ETSI TS 118 101 V3.22.0 (2021-02)

oneM2M;
Functional Architecture
(oneM2M TS-0001 version 3.22.0 Release 3)

Reference
RTS/oneM2M-000001v3

Keywords
architecture, IoT, M2M

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Important notice

The present document can be downloaded from:
http://www.etsi.org/standards-search

The present document may be made available in electronic versions and/or in print. The content of any electronic and/or print versions of the present document shall not be modified without the prior written authorization of ETSI. In case of any existing or perceived difference in contents between such versions and/or in print, the prevailing version of an ETSI deliverable is the one made publicly available in PDF format at www.etsi.org/deliver.

Users of the present document should be aware that the document may be subject to revision or change of status.
Information on the current status of this and other ETSI documents is available at
https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx

If you find errors in the present document, please send your comment to one of the following services:
https://portal.etsi.org/People/CommiteeSupportStaff.aspx

Contents

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (https://ipr.etsi.org/).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

Foreword

This Technical Specification (TS) has been produced by ETSI Partnership Project oneM2M (oneM2M).

1 Scope

The present document describes the end-to-end oneM2M functional architecture, including the description of the functional entities and associated reference points.

oneM2M functional architecture focuses on the Service Layer aspects and takes Underlying Network-independent view of the end-to-end services. The Underlying Network is used for the transport of data and potentially for other services.

2 References

2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

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

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are necessary for the application of the present document.

[1] ETSI TS 118 111: "oneM2M; Common Terminology (oneM2M TS-0011)".

[2] ETSI TS 118 103: "oneM2M; Security solutions (oneM2M TS-0003)".

[3] ETSI TS 118 104: "oneM2M; Service Layer Core Protocol Specification (oneM2M TS-0004)".

[4] W3C Recommendation: "RDF 1.1 Concepts and Abstract Syntax".

[5] W3C Recommendation: "SPARQL 1.1 Query Language".

[6] ETSI TS 118 112: "oneM2M; Base Ontology (oneM2M TS-0012)".

[7] ETSI TS 118 121: "oneM2M; oneM2M and AllJoyn® Interworking (oneM2M TS-0021)".

[8] ETSI TS 118 123: "oneM2M; Home Appliances Information Model and Mapping (oneM2M TS-0023)".

[9] ETSI TS 118 116: "oneM2M; Secure Environment Abstraction (oneM2M TS-0016)".

[10] ETSI TS 118 122: "oneM2M; Field Device Configuration (oneM2M TS-0022)".

[11] IETF RFC 5771: "IANA Guidelines for IPv4 Multicast Address Assignments".

[12] IETF RFC 2375: "IPv6 Multicast Address Assignments".

[13] ETSI TS 118 132: "MAF and MEF Interface Specification (oneM2M TS-0032)".

[14] oneM2M TS-0034: "Semantics Support".

[15] ETSI TS 118 126: "3GPP Interworking (oneM2M TS-0026)".

[16] Void.

[17] IETF RFC 4566: "SDP: Session Description Protocol".

[18] IETF RFC 3986: "Uniform Resource Identifier (URI): Generic Syntax".

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

[i.1] ETSI TS 118 102: "oneM2M Requirements (oneM2M TS-0002)".

[i.2] Broadband Forum TR-069: "CPE WAN Management Protocol Issue": 1 Amendment 5, November 2013.

[i.3] OMA-DM: "OMA Device Management Protocol", Version 1.3, Open Mobile Alliance.

[i.4] LWM2M: "OMA LightweightM2M", Version 1.0, Open Mobile Alliance.

[i.5] OMA-TS-MLP-V3-4-20130226-C: "Mobile Location Protocol", Version 3.4.

[i.6] OMA-TS-REST-NetAPI-TerminalLocation-V1-0-20130924-A: "RESTful Network API for Terminal Location", Version 1.0.

[i.7] IETF RFC 1035: "Domain names - Implementation and specification".

[i.8] IETF RFC 3588: "Diameter Base Protocol".

[i.9] IETF RFC 3596: "DNS Extensions to Support IP Version 6".

[i.10] Void.

[i.11] IETF RFC 4006: "Diameter Credit-Control Application".

[i.12] IETF RFC 6895: "Domain Name System (DNS) IANA Considerations".

[i.13] GSMA-IR.67: "DNS/ENU Guidelines for Service Providers & GRX/IPX Providers".

[i.14] ETSI TS 123 682: "Digital cellular telecommunications system (Phase 2+) (GSM); Universal Mobile Telecommunications System (UMTS); LTE; Architecture enhancements to facilitate communications with packet data networks and applications (3GPP TS 23.682 Release 13)".

[i.15] ETSI TS 132 240: "Digital cellular telecommunications system (Phase 2+) (GSM); Universal Mobile Telecommunications System (UMTS); LTE; Telecommunication management; Charging management; Charging architecture and principles (3GPP TS 32.240)".

[i.16] ETSI TS 132 299: "Digital cellular telecommunications system (Phase 2+) (GSM); Universal Mobile Telecommunications System (UMTS); LTE; Telecommunication management; Charging management; Diameter charging applications (3GPP TS 32.299)".

[i.17] 3GPP2 X.P0068: "Network Enhancements for Machine to Machine (M2M)".

[i.18] JNI 6.0 API Specification: "Java Native Interface 6.0 Specification".

[i.19] Void.

[i.20] Void.

[i.21] Void.

[i.22] Void.

[i.23] ETSI TS 123 003: "Digital cellular telecommunications system (Phase 2+) (GSM); Universal Mobile Telecommunications System (UMTS); LTE; 5G; Numbering, addressing and identification (3GPP TS 23.003)".

[i.24] Recommendation ITU-T X.660 | ISO/IEC 9834-1: "Information technology - Procedures for the operation of object identifier registration authorities: General procedures and top arcs of the international object identifier tree".

[i.25] ETSI TR 118 508: "Analysis of Security Solutions for the oneM2M System".

[i.26] IETF RFC 4122: "A Universally Unique IDentifier (UUID) URN Namespace".

[i.27] oneM2M Drafting Rules.

NOTE: Available at http://www.onem2m.org/images/files/oneM2M-Drafting-Rules.pdf.

[i.28] oneM2M TR-0007: "Study of Abstraction and Semantics Enablement".

[i.29] Void.

[i.30] Void.

[i.31] OMA-TS-REST-NetAPI-CommunicationPatterns-V1-0: "RESTful Network API for Communication Patterns", Version 1.0, Open Mobile Alliance.

[i.32] ETSI TS 123 246: "Universal Mobile Telecommunications System (UMTS); LTE; Multimedia Broadcast/Multicast Service (MBMS); Architecture and functional description (3GPP TS 23.246 Release 14)".

[i.33] ETSI TS 123 468: "LTE; Group Communication System Enablers for LTE (GCSE_LTE); Stage 2 (3GPP TS 23.468 version 14.0.0 Release 14)".

[i.34] IETF RFC 3171 (2001): "IANA Guidelines for IPv4 Multicast Address Assignments".

[i.35] IETF RFC 4291 (2006): "IP Version 6 Addressing Architecture".

[i.36] IETF RFC 6838 (2013): "Media Type Specifications and Registration Procedures".

[i.37] IETF RFC 3987: "Internationalized Resource Identifiers (IRIs)".

NOTE: Available at https://www.ietf.org/rfc/rfc3987.txt.

3 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the terms given in ETSI TS 118 111 [1] and the following apply:

NOTE: A term defined in the present document takes precedence over the definition of the same term, if any, in ETSI TS 118 111 [1].

access control attributes: set of parameters of the Originator, target resource, and environment against which there could be rules evaluated to control access

NOTE: An example of Access Control Attributes of Originator is a role. Examples of Access Control Attributes of Environment are time, day and IP address. An example of Access Control Attributes of targeted resource is creation time.

access decision: authorization reached when an entity's Privileges, as well as other Access Control Attributes, are evaluated

application layer: comprises oneM2M Applications and related business and operational logic

attribute: stores information pertaining to the resource

NOTE: An attribute has a name and a value. Only one attribute with a given name can belong to a given resource. For an attribute defined as having "multiplicity" greater than 1, the value of that attribute is a composite value, i.e. a list of different values.

child resource: sub-resource of another resource that is its parent resource

NOTE: The parent resource contains references to the child resources(s).

common services layer: consists of oneM2M service functions that enable oneM2M Applications (e.g. management, discovery and policy enforcement)

Common Services Function (CSF): informative architectural construct which conceptually groups together a number of sub-functions

NOTE: Those sub-functions are implemented as normative resources and procedures. A set of CSFs is contained in the CSE.

content based discovery: is the discovery operation for <contentInstance> resources which is matched with the given condition regarding *content* attribute of <contentInstance> resource under specific <container>

NOTE: Content based discovery is based on knowledge about data structure of M2M data stored at <container>.

execution environment: logical entity that represents an environment capable of running software modules

hosting CSE: CSE where the addressed resource is hosted

M2M service provider domain: is the part of the M2M System that is associated with a specific M2M Service Provider

managed entity: may be either an M2M Device, M2M Gateway, or a device in the M2M Area Network or the M2M Application Layer or M2M Service Layer software components

management proxy: entity within the Device Management Architecture, in conjunction with the Management Client, that acts as an intermediary between the Management Server and the Proxy Management Client

network services layer: provides transport, connectivity and service functions

node: logical entity that is identifiable in the M2M System

non-oneM2M node: node that does not contain oneM2M Entities

notifier: Hosting CSE that initiates notifications to Notification Targets in the subscription/notification framework or in the non-blocking asynchronous scheme

notification target: is an AE or CSE that receives notifications from the Notifier

NULL: *null* value

NOTE: Refer to ETSI TS 118 104 [3] for the definition of *null*.

originator: in case of a request traversing a single reference point, the Originator is the AE/CSE that sends the request

NOTE: In case of a request that traverses multiple reference points, the Originator is the AE/CSE that sends the first request in the sequence.

proxy management client: entity within the Device Management Architecture that provides local management capabilities to a device in an M2M Area Network

receiver: is the entity that receives the Request

NOTE: A Receiver can a CSE or can be and AE when notification is requested.

receiver CSE: any CSE that receives a request

registree: AE or CSE that registers with another CSE

registrar CSE: CSE is the CSE where an Application or another CSE has registered

resource: uniquely addressable entity in oneM2M architecture

NOTE: A resource is transferred and manipulated using CRUD operations. A resource can contain child resource(s) and attribute(s), which are also uniquely addressable.

role: collection of permissions that can be statically or dynamically granted to an entity

service charging and accounting: set of functionalities within the M2M Service Layer that enable configuration of information collection and charging policies, collection of Charging Records based on the policies, and correlation of Charging Records to users of M2M common services

service charging record: formatted collection of information about a chargeable operation

service layer offline charging: mechanism where charging information does not affect, in real-time, the service rendered

service layer online charging: mechanism where charging information can affect, in real-time, the service rendered, including real time credit control

software package: is an entity that can be deployed on the Execution Environment

NOTE: It can consist of entities such as software modules, configuration files, or other entities.

structured data: is data that either has a structure according to a specified Information Model or is otherwise organized in a defined manner

transit CSE: is any receiver CSE that is not a Hosting CSE

3.2 Symbols

Void.

3.3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

2G	Second Generation
3GPP	3rd Generation Partnership Project
3GPP2	3rd Generation Partnership Project 2
A/AAAA	IPv4/IPv6 DNS records that are used to map hostnames to an IP address
AAA	Authentication, Authorization, Accounting
AAAA	Authentication, Authorization, Accounting and Auditing
ABT	Additional Back-off Time
ACA	Accounting Answer
ACK	Acknowledged
ACP	Access Control Policy
ACR	Accounting Request
ADN	Application Dedicated Node
ADN-AE	AE which resides in the Application Dedicated Node
AE	Application Entity
AE/CSE	Application Entity/Common Services Entity
AE-ID	Application Entity Identifier
AID	Addressing and Identification
Annc	Announced
API	Application Program Interface
App-ID	Application Identifier
AS	Application Server
ASCII	American Standard Code for Information Interchange
ASM CSF	Application and Service Layer Management CSF
ASM	Application and Service Layer Management

ASN	Application Service Node
ASN/MN	Application Service Node/Middle Node
ASN-AE	Application Entity that is registered with the CSE at Application Service Node
ASN-CSE	CSE which resides in the Application Service Node
BBF	BroadBand Forum
CBOR	Concise Binary Object Representation
CDR	Charging Data Record
CF	Configuration Function
CHF	Charging Function
CM	Conditional Mandatory
CMDH	Communication Management and Delivery Handling
COSEM	Companion Specification for Energy Metering
CRUD	Create Retrieve Update Delete
CRUDN	Create Retrieve Update Delete Notify
CSE	Common Services Entity
CSE-ID	Common Service Entity Identifier
CSE-PoA	CSE Point of Access
CSF	Common Services Function
DAS	Dynamic Authorization System
DCF	Device Configuration Function
DDMF	Device Diagnostics and Monitoring Function
DFMF	Device Firmware Management Function
DIS CSF	Discovery CSF
DIS	Discovery
DM	Device Management
DMG CSF	Device Management CSF
DMG	Device Management
DMR	Data Management and Repository
DNS	Domain Name Server
DRX	Discontinuous Reception
DTMF	Device Topology Management Function
DWAPI	Device Web API
E2E	End-to-End
ESN	Electronic Serial Number
FIFO	First-In First-Out
FQDN	Fully Qualified Domain Name
GMG CSF	Group Management CSF
GMG	Group Management
GMLC	Gateway Mobile Location Center
GPRS	General Packet Radio Service
GPS	Global Positioning System
GSMA	Global System for Mobile Communications Association (GSM Association)
HA/LMA	Home Agent/Local Mobility Agent
HAAA	Home AAA
HAIM	Home Appliance Information Model
HLR	Home Location Register
HTTP	HyperText Transfer Protocol
IANA	Internet Assigned Numbers Authority
IBT	Initial Back-off Time
ID	Identifier
IETF	Internet Engineering Task Force
IGMP	Internet Group Management Protocol
IMEI	International Mobile Equipment Identity
IMS	IP Multimedia System
IMSI	International Mobile Subscriber Identity
IN	Infrastructure Node
IN-AE	Application Entity that is registered with the CSE in the Infrastructure Node
IN-CSE	CSE which resides in the Infrastructure Node
IN-DMG	Infrastructure Node Device ManaGement
IN-DMG-MA	Infrastructure Node Device ManaGement Management Adapter
IP	Internet Protocol
IPE	Interworking Proxy application Entity

IRI	Internationalized Resource Identifier
ISO	International Organization for Standardization
ITU-T	International Telecommunication Union - Telecommunication
IWF	InterWorking Function
JNI	Java Native Interface
JSON	JavaScript Object Notation
LOC CSF	Location Common Services Function
LOC	Location
LWM2M	Lightweight M2M
M2M	Machine to Machine
M2M-EXT-ID	Machine to Machine External IDentifier
M2M-IWF	M2M - InterWorking Function
M2M-Sub-ID	M2M service - Subscription - Identifier
MA	Mandatory Announced
MAF	M2M Authentication Function
MBMS	Multimedia Broadcast Multicast Service
MBT	Maximum Back-off Time
Mca	Reference Point for M2M Communication with AE
Mcc	Reference Point for M2M Communication with CSE
Mcc'	Reference Point for M2M Communication with CSE of different M2M Service Provider
Mch	Reference Point for M2M Communication with external charging server
Mcn	Reference Point for M2M Communication with NSE
Mcs	Reference Point to access functions and data protected within local secure environments
MEF	M2M Enrolment Function
MEID	Mobile Equipment Identifier
MIC	Message Integrity Code
MIP	Mobile IP
MLD	Multicast Listener Discovery
MN	Middle Node
MN-AE	Application Entity that is registered with the CSE in Middle Node
MN-CSE	CSE which resides in the Middle Node
MQTT	Message Queuing Telemetry Transport
MSISDN	Mobile Subscriber International Subscriber Directory Number
MTC	Machine Type Communications
MTE	M2M Trust Enabler
NA	Not Announced
NAT	Network Address Translation
NoDN	Non-oneM2M Node
NP	Not Present
NSE	Network Service Entity
NSSE CSF	Network Service Exposure, Service Execution and Triggering CSF
NSSE	Network Service Exposure, Service Execution and Triggering
NWA	NetWork Action
OA	Optional Announced
OID	Object Identifier
OMA	Open Mobile Alliance
OMA-DM	Open Mobile Alliance - Device Management
OS	Operating System
OUI	Organizationally Unique Identifier
OWL	Web Ontology Language
PDP	Packet Data Protocol
PDSN	Packet Data Serving Node
PEP	Policy Enforcement Point
PIP	Policy Information Point
PMIP	Proxy Mobile IP
PoA	Point of Access
PPM	Privacy Policy Manager
PPP	Point to Point Protocol
PRP	Policy Retrieval Point
PSM	Power Saving Mode
QoS	Qualify of Service
RAM	Random Access Memory

