CR3734R01	Various fixes in AuthenticateClient		
		CR3742R01	Clarify the meaning of 'empty' euiccCiPKIdToBeUsed		
		CR3745R01	Remove ReM		
		CR3751R01	No tag-length for eUICCChallenge before base64 encoding		
		CR3752R00	Add MEP in 1.9 feature support		
		CR3754R02	Fix metadata update alerting		
		CR3757R02	Clarifying additional Profile Package versions in euiccInfo2		
		CR3758R00	Clarifying GlobalPlatform version in euiccInfo2		
		CR3760R02	Consider LPA certChainV3Support when selecting cert chain variant		
		CR3761R00	Change INTEGER to NULL for LPA alerting		
		CR3762R03	Common command to trigger LPA pending operation alerting		
		CR3763R03	Rename ES2+.ConfirmDeviceChange to ES2+.HandleDeviceChangeRequest		
		CR3764R00	Resolving yellows in ES10b.PrepareDeviceChange		
		CR3766R00	Enterprise Profiles clarification		
		CR3767R00	Resolving yellowed text in Device Change procedure		
		NA	Correction of SmdpSigned5 definition		
		NA	Correction of SmdpSigned5 definition in Annex H		

Q.2 Other Information

Type	Description
Document Owner	Yolanda Sanz / GSMA
Editor / Company	Denis Praca / Thales

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.