RBAC	Role Based Access Control
RBT	Random Back-off Time
RDF	Resource Description Framework
REG CSF	Registration CSF
REG	Registration
RFC	Request for Comments
RO	Read Only
RPC	Remote Procedure Calls
RTP	Real-Time Transport *Protocol*
RW	Read Write
SCA CSF	Service Charging and Accounting CSF
SCA	Service Charging and Accounting
SCEF	Service Capability Exposure Function
SDO	Standards Developing Organization
SDP	Session Description Protocol
SE	Secure Environment
SEA	Security Association Endpoint
SEC CSF	Security CSF
SEC	Security
SEM CSF	Semantics Common Services Function
SLA	Service Level Agreement
SMF	Software Monitoring Function
SMI	Semantic Mashup Instance
SMJP	Semantic Mashup Job Profile
SMS	Short Messaging Service
SP	Service Provider
SPARQL	SPARQL Protocol and RDF Query Language
SP-ID	Service Provider Identifier
SSM	Service Session Management
SUB CSF	Subscription and Notification CSF
SUB	Subscription and Notification
SWT	Spreading Wait Time
TLS	Transport Layer Security
TMG CSF	Transaction Management CSF
TMG	Transaction Management
TMGI	Temporary Mobile Group Identity
TP	Traffic Patterns
TR	Technical Report
TS	Technical Specification
Tsms	Interface between Short Message Entity (SME) and Short Message Service Center (SMS SC)
Tsp	Interface between Service Capability Server (SCS) and Machine Type Communication (MTC) InterWorking Function
UE	User Equipment
UL	UpLink
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
URN	Uniform Resource Name
UUID	Universally Unique Identifier
WLAN	Wireless Local Area Network
WO	Write Once
XML	eXtensible Markup Language
XSD	XML Schema Definition

4 Conventions

The keywords "Shall", "Shall not", "May", "Need not", "Should", "Should not" in the present document are to be interpreted as described in the oneM2M Drafting Rules [i.27].

To improve readability:

- The information elements of oneM2M Request/Response messages will be referred to as parameters. Parameter abbreviations will be written in bold italic.
- The information elements of resources will be referred to as attributes and child resources. Attributes will be written in italics.

5 Architecture Model

5.1 General Concepts

Figure 5.1-1 depicts the oneM2M Layered Model for supporting End-to-End (E2E) M2M Services. This layered model comprises three layers: Application Layer, Common Services Layer and the underlying Network Services Layer.

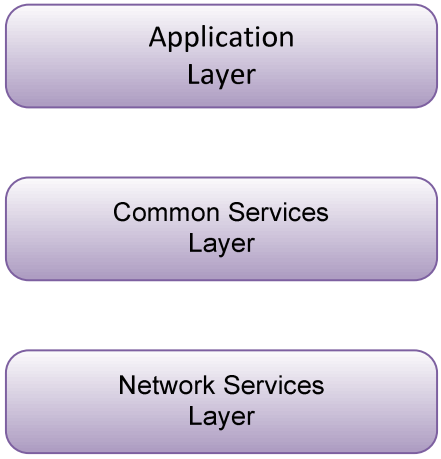


Figure 5.1-1: oneM2M Layered Model

5.2 Architecture Reference Model

5.2.1 Functional Architecture

Figure 5.2.1-1 illustrates the oneM2M functional architecture.

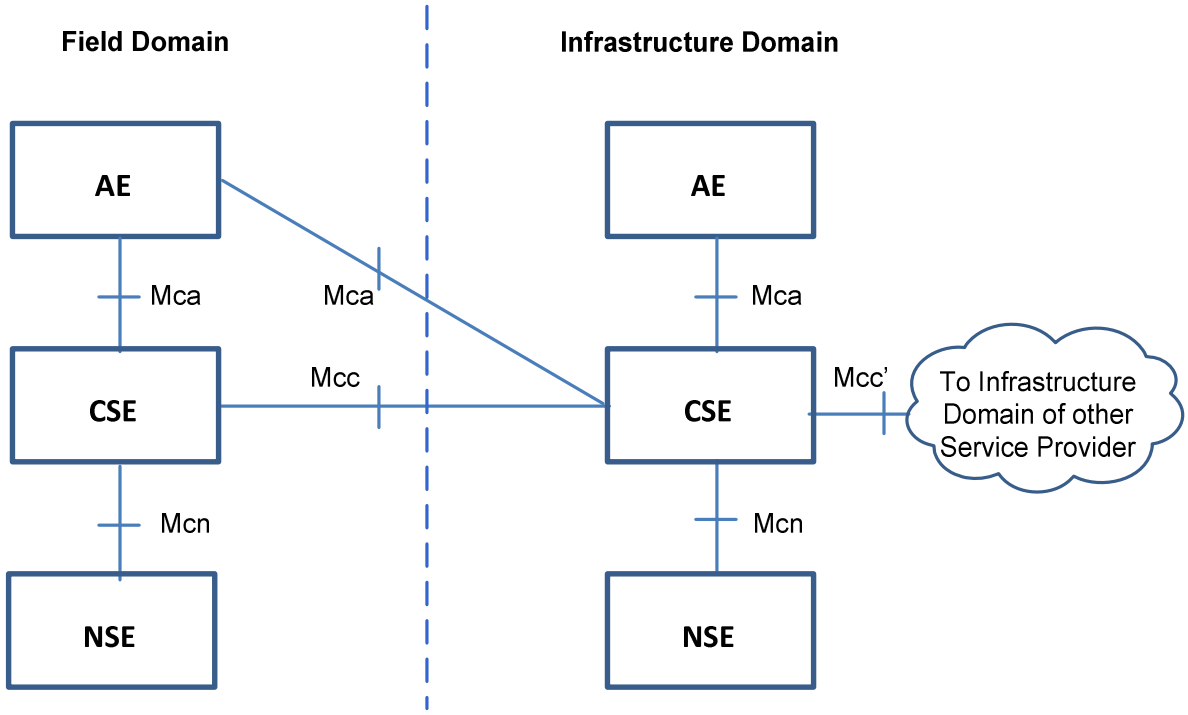


Figure 5.2.1-1: oneM2M Functional Architecture

NOTE 1: Other reference points are specified in other clauses of the present document. See clauses 6.2.4 and 12.2.1.

NOTE 2: The above architecture diagram is a functional diagram. For examples of physical mappings, see clause 6.

The oneM2M functional architecture in figure 5.2.1-1 comprises the following functions:

1) **Application Entity (AE):** Application Entity is an entity in the application layer that implements an M2M application service logic. Each application service logic can be resident in a number of M2M nodes and/or more than once on a single M2M node. Each execution instance of an application service logic is termed an "Application Entity" (AE) and is identified with a unique AE-ID (see clause 7.1.2). Examples of the AEs include an instance of a fleet tracking application, a remote blood sugar monitoring application, a power metering application, or a controlling application.

2) **Common Services Entity (CSE):** A Common Services Entity represents an instantiation of a set of "common service functions" of the M2M environments. Such service functions are exposed to other entities through the Mca and Mcc reference points. Reference point Mcn is used for accessing underlying Network Service Entities. Each Common Service Entity is identified with a unique CSE-ID (see clause 7.1.4).

   Examples of service functions offered by CSE include: Data Management, Device Management, M2M Service Subscription Management, and Location Services. Such "sub-functions" offered by a CSE may be logically and informatively conceptualized as Common Services Functions (CSFs). The normative Resources which implement the service functions in a CSE can be mandatory or optional.

3) Underlying **Network Services Entity (NSE):** A Network Services Entity provides services from the underlying network to the CSEs. Examples of such services include device management, location services and device triggering. No particular organization of the NSEs is assumed.

NOTE 3: Underlying networks provide data transport services between entities in the oneM2M System. Such data transport services are not included in the NSE.

5.2.2 Reference Points

5.2.2.0 Overview

A reference point consists of one or more interfaces of any kind. The following reference points are supported by the Common Services Entity (CSE). The "Mc(-)" nomenclature is based on the mnemonic "M2M communications".

NOTE: Information exchange between two M2M Entities assumes the usage of the transport and connectivity services of the Underlying Network, therefore, they are not explicitly defined as services provided by the underlying Network Service Entity(s) in the scope of the present document.

5.2.2.1 Mca Reference Point

Communication flows between an Application Entity (AE) and a Common Services Entity (CSE) cross the Mca reference point. These flows enable the AE to use the services supported by the CSE, and for the CSE to communicate with the AE.

NOTE: The AE and the CSE may or may not be co-located within the same physical entity.

5.2.2.2 Mcc Reference Point

Communication flows between two Common Services Entities (CSEs) cross the Mcc reference point. These flows enable a CSE to use the services supported by another CSE.

5.2.2.3 Mcn Reference Point

Communication flows between a Common Services Entity (CSE) and the Network Services Entity (NSE) cross the Mcn reference point. These flows enable a CSE to use the supported services (other than transport and connectivity services) provided by the NSE.

5.2.2.4 Mcc' Reference Point

Communication flows between two Common Services Entities (CSEs) in Infrastructure Nodes (IN) that are oneM2M compliant and that resides in different M2M SP domains cross the Mcc' reference point. These flows enable a CSE of an IN residing in the Infrastructure Domain of an M2M Service Provider to communicate with a CSE of another IN residing in the Infrastructure Domain of another M2M Service Provider to use its supported services, and vice versa.

Mcc' extends the reachability of services offered over the Mcc reference point, or a subset thereof.

The trigger for these communication flows may be initiated elsewhere in the oneM2M network.

5.2.2.5 Other Reference Points and Interfaces

- See clause 12.2.1 for Mch reference point.
- See clause 6.2.4 for Mc, Mp, Ms and La device management interfaces.
- See clause 6.2.10 for Mcs reference point.
- See clause 11.0 for Mmaf and Mmef reference points.

6 oneM2M Architecture Aspects

6.1 Configurations supported by oneM2M Architecture

The possible configurations of inter-connecting the various entities supported within the oneM2M system are illustrated in figure 6.1-1. The illustration does not constrain the multiplicity of the entities nor require that all relationships shown are present.

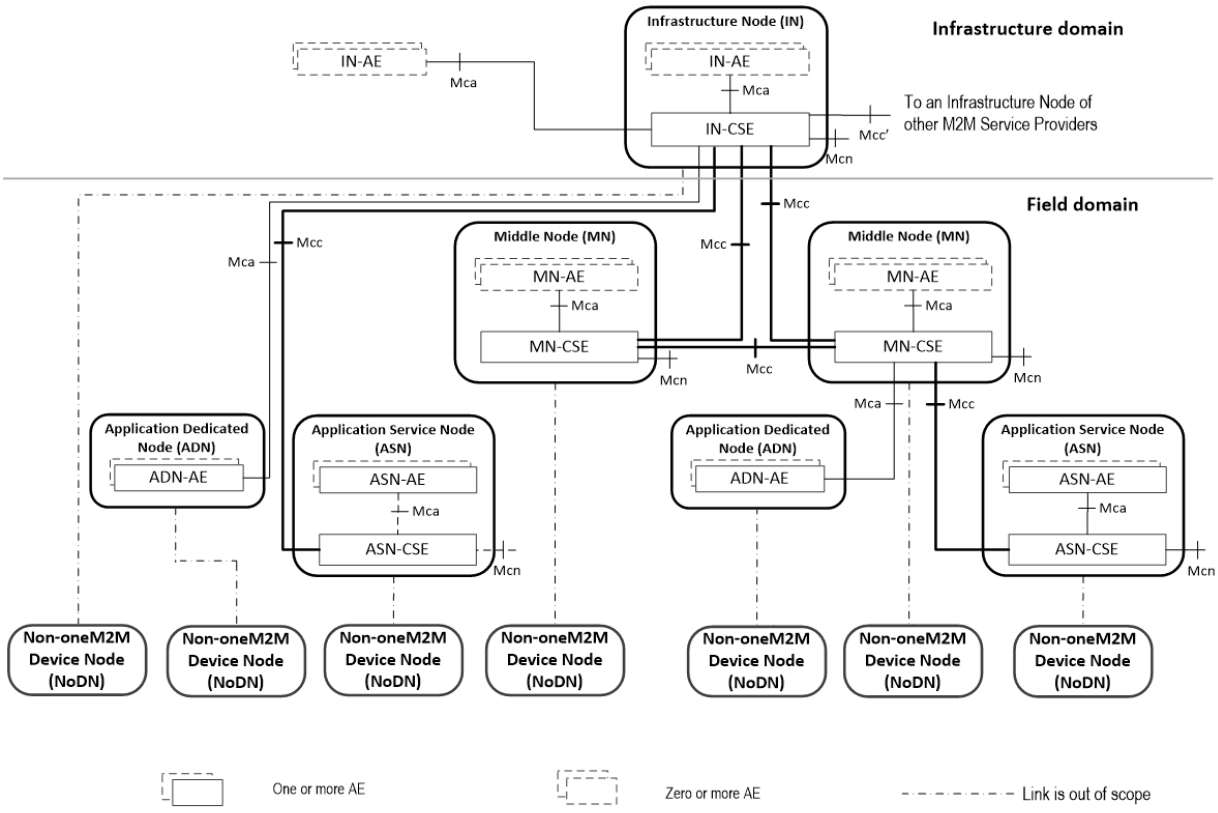


Figure 6.1-1: Configurations supported by oneM2M Architecture

Nodes:

Nodes are logical entities that are individually identifiable in the oneM2M System. Nodes are either CSE-Capable or Non-CSE-Capable:

- A CSE-Capable Node is a logical entity that contains one oneM2M CSE and contains zero or more oneM2M AEs. The ASN, IN and MN are examples of CSE-Capable Nodes.
- A Non-CSE-Capable Node is a logical entity that does not contain a oneM2M CSE and contains zero or more oneM2M AEs. The ADN and Non-oneM2M Node are examples of Non-CSE-Capable Nodes.

CSEs resident in different Nodes can be different and are dependent on the services supported by the CSE and the characteristics (e.g. different memory, firmware) of the physical entity that contains the CSE's Node.

Description of Node types:

The oneM2M architecture enables the following types of Nodes. As logical objects, such Nodes may or may not be mapped to physical objects.

Application Service Node (ASN):

An ASN is a Node that contains one CSE and contains at least one Application Entity (AE). There may be zero or more ASNs in the Field Domain of the oneM2M System.

The CSE in an ASN communicates over the Mcc reference point with one CSE residing in a MN or in an IN.

An AE in an ASN communicates over the Mca reference point with the CSE residing in the same ASN.

An ASN communicates over Mcn with NSEs.

Example of physical mapping: an ASN could reside in an M2M Device.

Application Dedicated Node (ADN):

An ADN is a Node that contains at least one AE and does not contain a CSE. There may be zero or more ADNs in the Field Domain of the oneM2M System.

An AE in the ADN communicates over the Mca reference point with a CSE residing in a MN or in an IN.

Example of physical mapping: an Application Dedicated Node could reside in a constrained M2M Device.

Middle Node (MN):

A MN is a Node that contains one CSE and contains zero or more AEs. There may be zero or more MNs in the Field Domain of the oneM2M System.

The CSE in a MN communicates over the Mcc reference point with one CSE residing in a MN or in an IN and with one or more other CSEs residing in MNs or in ASNs.

In addition, the CSE in the MN can communicate over the Mca reference point with AEs residing in the same MN or residing in an ADN.

A CSE in a MN communicates over Mcn with NSEs.

Example of physical mapping: a MN could reside in an M2M Gateway.

Infrastructure Node (IN):

An IN is a Node that contains one CSE and contains zero or more AEs. There is exactly one IN in the Infrastructure Domain per oneM2M Service Provider. A CSE in an IN may contain CSE functions not applicable to other node types.

The CSE in the IN communicates over the Mcc reference point with one or more CSEs residing in MN(s) and/or ASN(s).

The CSE in the IN communicates over the Mca reference point with one or more AEs residing in the same IN or residing in an ADN.

The CSE in the IN communicates over the Mcn reference point with NSEs, and over the Mcc' reference point with CSEs residing in the INs of other M2M Service Providers.

Example of physical mapping: an IN could reside in an M2M Service Infrastructure.

Non-oneM2M Node (NoDN):

A non-oneM2M Node is a Node that does not contain oneM2M Entities (neither AEs nor CSEs). Such Nodes represent devices attached to the oneM2M system for interworking purposes, including management.

A Non-oneM2M Node communicates (as shown by dotted lines in figure 6.1-1) with the oneM2M System according to annex F.

Domain Types:

The Infrastructure Domain of any particular M2M Service Provider contains exactly one Infrastructure Node.

The Field Domain of any particular M2M Service Provider can contain Application Service Nodes, Application Dedicated Nodes, Middle Nodes and Non-oneM2M Nodes.

6.2 Common Services Functions

6.2.0 Overview

This clause describes the services provided by the Common Services Layer in the M2M System. Such services reside within a CSE and are referred to as Common Services Functions (CSFs). The CSFs provide services to the AEs via the Mca reference point and to other CSEs via the Mcc reference point. CSEs interact with the NSE via the Mcn reference point. An instantiation of a CSE in a Node comprises a subset of the CSFs from the CSFs described in the present document.

The CSF descriptions in this clause are provided for the understanding of the oneM2M Architecture functionalities and are informative. The CSFs contained inside the CSE can interact with each other but how these interactions take place are not specified in the present document.

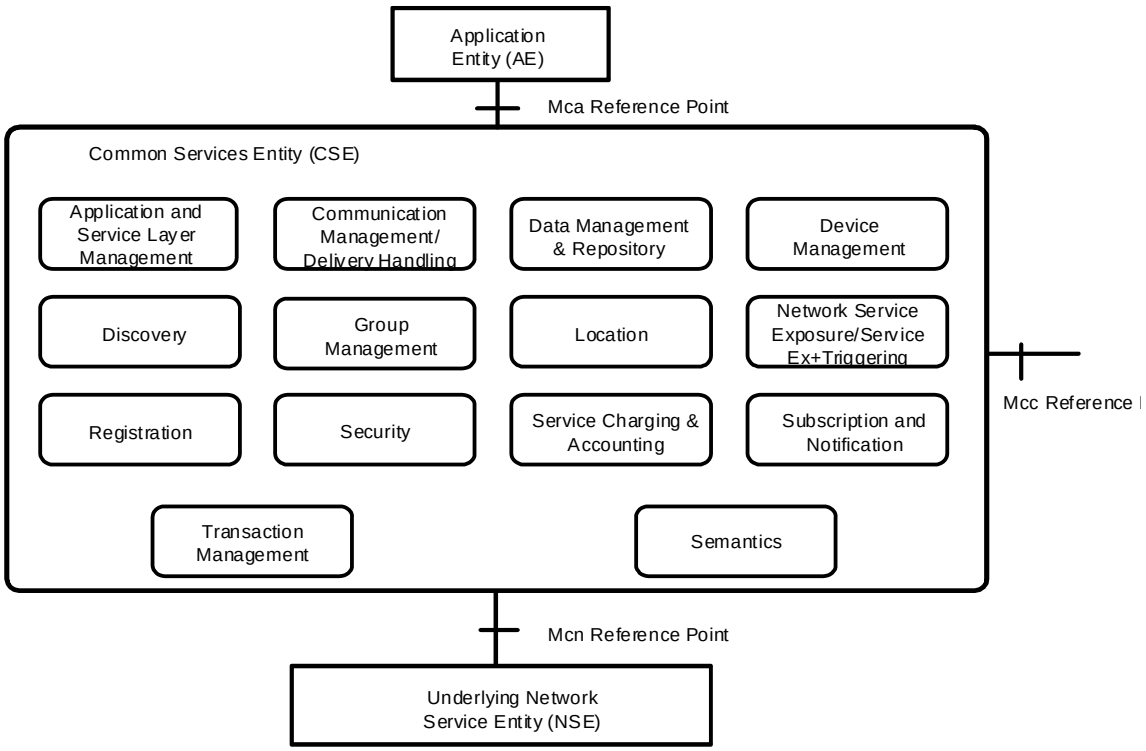


Figure 6.2.0-1: Common Services Functions

6.2.1 Application and Service Layer Management

6.2.1.1 General Concepts

The Application and Service Layer Management (ASM) CSF provides management of the AEs and CSEs on the ADNs, ASNs, MNs and INs. This includes capabilities to configure, troubleshoot and upgrade the functions of the CSE, as well as to upgrade the AEs.

6.2.1.2 Detailed Descriptions

6.2.1.2.0 Overview

The ASM CSF provides management capabilities for CSEs and AEs.

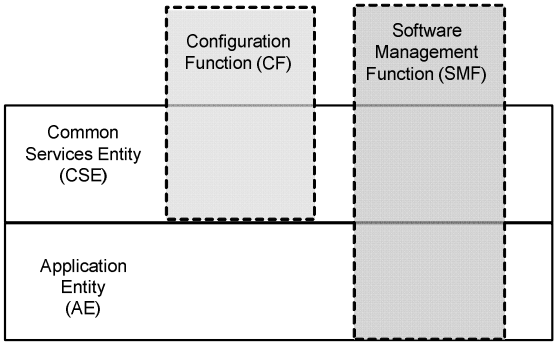


Figure 6.2.1.2.0-1: Management Layers and Function

The ASM CSF utilizes the functions provided by the Device Management (DMG) CSF for interaction with the Management Server.

The management functions include:

- Configuration Function (CF): This function enables the configuration of the capabilities and features of the CSE (e.g. CMDH policies).
- Software Management Function (SMF): This function provides lifecycle management for software components and associated artifacts (e.g. configuration files) for different entities such as CSE and AE.

6.2.1.2.1 Software Management Function

The Software Management Function (SMF) provides the capability to manage software components (e.g. Software Package, Software Module) for AEs and CSEs.

The ASM CSF provides the capability to manage the lifecycle of the Software Packages for a CSE or an AE. AE Software Packages may be deployed on any Node that supports the AE; including those on the MNs, ADNs and ASNs.

The lifecycle of a Software Package consists of states (e.g. Installing, Installed, Updating, Uninstalling and Uninstalled) that transition when an action (e.g. Download, Install, Update and Remove) is applied to the Software Package.

When a Software Package is installed into an execution environment the software component that is capable of executing in the Execution Environment is called a Software Module. The lifecycle of a Software Module consists of states (e.g. Idle, Starting, Active, Stopping) that transition when an action (e.g. Start, Stop) is applied to the Software Module.

6.2.2 Communication Management and Delivery Handling

6.2.2.1 General Concepts

The Communication Management and Delivery Handling (CMDH) CSF provides communications with other CSEs, AEs and NSEs.

The CMDH CSF decides at what time to use which communication connection for delivering communications (e.g. CSE-to-CSE communications) and, when needed and allowed, to buffer communication requests so that they can be forwarded at a later time. This processing in the CMDH CSF is carried out per the provisioned CMDH policies and delivery handling parameters that can be specific to each request for communication.

For communication using the Underlying Network data transport services, the Underlying Network can support the equivalent delivery handling functionality. In such case the CMDH CSF uses the Underlying Network, and it may act as a front end to access the Underlying Network equivalent delivery handling functionality.

6.2.2.2 Detailed Descriptions

The service that AEs or CSEs can request from the CMDH CSF is to transport some data to a specific target (CSE or AE), according to given delivery parameters while staying within the constraints of provisioned communication management and delivery handling policies.

The content of the data provided by the Originator does not influence the CMDH CSF behaviour. Consequently, the CMDH CSF is not aware of the specific operation requested at the target entity, including the parameters passed to the operation at the destination CSF. This means that all attributes intended to be delivered to the destination entity (e.g. which CSF is the destination on the target entity, what that CSF does with the data, etc.) are hidden to the CMDH CSF.

The target entity may be reached either directly or via the CSE(s) of a MN(s).

As part of the delivery request, the CMDH CSF can be provided with acceptable delivery parameters for the Originator (e.g. acceptable expiration time for delivery).

The functions supported by the CMDH CSF are as follows:

- Ability for the M2M Service Provider to derive CMDH policies describing details for the usage of the specific Underlying Network(s). These policies may be based on the M2M Service Subscription associated with Application and Common Service Entities (AEs and CSEs) in the Field Domain and on the agreements on usage of Underlying Network communication resources. CMDH Policies can be provisioned into the respective CSEs in the Field Domain.
- For the delivery of communication, ability to select appropriate communication path to use at any given time in line with provisioned CMDH policies and with CMDH-related parameters set by the Originator of requests, and when needed and allowed, how long to buffer communication requests so that they can be forwarded at a later time. This policy-driven use of communication resources allows an M2M Service Provider to control which Originators of requests are allowed to consume communication resources at certain times.
- For the delivery of communication, ability to detect a disconnection of communication channel. If the communication channel can be established by receiver-side only, it also detects a reconnection of the communication channel.
- For the delivery of communication, ability to be aware of the availability of the Underlying Networks.
- Ability to manage the proper use of buffers for store-and-forward processing through use of CMDH policies.

6.2.3 Data Management and Repository

6.2.3.1 General Concepts

One of the purposes of CSEs is to enable AEs to exchange data with each other.

The Data Management and Repository (DMR) CSF is responsible for providing data storage and mediation functions. It includes the capability of collecting data for the purpose of aggregating large amounts of data, converting this data into a specified format, and storing it for analytics and semantic processing. The data can be either raw data transparently retrieved from an M2M Device, or processed data which is calculated and/or aggregated by M2M entities.

NOTE: Collection of large amounts of data is known as the Big Data Repository and is not part of the present document.

6.2.3.2 Detailed Descriptions

The DMR CSF provides the capability to store data such as Application data, subscriber information, location information, device information, semantic information, communication status, access permission, etc. The data stored by the DMR CSF enables management of the data and provides the foundation of Big Data.

The following are examples of DMR CSF functionalities:

- Ability to store data in an organized fashion so it is discernible. This includes storage of contextual information such as data types, semantic information, time stamps, location, etc., to complement the data stored in order to access and search the data based on a set of parameters. This is part of data semantics capability which is not part of the present document.
- Provides the means to aggregate data received from different entities.
- Ability to grant access to data from remote CSEs and AEs based on defined access control policies, and trigger data processing based on data access.
- Ability to provide the means to perform data analytics on large amount of data to allow service providers to provide value-added services.

6.2.4 Device Management

6.2.4.1 General Concepts

6.2.4.1.0 Overview

The Device Management (DMG) CSF provides management of device capabilities on MNs (e.g. M2M Gateways), ASNs and ADNs (e.g. M2M Devices), as well as devices that reside within an M2M Area Network. Application Entities (AE) can manage the device capabilities on those Nodes by using the services provided by the DMG CSF alleviating the need for the AE to have knowledge of the technology specific protocols or data models. While the AE does not require an understanding of the technology specific protocols or data models, this information is provided to the AE so that an AE can utilize this information for administrative purposes (e.g. diagnostics, troubleshooting).

In order to manage the CSE and device capabilities of the MNs, ASNs and ADNs, the DMG can utilize existing technology specific protocols (e.g. BBF TR-069 [i.2], OMA-DM [i.3], and LWM2M [i.4]) in addition to resource operations across the Mcc and Mca reference points.

When non-oneM2M protocols are used to manage oneM2M Nodes the DMG of an IN or MN translates or adapts the management related oneM2M requests to/from the corresponding technology via a Management Adapter. The existing technology then supports operations between Management Servers and Management Clients. Architectural details regarding the use of non-oneM2M technology protocols is provided in clause 6.2.4.1.1.

The architectural model for the native Device Management uses the generic oneM2M architecture and reference points.

Both Device Management options (native oneM2M or non-oneM2M) use resources maintaining information and relationships that are specific to Device Management (i.e. Device Management Resources), as well as general purpose resources.

Device Management Resources maintain information and relationships used to:

- Manage technology specific data model objects via a Management Server which requires the information necessary to identify and access the Management Server.
- Invoke the security mechanism of the Management Server in order to authorize access to the technology specific data model objects.

Procedures for managing Device Management Resources are further detailed in clause 10.2.8 and apply to both Device Management options. For Device Management using external technologies, at most one Management Server is able to Create, Delete or Update addressable elements of a Management Resource.

6.2.4.1.1 Device Management using other existing technologies

6.2.4.1.1.1 Architecture

When non-oneM2M technologies are used to manage devices oneM2M resource operations need to be adapted to the specific protocol used (e.g. BBF TR-069 [i.2], OMA-DM [i.3], and LWM2M [i.4]). In order to perform the translation and adaptation functions, the DMG has a functional component termed the Management Adapter (figure 6.2.4.1.1.1-1). The Management Adapter in the DMG of the management server hosting M2M Node (e.g. IN-DMG-MA) performs the adaptation between the DMG and Management Servers using the **ms** interface; while the Management Adapter in the DMG of the management client hosting M2M Node (e.g. MN-DMG-MA or ASN-DMG-MA) performs translation and adaptation between the DMG and the Management Client using the **la** interface. Only one Management Adapter is shown in the DMG although it can interact with Management Server using different technology specific protocols.

The interface between Management Server and Management Client (figure 6.2.4.1.1.1-1) is the **mc** interface which is subject to the technology specific protocol that is used (e.g. BBF TR-069 [i.2] or LWM2M [i.4]). The **mc** interface is technology dependent and is outside the scope of the present document.

The DMG in the MNs or ASNs can be used to manage devices in the M2M Area Network. In this case, the DMG is deployed with proxy functionality that interacts with the Proxy Management Client using the **mp** interface. The **mp** interface is technology dependent and is outside the scope of the present document.

The Management Server and Management Client can be implemented as an entity external to the Node or they can be implemented as an entity embedded within the Node (figure 6.2.4.1.1.1-1). The Management Server and the Management Client are located on the boundary of the Node to indicate this situation as well as to depict that an IN can utilize multiple Management Servers from various M2M and Network Service Providers.

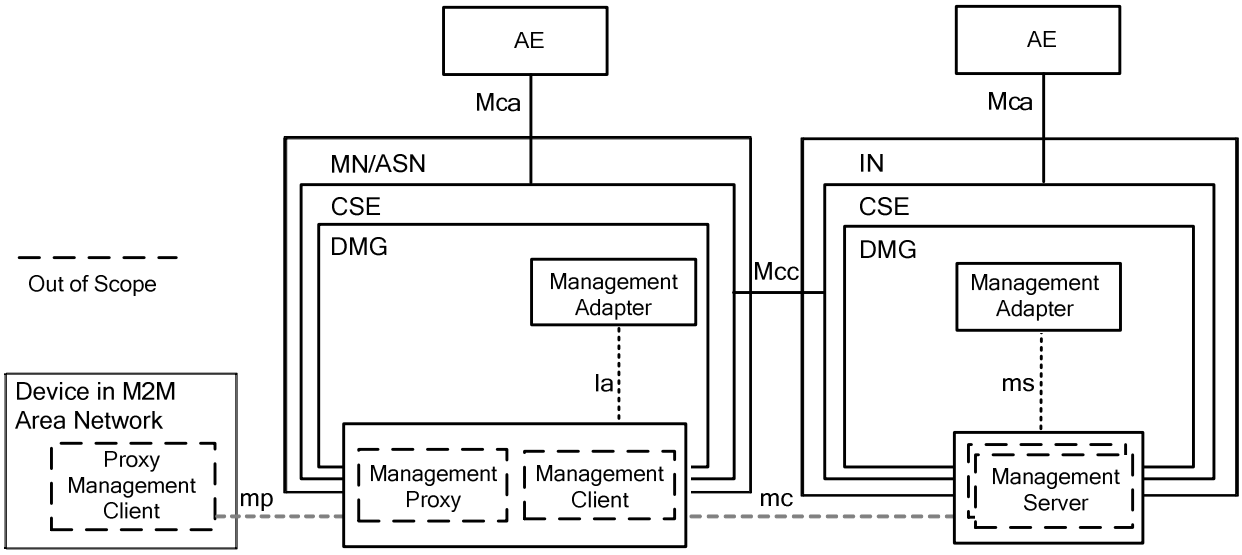


Figure 6.2.4.1.1.1-1: Device Management Architecture

6.2.4.1.1.2 Management Server Interaction

The DMG CSF in the IN has the capability to utilize Management Servers from technology specific protocols (e.g. BBF TR-069 [i.2], OMA DM [i.3], LWM2M [i.4]) to implement the Device Management functions. The Management Adapter in the DMG of the management server hosting M2M Node (e.g. IN-DMG-MA) communicates with the Management Server using the **ms** interface of the Management Server. Note that **ms** interface is outside the scope of the present document. The IN-DMG-MA takes the following roles:

- Protocol Translation between DMG and the Management Server:
  - After the DMG receives the requests from the request Originator, the Management Adapter in the DMG of the management server hosting M2M Node (e.g. IN-DMG-MA) translates the requests from the request Originator to requests with associated identifiers that can be understood by the Management Server. Likewise, the Management Adapter in the DMG of the management server Host (e.g. IN-DMG-MA) translates events from the Management Server and delivers the events to M2M Entities (e.g. AE, CSE) that are subscribed to the event. When the Management Server is embedded within the DMG, the Management Adapter translates the request and accepts events in the protocol understood by the Management Client.
- Interaction with the Management Server:
  - By using **ms** interface, the Management Adapter in the DMG of the management server hosting M2M Node (e.g. IN-DMG-MA) can communicate with the Management Server. This is for delivering the requests from the request Originator to the Management Server, or receiving information from the Management Server that will be notified to subscribing M2M Entities (e.g. AE, CSE). The communication between the Management Adapter and the Management Server requires an establishment of a session which provides security dimensions for Access Control, Authentication, Non-repudiation, Data confidentiality, Communication security, Data integrity and Privacy. The Management Adapter in the DMG of the management server Host (e.g. IN-DMG-MA) can utilize a policy that defines when a session with the Management Server is established and torn down.
- Management Server selection:
  - When the Management Adapter in the DMG of the management server hosting M2M Node (e.g. IN-DMG-MA) communicates with multiple Management Servers that have different level of access control privileges to resources from the Management Server, the Management Adapter selects the proper Management Server that has the access control privileges to perform the management requests. The access control policy information for resources from Management Servers may be discovered using the **ms** interface.
- Discovery of technology specific data model objects:
  - When the Management Adapter in the DMG of the management server Host (e.g. IN-DMG-MA) maintains information (i.e. metadata, values) of the technology specific data model objects managed by a Management Server using the **ms** interface, the Management Adapter will be capable of discovering and keep up to date the technology specific data model object's information that are managed by the DMG and a Management Server.

A Management Server can be located in the Underlying Network using the Mcn reference point as depicted in figure 6.2.4.1.1.2-1 or the Management Server can be located in the M2M Service Layer as depicted in figure 6.2.4.1.1.2-2.

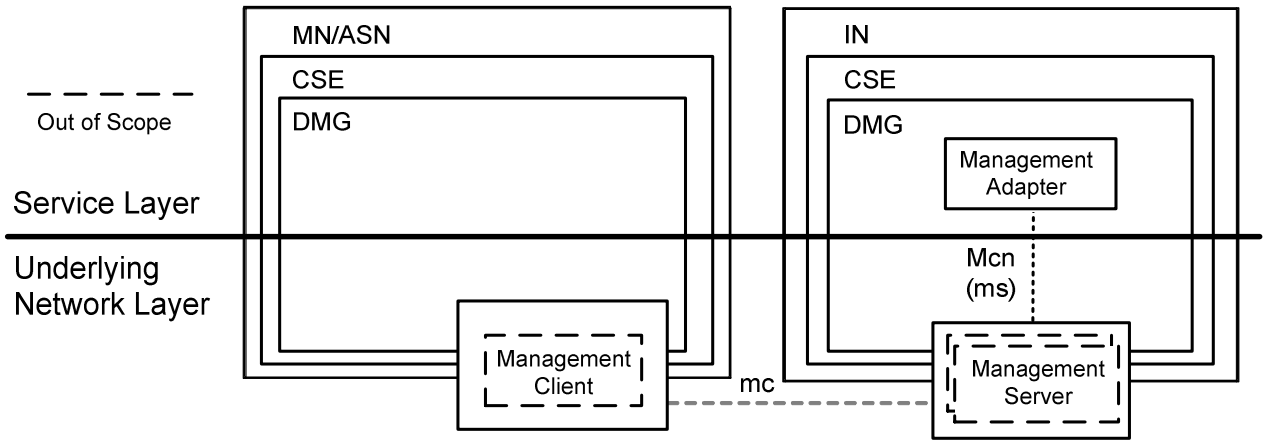


Figure 6.2.4.1.1.2-1: Management Server in Underlying Network

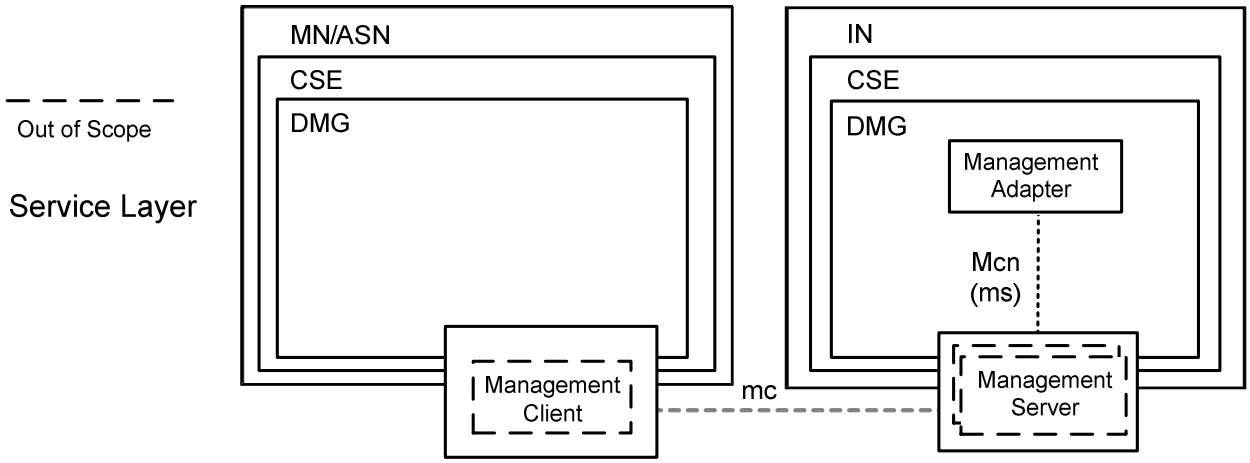


Figure 6.2.4.1.1.2-2: Management Server in M2M Service Layer

The **ms** interface is functionally the same interface regardless if the Management Server resides in the Underlying Network or the Service Layer. However, the access control privileges that the Management Server has for resources from the technology specific protocol can be different depending whether the Management Server resides in the Underlying Network or in the Services Layer. For example, in figure 6.2.4.1.1.2-1, the Management Server in the Underlying Network controls access of the exposed resources from the technology specific protocol, while, in the figure 6.2.4.1.1.2-2, the Management Server in the M2M Service Layer controls access to the resources.

6.2.4.1.1.3 Management Server - Access Permissions

When an operation on an M2M Service Layer Resource is performed and if the access to the Resource is granted and the operation for the Resource utilizes a Management Server external to the service layer, the DMG CSF of the management server Host selects one or more among the authenticated Management Servers necessary to access the requested resources. The procedure for the selection of Management Servers is implementation specific and outside the scope of the present document.

The DMG CSF management functions that cause impacts to the Underlying Network utilize access permissions that are delegated from the provider of the network service layer.

6.2.4.1.1.4 Management Server - External management object discovery

The Management Adapter of the Management Server Host (e.g. IN-DMG-MA) discovers information of the technology specific data model objects managed by a Management Server using the **ms** interface. The discovery of this information includes the:

- M2M devices, devices in the M2M Area Network and M2M Applications to which the Management Server has access.

- The metadata associated with the technology specific data model objects associated the M2M devices, devices in the M2M Area Network and M2M Applications. This metadata includes items such as the supported data/object model.

The Management Adapter of the Management Server Host (e.g. IN-DMG-MA) is capable of being kept up-to-date of the changes in the M2M Devices, devices in the M2M Area Network and M2M Applications or the metadata of the technology specific data model objects associated with those entities. In addition, the Management Adapter of the Management Server Host can maintain the value associated technology specific data model objects, associated the M2M devices, devices in the M2M Network and M2M Applications.

6.2.4.1.1.5 Management Client Interaction

The DMG CSF in the Management Client Host (e.g. MN or ASN) can use the Management Client from existing management technologies (e.g. BBF TR-069 [i.2], OMA DM [i.3], LWM2M [i.4]) to implement the Device Management functions. The Management Adapter in the Management Client Host (e.g. MN-DMG-MA, ASN-DMG-MA) communicates with the Management Client using the **la** interface (e.g. DM-7, 8, 9 ClientAPI in OMA DM [i.3]) that is provided by the Management Client. Note that the **la** interface is outside the scope of the present document. The Management Adapter in the Management Client Host takes the following roles:

- Interaction with the Management Client:
  - By using **la** interface, the Management Adapter can communicate with the Management Client to discover the technology specific data model objects supported by the Management Client.
- Mapping between the DMG and Management Client:
  - After the Management Adapter discovers the technology specific data model objects supported by the Management Client; the Management Adapter performs the mapping between the technology specific data model objects to resources. The DMG in the Management Client Host can create those resources in the Management Server hosting CSE, and the resources can be used by the device management AE to manage the device capabilities pertaining to the managed node.

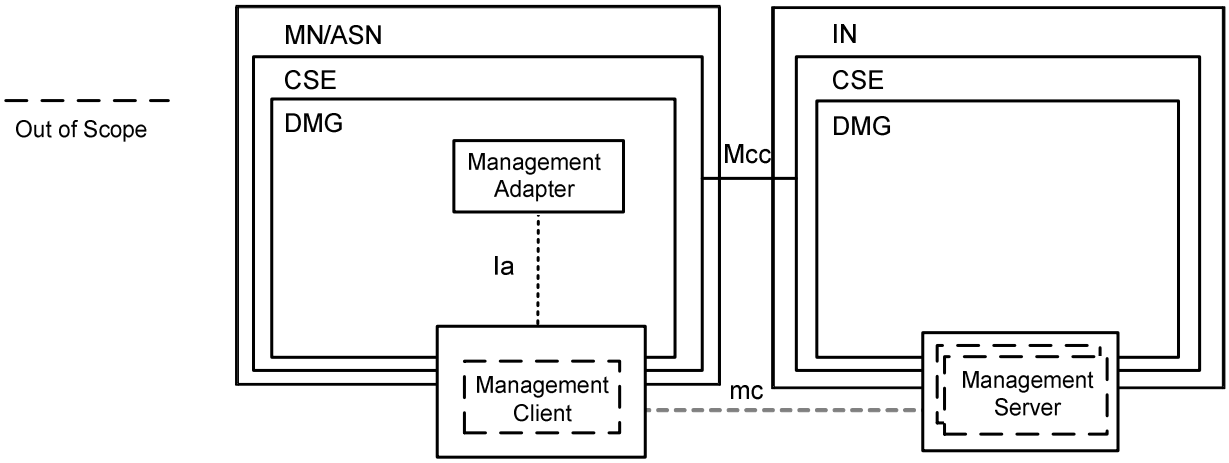


Figure 6.2.4.1.1.5-1: Management Client Interaction using "la" interface

6.2.4.2 Detailed Descriptions

6.2.4.2.0 Overview

The DMG CSF provides capabilities for the purpose of managing M2M Devices/Gateways as well as devices in M2M Area Networks.

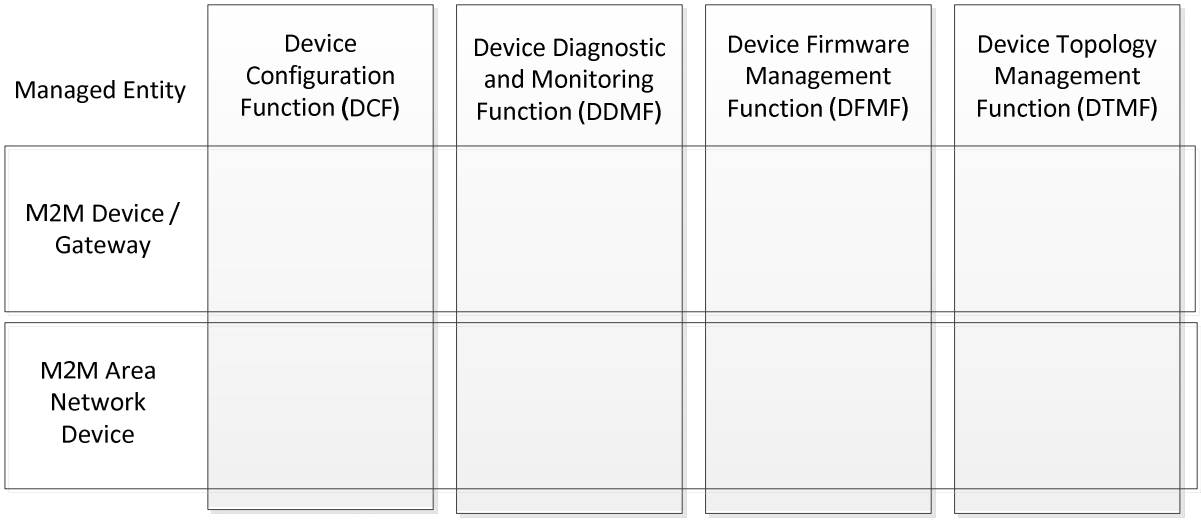


Figure 6.2.4.2.0-1: Device Management Entities and Functions

Such capabilities include:

- Device Configuration Function (DCF): This function includes the configuration of the capabilities of the M2M Device, M2M Gateway or device in the M2M Area Network.
- Device Diagnostics and Monitoring Function (DDMF): This function includes the troubleshooting through the use of diagnostic tests and retrieval of operational status and statistics associated with the M2M Device, M2M Gateway or device in the M2M Area Network.
- Device Firmware Management Function (DFMF): This function provides the software lifecycle management for firmware components and associated artefacts for the M2M Device, M2M Gateway or device in the M2M Area Network.
- Device Topology Management Function (DTMF): This function provides the management of the topology of the M2M Area Network. An M2M Area Network is comprised of ADNs and other devices in the M2M Area Network.

6.2.4.2.1 Device Configuration Function

The Device Configuration Function (DCF) provides the configuration of device capabilities that are necessary to support M2M Services and AEs in M2M Devices, M2M Gateways or devices in an M2M Area Network.

These device configuration capabilities include:

- Discovery of a device's management objects and attributes.
- Ability to enable or disable a device capability.
- Provisioning configuration parameters of a device.

6.2.4.2.2 Device Diagnostics and Monitoring Function

The Device Diagnostics and Monitoring Function (DDMF) permits the troubleshooting of device capabilities that are necessary to support M2M Services and AEs in M2M Devices, M2M Gateways or devices in an M2M Area Network.

These device diagnostic and monitoring capabilities include:

- Configuration of diagnostics and monitoring parameters on the device.
- Retrieval of device information that identifies a device and its model and manufacturer.
- Retrieval of device information for the software and firmware installed on the device.
- Retrieval of information related to a battery within the device.

- Retrieval of information associated with the memory in use by a device.
- Retrieval of the event logs from a device.
- Device reboot diagnostic operation.
- Device factory reset diagnostic operation.

6.2.4.2.3 Device Firmware Management Function

The Device Firmware Management Function (DFMF) provides lifecycle management for firmware associated with a device.

Device firmware is comprised of firmware modules and artefacts (e.g. configuration files) that are maintained on a device. A device can maintain more than one firmware image and the capability to manage individual firmware images. The firmware lifecycle includes actions to download, update or remove a firmware image. In addition, firmware could be downloaded and updated within the same action.

6.2.4.2.4 Device Topology Management Function

The Device Topology Management Function (DTMF) is a function that is specific to M2M Gateways where an M2M Gateway maintains zero or more M2M Area Networks.

These device topology management capabilities include:

- Configuration of the topology of the M2M Area Network.
- Retrieval of information related to the devices attached to the M2M Area Network.
- Retrieval of information that describes the transport protocol associated with the M2M Area Network.
- Retrieval of information that describes the characteristics associated with online/offline status of devices in the M2M Area Network.

6.2.5 Discovery

6.2.5.1 General Concepts

The Discovery (DIS) CSF searches information about applications and services as contained in attributes and resources. The result of a discovery request from an Originator depends upon the filter criteria and is subject to access control policy allowed by M2M Service Subscription. An Originator could be an AE or another CSE. The scope of the search could be within one CSE, or in more than one CSE. The discovery results are returned back to the Originator.

6.2.5.2 Detailed Descriptions

The DIS CSF uses the Originator provided filter criteria (e.g. a combination of keywords, identifiers, location and semantic information) that can limit the scope of information returned to the Originator.

The discovery request indicates the address of the resource where the discovery is to be performed. Upon receiving such request, the DIS CSF discovers, identifies, and returns the matching information regarding discovered resources according to the filter criteria.

A successful response includes the discovered information or address(es) pertaining to the discovered resources. In the latter case the Originator can retrieve the resources using such discovered address. Based on the policies or Originator request, the CSE which received the discovery request can forward the request to other registered ASN-CSEs, MN-CSEs or IN-CSEs.

6.2.6 Group Management

6.2.6.1 General Concepts

The Group Management (GMG) CSF is responsible for handling group related requests. The request is sent to manage a group and its membership as well as for the bulk operations supported by the group. When adding or removing members to/from a group, it is necessary to validate whether the group member complies with the purpose of the group. Bulk operations include read, write, subscribe, notify, device management, etc. Whenever a request or a subscription is made via the group, the group is responsible for aggregating its responses and notifications. The members of a group can have the same role with regards to access control policy control towards a resource. In this case, access control is facilitated by grouping. When the Underlying Network provides broadcasting and multicasting capability, the GMG CSF is able to utilize such capability.

6.2.6.2 Detailed Descriptions

The GMG CSF enables the M2M System to perform bulk operations on multiple devices, applications or resources that are part of a group. In addition, the GMG CSF supports bulk operations to multiple resources of interest and aggregates the results. It facilitates access control based on grouping. When needed and available, the GMG CSF can leverage the existing capabilities of the Underlying Network including broadcasting/multicasting.

When facilitating access control using a group, only members with the same access control policy towards a resource are included in the same group. Also, only AEs or CSEs which have a common role with regards to access control policy are included in the same group. This is used as a representation of the role when facilitating role based access control.

The service functions supported by the GMG CSF are as follows:

- Handles the requests to create, retrieve, update, and delete a group. An AE or a CSE may request the creation/retrieve/update/deletion of a group as well as the addition and deletion of members of the group.
- Creates one or more groups in CSEs in any of the Nodes in oneM2M System for a particular purpose (e.g. facilitation of access control, device management, fan-out common operations to a group of devices, etc.).
- Handles the requests to retrieve the information (e.g. address, metadata, etc.) of a group and its associated members.
- Manages group membership and handles requests to add or remove members to and from a group's member list. A member may belong to one or more groups. A group may be a member of another group. When new members are added to a group, the GMG CSF validates if the member complies with the purpose of the group.
- Leverages the capabilities of other CSFs in order to fulfill the functionalities supported by the GMG CSF service functions. Examples include: Security CSF for authentication and authorization.
- Forwards requests to all members in the group. In case the group contains another group as a member, the forwarding process is done recursively, i.e. the nested group forwards the request to its members. After forwarding the request to all members in the group, the GMG CSF generates an aggregated response by aggregating the corresponding responses from the Group members.
- Supports subscriptions to individual groups. Subscriptions to a group is made only if the subscriber is interested in all members of the group. If subscription to a group is made, the GMG CSF aggregates the notifications from the group members, and notifies the subscriber with the aggregated notification. Responses and event notifications relevant to a subscription may be selectively filtered by filtering criteria.

6.2.7 Location

6.2.7.1 General Concepts

The Location (LOC) CSF allows AEs to obtain geographical location information of Nodes (e.g. ASN, MN) for location-based services. Such location information requests can be from an AE residing on either a local Node or a remote Node.

NOTE: Geographical location information can include more than simply the longitude and the latitude information.

6.2.7.2 Detailed Descriptions

The LOC CSF obtains and manages geographical location information based on requests from AEs residing on either a local Node or a remote Node. The LOC CSF interacts with any of the following:

- a location server in the Underlying Network;
- a GPS module in an M2M device; or
- information for inferring location stored in other Nodes.

In order to update the location information, an AE can configure an attribute (e.g. update period). Based on such defined attributes, the LOC CSF can update the location information using one of the location retrieval mechanisms listed above.

NOTE: The location technology (e.g. Cell-ID, assisted-GPS, and fingerprint) used by the Underlying Network depends on its capabilities.

The functions supported by the LOC CSF are as follows:

- Requests other Nodes to share and report their own or other Nodes' geographical location information with the requesting AEs.
- Provides means for protecting the confidentiality of geographical location information.

6.2.8 Network Service Exposure, Service Execution and Triggering

6.2.8.1 General Concepts

The Network Service Exposure, Service Execution and Triggering (NSSE) CSF manages communications with the Underlying Networks for accessing network service functions over the Mcn reference point. The NSSE CSF uses the available/supported methods for service "requests" on behalf of AEs. The NSSE CSF shields other CSFs and AEs from the specific technologies and mechanisms supported by the Underlying Networks.

NOTE: The NSSE CSF provides adaptation for different sets of network service functions supported by various Underlying Networks.

The network service functions provided by the Underlying Network include service functions such as, but not limited to, device triggering, small data transmission, location notification, policy rules setting, location queries, IMS services, device management. Such services do not include the general transport services.

6.2.8.2 Detailed Descriptions

The NSSE CSF manages communication with the Underlying Networks for obtaining network service functions on behalf of other CSFs, remote CSEs or AEs. The NSSE CSF uses the Mcn reference point for communicating with the Underlying Networks.

The M2M System allows the Underlying Networks to control network service procedures and information exchange over the Underlying Networks while providing such network services. For example, some Underlying Network can choose to provide the network services based on control plane signalling mechanisms.

Other CSFs in a CSE that need to use the services offered by the Underlying Network use the NSSE CSF.

The service functions supported by the NSSE CSF are as follows:

- The NSSE CSF shields other CSFs and AEs from the specific technology and mechanisms supported by the Underlying Networks.

NOTE: The NSSE CSF provides adaptation for different sets of network service functions supported by various Underlying Networks.

- The NSSE CSF maintains the necessary connections and/or sessions over the Mcn reference point, between the CSE and the Underlying Network when local CSFs are in need of a network service.
- The NSSE CSF provides information to the CMDH CSF related to the Underlying Network so the CMDH CSF can include that information to determine proper communication handling.

6.2.9 Registration

6.2.9.1 General Concepts

The Registration (REG) CSF processes a request from an AE or another CSE to register with a Registrar CSE in order to allow the registered entities to use the services offered by the Registrar CSE.

6.2.9.2 Detailed Descriptions

Registration is the process of delivering AE or CSE information to another CSE in order to use M2M Services.

An AE on an ASN, an MN or an IN performs registration locally with the corresponding CSE in order to use M2M services offered by that CSE. An AE on an ADN performs registration with the CSE on an MN or an IN in order to use M2M services offered by that CSE. An IN-AE performs registration with the corresponding CSE on an IN in order to use M2M services offered by that IN CSE. An AE can have interactions with its Registrar CSE (when it is the target CSE) without the need to have the Registrar CSE register with other CSE.

The CSE on an ASN performs registration with the CSE in the MN in order to be able to use M2M Services offered by the CSE in the MN. As a result of successful ASN-CSE registration with the MN-CSE, the CSEs on the ASN and the MN establish a relationship allowing them to exchange information.

The CSE on an MN performs registration with the CSE of another MN in order to be able to use M2M Services offered by the CSE in the other MN. As a result of successful MN-CSE registration with the other MN-CSE, the CSEs on the MNs establish a relationship allowing them to exchange information.

The CSE on an ASN or on an MN perform registration with the CSE in the IN in order to be able to use M2M Services offered by the CSE in the IN. As a result of successful ASN/MN registration with the IN-CSE, the CSEs on ASN/MN and IN establish a relationship allowing them to exchange information.

Following a successful registration of an AE to a CSE, the AE is able to access, assuming access privilege is granted, the resources in all the CSEs that are potential targets of request from the Registrar CSE.

The capabilities supported by the REG CSF are as follows:

- ability for AE to register to its Registrar CSE where the hop count is zero, as per table 6.4-1;
- ability for CSE to register to its Registrar CSE where the hop count is zero, as per table 6.4-1;
- ability for an ASN-CSE/MN-CSE or ADN-AE to register association of its M2M-Ext-ID (if available) with its CSE-ID or AE-ID (see clause 7.1.8);
- ability for an ASN-CSE/MN-CSE or ADN-AE to register association of its Trigger-Recipient-ID (if available) with its CSE-ID or AE-ID (see clause 7.1.10). When Trigger-Recipient-ID is not present, it is assumed that the ADN-AE or CSE is not able to receive triggers.

NOTE: Such registrations are applicable to a single M2M Service Provider Domain.

Registration information for a Node includes:

- Identifier of the Node.
- Reachability schedules; which are elements of a Node's policy, and specify when messaging can occur between Nodes. Reachability schedules can be used in conjunction with other policy elements. When reachability schedules are not present in a Node then that Node is expected to be always reachable.
- Managing connection state of communication channel to the registered AE or CSE.

6.2.10 Security

6.2.10.1 General Concepts

The Security (SEC) CSF comprises the following functionalities:

- Sensitive data handling;
- Security administration;
- Security association establishment;
- Remote security provisioning;
- Identification and authentication;
- Authorization;
- Identity management.

Sensitive data handling functionality in the SEC CSF protects the local credentials on which security relies during storage and manipulation. Sensitive data handling functionality performs other sensitive functions such as security algorithms. This functionality is able to support several cryptographically separated security environments. Those secure environments are accessible via the Mcs reference point. This reference point abstracts different types of secure environments and is defined in ETSI TS 118 116 'Secure Environment Abstraction' [9].

Security management capabilities are provided by the Security Administration functionality as specified in ETSI TS 118 103 [2].

NOTE: ASM and DMG CSFs do not include security management capabilities of the SEC CSF.

Security administration functionality enables services such as the following:

- Creation and administration of dedicated security environment supported by Sensitive Data Handling functionality.
- Post-provisioning of a root credential protected by the security environment.
- Provisioning and administration of subscriptions related to M2M Common Services and M2M Application Services.

Security association establishment functionality establishes security association between corresponding M2M Nodes, in order to provide services such as confidentiality and integrity.

6.2.10.2 Detailed Descriptions

The functionalities supported by the SEC CSF are as follows:

- Sensitive data handling:
  - Provides the capability to protect the local credentials on which security relies during storage and manipulation.
  - Extends sensitive data handling functionality to other sensitive data used in the M2M Systems such as subscription related information, access control policies and personal data pertaining to individuals.
  - Performs other sensitive functions as well, such as security algorithms running in cryptographically separated secure environments.
- Security administration:
  - Creates and administers dedicated secure environment supported by sensitive data handling functionality.
  - Post-provisions master credentials protected by the secure environment.

NOTE 1: The secure environment can also be pre-provisioned with a master credentials prior to deployment; therefore this capability is not always required. Post-provisioning is required when secure remote provisioning needs to be performed or re-initiated after deployment.

- Provisioning and administration of subscriptions related to M2M Services and M2M application services. Besides the associated master credentials, a subscription includes other information classified as sensitive data such as authorization roles and identifiers for access control management.

- Security association establishment:

  - Establishes security associations between corresponding M2M Nodes in order to provide specific security services (e.g. confidentiality, integrity, or support for application level signature generation and verification) involving specified security algorithms and sensitive data. This involves key derivation based on provisioned master credentials. This functionality of the SEC CSF is mandatory when security is supported.

- MAF-based security association establishment:

  - These security frameworks use a M2M Authentication Function (MAF) to provide authentication and distribution of symmetric key for use by a Source End-Point initiating establishment of the symmetric key, and one or more target end-points. The symmetric key can be used in one of Security Association Establishment Framework, End-to-end security of Data (ESData) and End-to-end security of Primitives (ESPrim).

- End-to-end security of Data (ESData) and Primitives (ESPrim):

  - End-to-End Security of Primitives (ESPrim) allows a Hosting CSE or AE to authenticate the Originator of a request primitives that are handled by other CSEs. ESPrim also provides confidentiality and integrity protection of these request and response primitives. End-to-End Security of Data (ESData) provides an interoperable framework for protecting data such that it can be transported via transit CSEs which do not need to be trusted.

- Remote Security Provisioning:

  - Remote Security Provisioning Frameworks (RSPFs) enable an M2M Enrolment Function (MEF) to provision credentials to an Enrolee, which is a CSE or AE, as part of the Enrolment of the Enrolee to an M2M SP or third party M2M Trust Enabler (MTE). The credentials are either a symmetric key shared by the Enrolee and an Enrolment Target or Certificate(s) for which the Enrolee knows the corresponding private key, and a set of trust anchors for authenticating the M2M SP or MTE's MAF or other entities enrolled with the M2M SP or MTE.

- Authorization:

  - Role Based Access Control (RBAC) allows the Hosting CSE to authorize accesses on resources according to the roles assigned to the Originators.

  - Token Based Access Control allows the Hosting CSE to authorize accesses on resources according to the authorization information in tokens provided by the Originators.

  - Dynamic Authorization provides an interoperable framework for an Originator to be dynamically issued with temporary permissions providing the Originator with access to one or more resources on one or more CSEs at runtime.

  - The entire authorization function can be split into four sub-functions: Policy Enforcement Point (PEP), Policy Decision Point (PDP), Policy Retrieval Point (PRP) and Policy Information Point (PIP). Distributed Authorization provides an interoperable framework which allows PEP, PDP, PRP and PIP to be distributed in different CSEs.

  - Privacy Policy Manager (PPM) provides a standardized list of privacy attribute value pairs, automatic comparison of a user's privacy preferences with applications privacy policies and management of related access control policies.

- Identity management:
  - Identity management in the oneM2M context covers the lifecycle (creation, storage and destruction) of identifiers related to oneM2M entities.

NOTE 2: This functionality is not part of the present release.

Detailed functionalities are described in the ETSI TS 118 103 [2].

Sensitive security functions and information within a node are protected by local Secure Environments (SE). A Secure Environment is an abstraction of a secure area, within a computing system on a node (ADN, ASN, MN or IN), that provides a defined level of protection for code and data at rest, i.e. in storage, and in use, i.e. during process execution or data manipulation, as specified in ETSI TS 118 116 [9]. An SE provides resources for the purposes described above that can be manipulated via the Mcs reference point. Details on the SE resources can be found in ETSI TS 118 116 [9].

6.2.11 Service Charging and Accounting

6.2.11.1 General Concepts

The Service Charging and Accounting (SCA) CSF provides charging functions for the Service Layer. It supports different charging models which also include online real time credit control. The SCA CSF manages service layer charging policies and configuration capturing service layer chargeable events, generating charging records and charging information. The SCA CSF can interact with the charging System in the Underlying Network also. The SCA CSF in the IN-CSE handles the charging information.

6.2.11.2 Detailed Descriptions

The SCA CSF performs information recording corresponding to a chargeable event based on the configured charging policies. The SCA CSF sends the charging information transformed from the specific recorded information to the billing domain by the use of a standard or proprietary interface for charging purposes.

The SCA CSF supports "independent service layer charging" and "correlated charging with the Underlying Network" charging system. For independent service layer charging, only charging functions in the M2M service layer are involved. For correlated charging, charging functions in both the service layer and the Underlying Network are involved.

The SCA CSF supports one or multiple charging models, such as the following:

- Subscription based charging: A service subscriber is charged based on service layer subscriptions.
- Event based charging: Charging is based on service layer chargeable events. A chargeable event refers to the discrete transactions. For example, an operation on data (Create, Update, Retrieve) can be an event. Chargeable event can also be timer based. Chargeable events are configurable to initiate information recording. More than one chargeable event can be simultaneously configured and triggered for information recording.

The Service Layer charging system consists of the following logical functions:

- Charging management function: This function handles charging related policies, configurations, function communications and interacting with the charging system in the Underlying Network. Charging related policies.
- Charging triggering function: This function resides in the service layer. It captures the chargeable event and generates recorded information for charging. Recorded information may contain mandatory and optional elements.
- Offline charging function: This function handles offline charging related operations. Offline charging does not affect services provided in real time. Charging triggering information is generated at the CSFs where the chargeable transaction happens. The offline charging function generates service charging records based on recorded information. A service charging record is a formatted collection of information about a chargeable event (e.g. amount of data transferred) for use in billing and accounting.

NOTE: Charging triggering and offline charging function are based on charging policies. The system may record information for other purposes such as for event logging. Some of such information may be applicable for charging purposes.

6.2.12 Subscription and Notification

6.2.12.1 General Concepts

The Subscription and Notification (SUB) CSF provides notifications pertaining to a subscription that tracks event changes on a resource (e.g. deletion of a resource). A subscription to a resource is initiated by an AE or a CSE, and is granted by the Hosting CSE subject to access control policies. During an active resource subscription, the Hosting CSE sends a notification regarding a notification event to the address(es) where the resource subscriber wants to receive it.

6.2.12.2 Detailed Descriptions

The SUB CSF manages subscriptions to resources, subject to access control policies, and sends corresponding notifications to the address(es) where the resource subscribers want to receive them. An AE or a CSE is the subscription resource subscriber. AEs and CSEs subscribe to resources of other CSEs. A subscription Hosting CSE sends notifications to the address(es) specified by the resource subscriber when modifications to a resource are made. The scope of a resource subscription includes tracking changes of attribute(s) and direct child resource(s) of the subscribed-to resource. It does not include tracking the change of attribute(s) of the child resource(s). Furthermore, the scope includes tracking operations on attributes and direct child resources, but does not include tracking operations on attributes of child resources. Each subscription may include notification policies that specify which, when, and how notifications are sent. These notification policies may work in conjunction with CMDH policies.

A subscription is represented as resource subscription in the CSE resource structure.

The functions supported by the SUB CSF are as follows:

- Inclusion of the resource subscriber ID, the hosting CSE-ID and subscribed-to resource address(es) per resource subscription request. It may also include other criteria (e.g. resource modifications of interest and notification policy) and the address(es) where to send the notifications.
- Ability to subscribe to a single resource via a single subscription, or subscribe to multiple resources via a single subscription when they are grouped and represented as a single group resource.

6.2.13 Transaction Management

6.2.13.1 General Concepts

The Transaction Management (TMG) CSF assists applications with the atomic and consistent processing of oneM2M request primitives. The TMG CSF supports scheduling of a transaction, locking and unlocking of resources targeted by a transaction, the atomic and consistent execution of a transaction on targeted resources, the committal of successful transaction results, and the abort and rollback of non-successful transactions.

6.2.13.2 Detailed Descriptions

The TMG CSF uses an Originator provided transaction consisting of set of multiple oneM2M request primitives targeting multiple oneM2M resources. The targeted resources may reside on one or more CSEs. The TMG CSF handles the atomic and consistent processing of the set of request primitives such that all the requests are completed successfully. If one or more requests do not complete successfully, then the TMG CSF handles rolling back all the targeted resources to the state they were in prior to the transaction being processed.

NOTE: Rollback of a transaction after it is committed (commonly referred to as revoking or reversing) is not supported by the TMG CSF. Only rollback of a transaction before it is committed is supported.

The TMG CSF also supports the capability to schedule the execution of the oneM2M request primitives defined within a transaction. The TMG CSF may use an Originator specified transaction lock time, execution time and/or committal time to schedule when the processing of the transaction is to take place.

6.2.14 Semantics

6.2.14.1 General Concepts

The Semantics (SEM) CSF enables applications to manage semantic information and provides functionalities based on this information. Thus the SEM CSF brings value-added features related to the meaning of data and resources. The SEM CSF functionality is based on semantic descriptions and supports features such as: annotation, resource filtering and discovery, querying, validation, mash-up, reasoning, analytics, etc. The SEM CSF also provides input for Access Control applied to semantic content and is responsible for the management of ontologies.

6.2.14.2 Detailed Descriptions

The SEM CSF uses semantic descriptions pertaining to resources based on ontologies which annotate the corresponding resources. The SEM CSF handles the processes of discovery of resources and querying of semantic information, respectively, based on syntactic, semantic and structural information contained in semantic content data (such as RDF triples). When executing semantic operations and accessing RDF triple content, the SEM CSF uses access control information applicable to resources.

The SEM CSF enables also the creation, execution and result retrieval of functions based on semantic mashup, the validation of semantic content, and the use and management of ontologies.

6.3 Security Aspects

ETSI TR 118 508 [i.25] on Analysis of Security Solutions for the oneM2M System differentiates security domains related to the transport layer (Underlying Network), service layer (M2M common services) and Application Layer. It also considers possible trust scenarios involving these different security domains, and investigates countermeasures to threats that potentially affect the security of the M2M System.

Each of the security domains may provide their own set of security capabilities. The oneM2M security solution shall provide configurable security services through an API for upper security domains to leverage, or enable the use of the exposed security features of other security domains when appropriate.

As a result, beyond providing security solutions that protect the integrity of the M2M Service Layer, the oneM2M architecture exposes, through its APIs, further security services that are made available to M2M Applications. This enables M2M Applications to benefit from security solutions deployed in the M2M Service Architecture, without adding redundant and/or proprietary security solutions.

NOTE: It remains the responsibility of M2M Application Service Providers to perform their own risk assessment process to identifying the specific threats affecting them and derive their actual security needs.

Security aspects are described in ETSI TS 118 103 [2].

6.4 Intra-M2M SP Communication

Within the same SP domain, a CSE shall perform registration with another CSE over the Mcc reference point to be able to use M2M Services offered by that CSE and to allow the other CSE to use its services. As a result of successful registration the CSEs establish a relationship allowing them to exchange information.

An AE shall perform registration with a CSE in order to be able to use M2M Services offered by that CSE. As a result of successful AE registration, the AE and the CSE establish a relationship allowing them to exchange information.

Table 6.4-1 shows which oneM2M entity types shall be able to register with which other entity types.

Table 6.4-1: Entity Registration

Originator (Registree)	Receiver (Registrar)	Registration Procedure
ADN-AE	MN-CSE, IN-CSE	AE registration procedure see clause 10.2.2.2
ASN-AE	ASN-CSE	
MN-AE	MN-CSE	
IN-AE	IN-CSE	
ASN-CSE	MN-CSE, IN-CSE	CSE registration procedure see clause 10.2.2.7
MN-CSE	MN-CSE, IN-CSE	

The Originator (Registree) in table 6.4-1 requests the registration and the Receiver (Registrar) is responsible for verifying the request, and checking the authentication and authorization of the Originator in order to establish a peer relationship:

- An AE shall not be registered to more than one CSE (ASN-CSE, MN-CSE or IN-CSE).
- An ASN-CSE shall be able to be registered to at most one other CSE (MN-CSE or IN-CSE).
- An MN-CSE shall be able to be registered to at most one other CSE (MN-CSE or IN-CSE).

An MN-CSE shall be able to support only a single registration towards another MN-CSE or an IN-CSE. A concatenation (registration chain) of multiple uni-directional registrations shall not form a loop. E.g. two MN-CSEs A and B, cannot register with each other. Three MN-CSEs A, B and C, where A registers to B, and B registers to C, then C cannot register to A.

6.5 Inter-M2M SP Communication

6.5.1 Inter M2M SP Communication for oneM2M Compliant Nodes

6.5.1.0 Overview

To enable M2M entities (e.g. CSE, AE) in different M2M Service Provider (SP) domains to communicate, configuration within the M2M domain determines if such a communication is allowed. If allowed, the M2M System shall support routing of the traffic across the originating M2M SP domain and within the target M2M SP domain.

Communication between different M2M SPs which occurs over the reference point Mcc', is subject to business agreements. The offered functionality is typically a subset of the functionality offered over the Mcc reference point.

Any interM2M SP communication in support of a request originating from one M2M SP domain shall be processed and forwarded through the Infrastructure Node of the originating M2M domain towards the Infrastructure Node of the target M2M SP domain and finally forwarded to its target CSE, if different from the Infrastructure Node. Hence the Infrastructure Node in both M2M domains shall be the exit and entry points, respectively, for all inter M2M SP communication traffic.

In this configuration approach, public DNS shall be used to support traffic routing for inter M2M SP communication in accordance with [i.13]. This relies on public domain names being allocated to communicating CSE entities within the oneM2M architecture, and to whom access across domains is permitted through policies. To that effect, an M2M SP supporting inter- M2M SP communication shall ensure that the public domain names for the CSEs whose functionality is available across domains are held in its public DNS and shall always point to the IP address associated with the Infrastructure Node for the domain (being the entry point) for accessibility purposes.

The M2M SP could optionally also have additional policies (e.g. black list or white list) that governs accessibility from other domains to CSE functionality located within its own domain. These policies are however out of scope of the present document.

The public domain names of CSEs to whom access from other domains is allowed by policies, shall be created in the DNS of the M2M SP by the Infrastructure Node at registration time of these CSEs, and shall be removed at de-registration. DNS entries for CSEs can also be created/removed for registered CSEs at any time by the M2M SP through administrative means to handle dynamic policies.

6.5.1.1 Public Domain Names and CSEs

To enable the usage of public DNSs as described above, there is a need for a naming convention for public names for CSEs. This naming convention facilitates the creation of the necessary entries of the public domain names of CSEs in the DNS by the infrastructure node.

CSEs public domain names shall be a sub-domain of the Infrastructure Node's public domain name. This naming convention allows the Infrastructure Node to include the needed DNS entry corresponding to the CSE to whom access from other domains is allowed. This would typically occur when the CSE registers with the Infrastructure Node, subject to policies, or administratively.

Accordingly, the structure of the public domain of the CSEs in IN/MN/ASN shall follow the following naming convention, which relies on the CSE identifier (CSE-ID) as part of the naming convention to facilitate the DNS entry creation:

- Infrastructure Node CSE public domain name: <Infrastructure Node CSE Identifier>.<M2M Service Provider domain name>.
- Middle Node CSE public domain name: <Middle Node CSE Identifier>.<Infrastructure Node public domain name>.
- Application Service Node CSE public domain name: <Application Service Node CSE Identifier>.<Infrastructure Node public domain name>.

Both the MN-CSE and the ASN-CSE public domain names are sub-domains of the Infrastructure Node public domain name.

The A/AAAA records in the DNS, as per [i.7], [i.9] and [i.12] shall consist of the public domain name of the CSE and the IP address of the M2M Infrastructure Node, since the M2M Infrastructure Node is the entry point of the M2M Service Provider domain name where it belongs to.

Note that entries in the public domain names of the three nodes depicted above do not imply that the actual CSE-Identifier allocated for that node has to be used in the DNS entry. Rather any name, including indeed the CSE Identifier for the node, can be used there as long as the entry resolves to the intended Node.

EXAMPLE:

These 3 host entries are valid entries in the DNS:

- MN-CSEID.IN-CSEID.m2m.myoperator.org
- node1.node2.m2m.myoperator.org
- MN-CSEID.node22.m2m.myoperator.org

6.5.2 Inter M2M SP Generic Procedures

6.5.2.0 Overview

This clause describes the behaviour of the M2M Nodes in support of inter-M2M SP procedures.

6.5.2.1 Actions of the Originating M2M Node in the Originating Domain

The Originator in the originating domain can be any M2M Node such as ADN, an MN, or an ASN, and shall send a request to the Registrar CSE to retrieve a resource located in another M2M SP domain.

The Originator shall use any of the options defined in clause 9.3.1 to identify the target host and resource for that purpose.

6.5.2.2 Actions of the Receiving CSE in the Originating Domain

The receiving CSE in the originating domain shall check if the addressed resource is locally available. If the addressed resource is not locally available, then the request shall be forwarded to the next CSE.

If the receiving CSE is on an IN, it shall check if the addressed resource is locally available within its domain. If the addressed resource is not located within its own domain, then the IN shall perform a DNS lookup by using the target hostname provided in the RETRIEVE request. A successful DNS lookup shall return to the origin IN in the originating domain the IP address of the M2M IN residing in the target M2M SP domain.

Subsequently, the IN in the originating domain shall forward the request to the IN of the target domain.

6.5.2.3 Actions in the IN of the Target Domain

The IN is the entry point of the target M2M SP domain. The IN shall check if the addressed resource is a local resource. If it is not a local resource it shall forward the request to the appropriate CSE, after identifying the Hosting CSE within its domain, using the pointOfAccess attribute.

Once the request reaches the target Hosting CSE, the CSE shall apply the access control policies applicable to the request. Consequently, the Hosting CSE shall forward the response for the incoming request following the same path of the incoming request.

6.5.3 DNS Provisioning for Inter-M2M SP Communication

6.5.3.0 Overview

As specified previously, any M2M SP supporting inter-M2M SP communication shall ensure that the public domain names for the CSEs whose functionality is available across domains are held in the M2M SP's DNS and shall always point to the IP address associated with the Infrastructure domain CSE (being the entry point) for accessibility purposes.

This implies that the IN-CSE shall be responsible for creating the appropriate entry in the DNS for a successfully registered CSE in the IN-CSE, if the M2M SP policies do allow access to the CSE across multiple M2M domains. Similarly, the IN-CSE shall be responsible for deleting the appropriate entry in the M2M SP's DNS for a successfully de-registered CSE in the IN-CSE if the M2M SP policies do allow access to the CSE across multiple M2M domains.

6.5.3.1 Inter-M2M SP Communication Access Control Policies

Additional M2M SP policies that further restrict access to CSEs to requests originating from configured M2M SPs only, can complement the DNS entries created by the IN-CSE. These policies are out of scope of the present document.

6.5.4 Conditional Inter-M2M Service Provider CSE Registration

Inter-M2M Service Provider CSE registration shall be supported to enable M2M entities (e.g. CSE, AE) in peer M2M Service Provider (SP) domains with the ability to create and operate resources with the equivalent set of possibilities as offered in the intra-M2M Service Provider domain, subject to the following:

- The AE or CSE in either domain requires a representation of its own domain, notably the IN-CSE of its domain, in the peer domain to create resources in the peer domain. As an example, when it is required for an AE or a CSE to create and operate under the representation of an IN-CSE resource from a different M2M SP Domain. This enables the AE or CSE to have a behaviour that is identical in both the intra- and inter-M2M SP cases.

An AE or CSE that does not require to use the remoteCSE representations of the other domain as parent resources, can create resources in the peer domain if it knows the parent of the resource to be created and as such does not require IN to IN registration. Hence creating subscriptions within a peer M2M SP shall not require IN to IN registration between peer domains (but remains subject to inter -M2M SP business agreements, and access control policies).

Registration between M2M SPs occurs over the reference point Mcc', and is subject to business agreements. These agreements can limit the offered functionalities in comparison to those offered over the Mcc reference point.

No additional security is required respect to the basic procedure as described in clauses 6.5.1, 6.5.2 and 6.5.3.

Table 6.5.4-1 shows which oneM2M entity types can register with which other entity types across the Mcc' reference point.

Table 6.5.4-1: Inter M2M SP Entity Registration

Originator (Registree)	Receiver (Registrar)	Registration Procedure
IN-CSE	IN-CSE	CSE registration procedure. See clause 10.2.2.7

An IN-CSE is allowed to register to the IN-CSE of multiple different M2M SP domains in the oneM2M System.

Any inter-M2M SP communications in support of a request originating from one M2M SP domain shall be processed and forwarded through the IN of the originating M2M domain towards the IN of the target M2M SP domain and finally forwarded to its target CSE, if different from the target domain's IN. Hence the IN in both M2M domains shall be the exit and entry points, respectively, for all inter-M2M SP communication traffic.

6.6 M2M Service Subscription

The M2M Service Subscription defines the technical part of the contract between an M2M Subscriber (typically an M2M Application Service Provider) and an M2M Service Provider. Each M2M Service Subscription shall have a unique identifier, the M2M-Sub-ID, as specified in clause 7.1.6. An M2M Service Subscription establishes a link between one or more AEs; one or more M2M Nodes.

How to authorize the request operation based on M2M Service Subscription resource are defined in ETSI TS 118 103 [2].

An M2M Service Subscription shall be used for the following purposes:

- Serve as a basis for authorization for resource operations.
- Serve as the basis for charging.
- Identify which Nodes are part of this M2M Service Subscription.

7 M2M Entities and Object Identification

7.1 M2M Identifiers

7.1.0 Overview

This clause provides a list of identifiers required for the purpose of interworking within the oneM2M architectural model.

An M2M identifier is a sequence of characters used to refer to an entity (such as CSE or an AE), a resource (such as defined in clause 9) or an object (such as an M2M Service Provider or an M2M Node) defined in oneM2M. An M2M identifier has a consistent meaning when applied (i.e. it refers consistently to the same resource, entity or object for the duration of their lifetime, as defined in the clause 7.2) in a particular context.

7.1.1 M2M Service Provider Identifier (M2M-SP-ID)

An M2M Service Provider shall be uniquely identified by the M2M Service Provider Identifier (M2M-SP-ID). This is a static value assigned to the Service Provider.

7.1.2 Application Entity Identifier (AE-ID)

An Application Entity Identifier (AE-ID) uniquely identifies an AE resident on an M2M Node, or an AE that requests to interact with an M2M Node. An AE-ID shall identify an Application Entity for the purpose of all interactions within the M2M System.

The AE-ID is globally unique and when used internally within a specific M2M SP domain, it is sufficient to be unique within that M2M Service Provider domain. It is extended to become globally unique when used outside the M2M Service Provider boundaries. The IN-CSE shall perform this task of adding or removing identifier portions (identifying the M2M SP) according to clause 7.2.

The AE-ID, when used in the context of a specific CSE where the AE is registered, it is sufficient to be unique within the scope of that specific CSE. It is extended to become M2M Service Provider unique when used outside such specific CSE.

The Hosting CSE of the AE shall perform this task of adding or removing the identifier portions according to clause 7.2.

7.1.3 Application Identifier (App-ID)

An Application Identifier (App-ID) uniquely identifies an M2M Application in a given context. More precisely, there are two types of App-ID: registration authority defined App-ID (registered App-ID) and non-registered App-ID. The establishment of the registered App-ID is guaranteed to be globally unique; the non-registered App-ID is not guaranteed to be globally unique. The detail format is described in clause 7.2.

7.1.4 CSE Identifier (CSE-ID)

A CSE shall be identified by a unique identifier, the CSE-ID, when instantiated within an M2M Node in the M2M System.

The CSE-ID is unique in an M2M Service Provider Domain. It becomes globally unique when the M2M-SP-ID is added in front.

The CSE-ID in a resource identifier (e.g. the ***To*** parameter) indicates the Hosting CSE of the resource.

7.1.5 M2M Node Identifier (M2M-Node-ID)

An M2M Node, hosting a CSE and/or Application(s) shall be identified by a globally unique identifier, the M2M-Node-ID.

The M2M System shall allow the M2M Service Provider to set the CSE-ID and the M2M-Node-ID to the same value.

The M2M-Node-ID enables the M2M Service Provider to bind a CSE-ID to a specific M2M Node.

Examples of allocating a globally unique M2M-Node-ID include the use of Object Identity (OID) and IMEI. For details on OID, see annex H.

7.1.6 M2M Service Subscription Identifier (M2M-Sub-ID)

The M2M-Sub-ID enables the M2M Service Provider to bind application(s), M2M Nodes, CSEs and services identified by service identifiers, as well as administrative information, such as billing address, etc., to a particular M2M Service Subscription between an M2M subscriber and the M2M Service Provider. The M2M-Sub-ID is unique for every M2M subscriber.

The M2M Service Subscription Identifier has the following characteristics:

- belongs to the M2M Service Provider;
- identifies the subscription to an M2M Service Provider;
- enables communication with the M2M Service Provider;
- can differ from the M2M Underlying Network Subscription Identifier.

There can be multiple M2M Service Subscription Identifiers per M2M Underlying Network subscription.

The M2M-Sub-ID shall not be exposed over any interface.

7.1.7 M2M Request Identifier (M2M-Request-ID)

The M2M-Request-ID tracks a Request initiated by an AE over the Mca reference point, and by a CSE over the Mcc reference point, if applicable, end to end. It is also included in the Response to the Request over the Mca or Mcc reference points.

To enable an AE to track Requests and corresponding Responses over the Mca reference point, AEs shall include a distinct M2M Request Identifier per request over the Mca Reference point to the CSE for any initiated request.

The CSE shall make such M2M Request Identifier unique by prepending the AE-ID-Stem (see clause 7.2) and slash ('/') in front of it (e.g. C190XX7T/001).

If the CSE creates an M2M Request Identifier, then the CSE shall maintain a binding between the M2M Request Identifier received from the AE and the M2M Request Identifier it created in its interactions towards other peer CSEs. The CSE shall include the M2M Request Identifier received from the AE in its Response to the AE. This binding shall be maintained by the CSE until the Request message sequence is completed. Note that the Request initiated by the CSE could be the result of an application Request, or a request initiated autonomously by the CSE to fulfil a service.

In case an IN-CSE needs to send a request to a receiving CSE or ADN-AE that is not reachable over any of the underlying networks, the IN-CSE initiates the procedure for "waking up" the Node hosting the receiving CSE or ADN-AE by using procedures such as device triggering over the Mcn reference point. For Device Triggering, the triggering reference number to co-relate device triggering response is independent of the M2M Request Identifier. An IN-CSE may use the same value of an M2M-Request-Identifier in an incoming request for the triggering reference number in its interaction with the underlying network.

A CSE receiving a Request from a peer CSE shall include the received M2M Request Identifier in all additional Requests unspanned (i.e. 1:1) it has to generate (including propagation of the incoming Request) and that are associated with the incoming Request, where applicable.

If a Receiver CSE receives a request from an Originator for which another request with the same Request Identifier is already pending, the request shall be rejected. Otherwise - even if the same Request Identifier was already used by the same Originator sometime in the past, the request shall be treated as a new request.

7.1.8 M2M External Identifier (M2M-Ext-ID)

The M2M-Ext-ID is used by an M2M Service Provider (M2M SP) when services targeted to a M2M Device, are requested from the Underlying Network.

The M2M External Identifier allows the Underlying Network to identify the M2M Device (e.g. ADN, ASN, MN). To that effect, the Underlying Network maps the M2M-Ext-ID to the Underlying Network specific Identifier it allocated to the target M2M Device. In addition, the M2M SP shall maintain the association between the CSE-ID or AE-ID, the M2M-Ext-ID and the identity of the Underlying Network.

Both pre-provisioned and dynamic association between the M2M-Ext-ID with the CSE-ID or ADN AE-ID are supported.

NOTE 1: For each CSE-ID or ADN AE-ID, there is only one M2M-Ext-ID for a specific UNetwork-ID. Hence an M2M SP interworking with multiple Underlying Networks has different M2M-Ext-IDs associated with the same CSE-ID or ADN AE-ID, one per Underlying Network and selects the appropriate M2M-Ext-ID for any service request it initiates towards an Underlying Network.

NOTE 2: The mapping by the Underlying Network of the M2M-Ext-ID to the M2M Device is Underlying Network specific.

NOTE 3: The Underlying Network provider and the M2M Service Provider collaborate for the assignment of an M2M-Ext-ID to each M2M Device. At the same time, the Underlying Network provider maintains association of the M2M-Ext-ID with the Underlying Network specific Identifier allocated to the M2M Device that hosts such CSE.

For pre-provisioned M2M-Ext-IDs, the M2M-Ext-ID along with the associated CSE-ID or ADN AE-ID shall be made available at the Infrastructure Node. The CSE or AE at M2M Device does not need to have knowledge of the M2M-Ext-ID assigned to it.

For dynamic M2M-Ext-IDs, the M2M-Ext-ID specific to the Underlying Network shall be made available at the M2M Device in the Field Domain. Such M2M-Ext-ID shall be conveyed to the IN-CSE during Registration.

The M2M-Ext-ID is to be used by the underlying network to identify an AE for verification when an AE retrieves the location information of a remote M2M device from a network-based location server of the underlying network (e.g. the 3GPP location server GMLC).

NOTE 4: The mapping by the Underlying Network of the M2M-Ext-ID to the AE is Underlying Network specific. And how the underlying network performs the privacy control is out of the scope.

NOTE 5: When the M2M-Ext-ID is targeted to an AE, the format is defined by the Underlying Network.

7.1.9 Underlying Network Identifier (UNetwork-ID)

The UNetwork-ID is used for identifying an Underlying Network. UNetwork-ID is a static value and unique within a M2M Service Provider domain.

One or more Underlying Networks may be available at an M2M Node offering different sets of capabilities, availability schedules etc. Based on the "policy" information at the Node and the capabilities offered by the available Underlying Networks, appropriate Underlying Network can be chosen by using UNetwork-ID. For example, based on "policy", scheduling of traffic triggered by a certain event category in certain time periods may be allowed over Underlying Network "WLAN" but may not be allowed over Underlying Network "2G Cellular".

7.1.10 Trigger Recipient Identifier (Trigger-Recipient-ID)

The Trigger-Recipient-ID is used when device triggering services are requested from the Underlying Network, to identify an instance of an ASN/MN-CSE or ADN-AE on an execution environment, to which the trigger is routed.

EXAMPLE: When 3GPP device triggering is used, the Trigger-Recipient-ID maps to the Application-Port-Identifier (ETSI TS 123 682 [i.14]).

NOTE 1: For pre-provisioned M2M-Ext-IDs, Trigger-Recipient-ID is provisioned at the Infrastructure Node along with the M2M-Ext-ID and the associated CSE-ID or ADN AE-ID.

NOTE 2: For dynamic M2M-Ext-IDs, Trigger-Recipient-ID specific to the Underlying Network is provisioned at each M2M Device in the Field Domain. Such Trigger-Recipient-ID is conveyed to the IN-CSE during Registration.

7.1.11 Void

7.1.12 Void

7.1.13 M2M Service Profile Identifier (M2M-Service-Profile-ID)

An M2M Service Profile Identifier defines applicable rules governing the AEs registering with M2M Nodes and the AEs residing on these nodes. Every M2M Service Profile is allocated an identifier so it can be retrieved for verification purposes.

The M2M-Service-Profile-ID enables the M2M Service Provider to bind AE(s), applicable rules to these AEs, as well as M2M Service Roles to M2M nodes.

An M2M-Service-Profile-ID shall be allocated to every M2M Node.

The M2M Service Profile Identifier has the following characteristics:

- belongs to the M2M Service Provider;
- identifies applicable rules governing AEs registering with an M2M node.

7.1.14 Role Identifier (Role-ID)

A Role identifier (Role-ID) is an identifier that a request originator may use in order to allow the CSE to enforce access control for resources. An originator may only use a Role-ID that is allowed by his service subscription profile.

7.1.15 Token Identifier (Token-ID)

A Token identifier (Token-ID) is the identifier for a Token. The Token-ID is assigned by the issuer of the Token.

Token-IDs shall meet the following criteria:

- A Token-ID shall identify the issuer of the Token.
- The Token-ID's uniqueness shall be global, with the proviso that a Token-ID value assigned to a Token may be assigned to another Token once the former Token has expired.

7.1.16 Local Token Identifier (Local-Token-ID)

A local token identifier (Local-Token-ID) is an identifier for a Token which can be assigned by a Hosting CSE making an accessing decision when it receives a request from an Originator which includes that Token or Token-ID in the request parameters (see clause 11.5.3).

In these scenarios, the request from the Originator included either the Token or the Token's Token-ID assigned by the Token's Issuer (see clause 7.1.15). In the latter case the Hosting CSE retrieves the Token using the Token-ID. The Hosting CSE assigns a Local-Token-ID to the Token. In the corresponding response message, the Hosting CSE provides the Originator with the mapping from the Local-Token-ID to the corresponding Token-ID. In subsequent requests to the Hosting CSE, the Originator can provide the Local-Token-ID in the place of the corresponding Token-ID or Token. - The intention is that the Local-Token-ID would be significantly shorter than the Token or issuer-assigned Token-ID in order to reduce the size of the subsequent request messages. For more details regarding the use of Local-Token-ID, see clause 11.5.3.

Local-Token-IDs shall meet the following criteria:

- The Local-Token-ID shall be assigned by the Hosting CSE making access decisions using the corresponding Token.
- The Local-Token-ID's uniqueness shall be local to the Hosting CSE, with the proviso that a Local-Token-ID value assigned to a Token may be assigned to another Token once the former Token has expired.

7.2 M2M-SP-ID, CSE-ID, App-ID and AE-ID and resource Identifier formats

As a general rule, the identifiers of AEs, CSEs and resources are globally unique. In order to optimize their use, the identifiers shall be shortened when their scope can be derived from their context of use by the CSEs and the AEs. Such shortened identifiers are defined as 'relative' formats of the identifiers.

TheM2M system shall use the identifiers M2M-SP-ID, CSE-ID, App-ID and AE-ID and resource identifiers according to the formats and the rules specified in table 7.2-1.

Table 7.2-1: Identifier formats and rules of use

Identifier Name	Absolute & Format-Designator or Relative & Format-Designator & Context	Format	Rule of use

M2M-SP-ID	Absolute
	
	M2M-SP-ID
		The M2M-SP-ID shall conform to the FQDN format defined in the IETF RFC 1035 [i.7] prefixed by '//'
		
		The format then has the structure of
		
		//{FQDN}
		
		Where {FQDN} is a placeholder for the Fully Qualified Domain Name of the M2M Service Provider Domain
		
		EXAMPLES:
		- //www.m2mprovider.com
		- //globalm2m.org
		
		The following two M2M-SP-IDs could be used to separate two service segments:
		
		//automotive.m2m.telematics-service-company.com
		
		//building-management.m2m.telematics-service-company.com
			Whenever The M2M-SP-ID is used, only an Absolute format of the M2M-SP-ID defined herein applies.

CSE-ID	Relative SP-relative-CSE-ID Context: M2MService Provider Domain of the CSE	The SP-relative-CSE-ID begins with a slash character '/' and is followed by a sequence of characters that may include any of the unreserved characters defined in the clause 2.3 of the IETF RFC 3986 [18]. The SP-relative-CSE-ID is unique within the context of the M2M-SP Domain hosting the CSE. The M2M-SP is assigning the SP-Relative-CSE-ID and is responsible for guaranteeing that the SP-Relative-CSE-ID is unique in the context of the hosting M2M-SP Domain. EXAMPLES: • /123A38ZZY • /CSE090112 • /3ace4fd3	On the Mca and Mcc reference points: to refer to CSEs that are in the same M2M Service Provider Domain of the Receiver CSE.

Identifier Name	Absolute & Format-Designator or Relative & Format-Designator & Context	Format	Rule of use
	Absolute Absolute-CSE-ID	Concatenation according to the format {M2M-SP-ID}{SP-relative-CSE-ID} where {M2M-SP-ID} and {SP-relative-CSE-ID} are placeholders for the M2M-SP-ID and the SP-relative-CSE-ID format of the CSE-ID, respectively. The Absolute-CSE-ID complies with what is specified in clause 3 of IETF RFC 3986 [18] under "hier-part". EXAMPLES: • //www.m2mprovider.com/C3219 • //m2m.thingscompany.com/ab3f124a	On Mca, Mcc and Mcc' reference points: to refer to CSEs that are in different M2M Service Provider Domains.

Identifier Name	Absolute & Format-Designator or Relative & Format-Designator & Context	Format	Rule of use
AE-ID	Relative AE-ID-Stem Context: • Registrar CSE of the AE or • M2MService Provider Domain of the AE	The AE-ID-Stem is a sequence of characters that may include any of the unreserved characters defined in the clause 2.3 of the IETF RFC 3986 [18]. The first character of the AE-ID-Stem has a specific meaning and its value shall be as follows: 1. Fist character of AE-ID-Stem is 'C' The AE-ID-Stem is assigned by the Registrar CSE of the AE. In this case, the AE-ID-Stem shall be unique within the context of the Registrar CSE of the AE. The Hosting CSE is responsible for guaranteeing that the AE-ID-Stem is unique in the context of the Hosting CSE. EXAMPLES: • C190XX7T • Ca3e3f3ab 2. Fist character of AE-ID-Stem is 'S': The AE-ID-Stem is assigned by the M2M-SP. In this case, the AE-ID-Stem shall be unique within the context of the M2M-SP Domain. The M2M-SP is responsible for guaranteeing that the AE-ID-Stem is unique in the context of the M2M-SP Domain. EXAMPLES: • S190XX7T • Sa3e3f3ab Use of other values for the first character of AE-ID-Stem is reserved. Which of the cases above shall apply will be determined during the AE registration procedure. The details of the process how an AE-ID-Stem unique within the M2M-SP Domain is assigned by the M2M-SP are described in the AE registration procedure description.	On the Mca reference point: to refer to AEs that registered to the Receiver CSE.

Identifier Name	Absolute & Format-Designator or Relative & Format-Designator & Context	Format	Rule of use
	Relative SP-relative-AE-ID Context: M2M Service Provider Domain of the AE	1. In the case the AE-ID-Stem starts with the letter 'C', the SP-relative-AE-ID is a concatenation according to the format {SP-relative-CSE-ID}/{AE-ID-Stem} where {SP-relative-CSE-ID} and {AE-ID-Stem} are placeholders for the SP-relative-CSE-ID of the Registrar CSE of the AE and the AE-ID-Stem format of the AE-ID, respectively. EXAMPLES: • /CSE090112/C190XX7T • /3ace4fd3/Ca3e3f3ab 2. In the case the AE-ID-Stem starts with the letter 'S', the AE-ID-Stem is unique within the M2M-SP Domain. In that case the SP-relative-AE-ID is a concatenation according to the format /{AE-ID-Stem} where {AE-ID-Stem} is a placeholder for the AE-ID-Stem format of the AE-ID. EXAMPLES: • /S190XX7T • /Sa3e3f3ab The SP-relative-AE-ID begins with a slash character '/', and it complies with what is specified in clause 4.2 of IETF RFC 3986 [18] under "absolute-path reference".	On the Mca and Mcc reference points: to refer to AEs in the same M2M Service Provider Domain.

Identifier Name	Absolute & Format-Designator or Relative & Format-Designator & Context	Format	Rule of use
	Absolute Absolute-AE-ID	The Absolute-AE-ID format of the AE-ID is a concatenation according to the format: {M2M-SP-ID}{SP-relative-AE-ID} where {M2M-SP-ID} and {SP-relative-AE-ID} are placeholders for the M2M-SP-ID and the SP-relative-AE-ID format of the AE-ID, respectively. The absolute AE-ID complies with what is specified in clause 3 of IETF RFC 3986 [18] under "hier-part". EXAMPLES: • //m2m.prov.com/CSE3219/C9886 • //m2m.things.com/ab3f124a/Ca2efb3f4 • //m2m.things.com/S98821	On the Mca, Mcc and Mcc' reference points: to refer to AEs that are in different M2M Service Provider Domains.
Resource identifier	Relative Unstructured-CSE-relative-Resource-ID Context: CSE hosting the Resource	An Unstructured-CSE-relative-Resource-ID is a sequence of characters that may include any of the unreserved characters defined in the clause 2.3 of the IETF RFC 3986 [18]. An Unstructured-CSE-relative-Resource-ID is unique in the context of the CSE hosting the resource. The Hosting CSE of the resource is responsible for guaranteeing that Unstructured-CSE-relative Resource-IDs are unique in the context of the Hosting CSE. EXAMPLES: - container123 - a1b2c3d4b0b00f0fa66a123456789abc - xxyz1234	On the Mca and Mcc reference point: to refer to resources that are hosted by the CSE which is the Registrar CSE of the Originator.

Identifier Name	Absolute & Format-Designator or Relative & Format-Designator & Context	Format	Rule of use
	Relative Structured-CSE-relative-Resource-ID Context: CSE hosting the resource	A Structured-CSE-relative-Resource-ID is a sequence of characters that may include any of the unreserved characters defined in the clause 2.3 of the IETF RFC 3986 [18], as well as the slash character. It shall not start with the slash character. A Structured-CSE-relative Resource-ID is unique in the context of the CSE hosting the resource. The structure represents a chain of parent-child-relationships using resource IDs or resource names of parents and resource names of their children for segments that are separated by the '/' character. The first segment is one of the following: A. the resource name of <CSEBase> resource; B. the character "-" (dash) as a shortcut for the resource name of <CSEBase> resource; C. the Unstructured-CSE-relative-Resource-ID of a parent resource on the Hosting CSE. When this is used, the second segment shall be the resourceName of a virtual resource. NOTE: In case of C above, for convenience it is called a hybrid resource identifier. The Hosting CSE of the resource is responsible for guaranteeing that resource names - which are used to construct Structured-CSE-relative-Resource-ID formats - are unique in the context of a set of sibling resources sharing the same parent resource on the Hosting CSE. EXAMPLES: - bigCSE025/mainStreet/house5432/livingRoom/temperature This example is the Structured-CSE-relative-Resource-ID of a <*container*> resource, where "bigCSE025" is assumed to be the name of the <*CSEBase*> resource, followed by four "/"-separated segments with names of <*container*> resources that are nested child resources thereof.	On the Mca and Mcc reference point: To refer to resources that are hosted by the CSE receiving a request targeting a resource.

Identifier Name	Absolute & Format-Designator or Relative & Format-Designator & Context	Format	Rule of use
		- CSE-Building-A3/HVAC-AE/WaterTemp/sample0098 This example is the Structured-CSE-relative-Resource-ID of a <*contentInstance*> resource, where "CSE-Building-A3" is assumed to be the name of the <*CSEBase*> resource, followed by "/" plus the name "HVAC-AE" of an <*AE*> child resource, followed by "/" plus the name "WaterTemp" of a <*container*> child resources, followed by "/" plus the name "sample0098" of a child <*contentInstance*> resource. - ./HVAC-AE/WaterTemp/sample0098 This example is the Structured-CSE-relative-Resource-ID of a <*contentInstance*> resource, where the dash symbol "-" is used as a shortcut for the name of the <*CSEBase*> resource, followed by "/" plus the name "HVAC-AE" of an <*AE*> child resource, followed by "/" plus the name "WaterTemp" of a <*container*> child resource, followed by "/" plus the name "sample0098" of a child <*contentInstance*> resource. - - 000AFE030003/sample0098 This example is the Structured-CSE-relative-Resource-ID of a <*contentInstance*> resource, where "000AFE030003" is assumed to be the Unstructured-CSE-relative-Resource-ID of a <*container*> resource, followed by "/" plus the name "sample0098" of a child <*contentInstance*> resource.	

Identifier Name	Absolute & Format-Designator or Relative & Format-Designator & Context	Format	Rule of use
	Relative SP-relative Resource-ID Context: M2MService Provider Domain hosting the resource	Concatenation according to the format: {SP-relative-CSE-ID}/{Unstructured-CSE-relative Resource ID} {SP-relative-CSE-ID}/{Structured-CSE-relative Resource ID} where {SP-relative-CSE-ID}, {Unstructured-CSE-relative Resource ID}, {Structured-CSE-relative Resource ID} are placeholders for the SP-relative-CSE-ID format of the CSE-ID and the Unstructured-CSE-relative-Resource-ID or a Structured-CSE-relative-Resource-ID format of the Resource ID, respectively. The SP-relative-Resource-ID begins with a slash character, and it complies with what is specified in clause 4.2 of IETF RFC 3986 [18] under "absolute-path reference". The SP-relative Resource ID is unique in the context of the Service Provider. EXAMPLES: • /CSE987776/a234361 This example is the SP-relative Resource-ID of a resource - not assuming any specific resource type - where the resource is hosted on a CSE with the SP-relative-CSE-ID "/CSE987776" and where the Unstructured-CSE-relative-Resource-ID is "a234361". • /CSE00030F003A/CSE-Building-A3/HVAC-AE/WaterTemp/sample0098 This example is the SP-relative Resource-ID of a <*contentInstance*> resource, where the targeted resource is hosted on a CSE with the SP-relative-CSE-ID "/CSE00030F003A" and where the CSE-ID is followed by "/" plus the name "CSE-Building-A3" of the <*CSEBase*> resource, followed by "/" plus the name "HVAC-AE" of an <*AE*> child resource, followed by "/" plus the name "WaterTemp" of a <*container*> child resource, followed by "/" plus the name "sample0098" of the targeted child <*contentInstance*> resource.	On the Mca and Mcc reference points: to refer to resources that are hosted by the CSE in the same M2M Service Provider Domain as the Originator.

Identifier Name	Absolute & Format-Designator or Relative & Format-Designator & Context	Format	Rule of use
		• /CSE00030F003A/./HVAC-AE/WaterTemp/sample0098 This example is the SP-relative Resource-ID of a <*contentInstance*> resource, where the targeted resource is hosted on a CSE with the SP-relative-CSE-ID "/CSE00030F003A" and where the CSE-ID is followed by "/" plus the dash symbol "-" as a shortcut for the name of the <*CSEBase*> resource, followed by "/" plus the name "HVAC-AE" of an <*AE*> child resource, followed by "/" plus the name "WaterTemp" of a <*container*> child resource, followed by "/" plus the name "sample0098" of the targeted child <*contentInstance*> resource. • /CSE00030F003A/000AFE030003/sample0098 This example is the SP-relative Resource-ID of a <*contentInstance*> resource, where the targeted resource is hosted on a CSE with the SP-relative-CSE-ID "/CSE00030F003A" and where the CSE-ID is followed by "/" plus the Unstructured-CSE-relative-Resource-ID "000AFE030003" of a <*container*> resource, followed by "/" plus the name "sample0098" of the targeted child <*contentInstance*> resource.	

Identifier Name	Absolute & Format-Designator or Relative & Format-Designator & Context	Format	Rule of use
	Absolute Absolute Resource ID	Concatenation according to the format: {M2M-SP-ID}{SP-relative Resource ID} where {M2M-SP-ID} and {SP-relative Resource ID} are placeholders for the M2M-SP-ID and the SP-relative Resource ID format of the Resource ID, respectively. The Absolute-CSE-ID complies with what is specified in clause 3 of IETF RFC 3986 [18] under "hier-part". EXAMPLES: - //www.m2mprovider.com / CSE987776/a234361 This example is the Absolute Resource-ID of a resource - not assuming any specific resource type - where the resource is hosted within the domain of the M2M-Service Provider with the M2M-SP-ID "//www.m2mprovider.com" on a CSE with SP-relative-CSE-ID "/CSE987776" and where the Unstructured-CSE-relative-Resource-ID of the targeted resource is "a234361". //www.m2mprovider.com /CSE00030F003A/CSE-Building-A3/HVAC-AE/WaterTemp/sample0098 This example is the Absolute Resource-ID of a <*contentInstance*> resource, where the targeted resource is hosted within the domain of the M2M-Service Provider with the M2M-SP-ID "//www.m2mprovider.com" on a CSE with the SP-relative-CSE-ID "/CSE00030F003A" and where the CSE-ID is followed by "/" plus the name "CSE-Building-A3" of the <*CSEBase*> resource, followed by "/" plus the name "HVAC-AE" of an <*AE*> child resource, followed by "/" plus the name "WaterTemp" of a <*container*> child resource, followed by "/" plus the name "sample0098" of the targeted child <*contentInstance*> resource.	On Mca, Mcc and Mcc' reference **points**: to refer to resources that are hosted by the CSE in a different M2M Service Provider Domain than the Originator's.
	App-ID	App-ID is either registered with the M2M App-ID Registration Authority or non-registered. Registered App-IDs shall be in the format: R{authority-ID}.{reverseDNS}.{applicationName}	AE Registration Procedure described in clause 10.2.2.2. The first character of the App-ID shall be a capital letter of 'R' for registered and 'N' for non-registered.

Identifier Name	Absolute & Format-Designator or Relative & Format-Designator & Context	Format	Rule of use
		The {reverseDNS} part shall be a string value following 'reverse DNS notation', which is constructed in the reverse order of domain name components (see IETF RFC 1035 [i.7]) Non-registered App-IDs shall be in the format: N{non-registered-App-ID} EXAMPLES: • Ra01.com.company.smartcity - Nk836-t071-fc022	
APP-ID		•	

The format (i.e. CSE-relative, SP-relative or absolute) of resource identifier (e.g. the ***To*** parameter, *accessControlPolicyIDs* attribute) shall be correctly set by the Originator in an initial request, while the format of AE-ID or CSE-ID in the ***From*** parameter shall be set in a shortest format by the Originator in the initial request and it shall be converted in another format by the Registrar CSE or IN-CSE as the following.

When an AE is the Originator, the ***From*** parameter shall be in AE-ID-Stem. When the Registrar CSE receives the request, it shall convert the format into SP-relative AE-ID in case the stem is CSE-relative and the ***To*** parameter refers to a resource hosted by a different CSE.

When an CSE is the Originator, the ***From*** parameter shall be in SP-relative CSE-ID.

The IN-CSE shall convert the format of the ***From*** parameter in a request that is received from SP-relative to absolute if the ***To*** parameter refers to a resource is hosted by a CSE in a different M2M Service Provider Domain.

7.3 M2M Identifiers lifecycle and characteristics

Table 7.3-1: M2M Identifiers lifecycle and characteristics

Identifier	Assigned by	Assigned to	Assigned during	Lifetime	Uniqueness	Used during	Remarks
M2M Service Provider Identifier	Out of scope	AE, CSE	Out of scope	Out of scope	Global	Provisioning	
Application Entity Identifier	AE or Registrar CSE	AE	AE start-up	Application Entity Registration	Global	- Application Entity Registration - Security Context Establishment - All other operations initiated by the AE	Security requirements apply for Security Context Establishment.
Application Identifier	Out of scope	Out of scope	Pre-provisioned	Out of scope	Specific to M2M service deployment	- Application Entity registration	
CSE Identifier	M2M SP	CSE	Security Provisioning	Life of the CSE	Global	- Information flows (clause 10) - Security Context Establishment	Security requirements apply for Security Context Establishment.
M2M Node Identifier	Out of Scope	All M2M Nodes	Pre-provisioned	Life of the M2M Node	Global	- Device Management	Needs to be Read Only
M2M Subscription Identifier	M2M SP, Out of Scope	Application Entities, and one or more CSEs belonging to the same M2M subscriber	At service signup	Life of the M2M Service Subscription with the M2M Service Provider	Global	- Charging and Information Recorded - Role based access control - Authentication	Multiple CSEs can be allocated the same M2M Subscription Identifier.
M2M Service Profile Identifier	M2M SP	Every M2M Node	At service signup	Life of M2M Service Subscriptions with the M2M Service Provider	Global for roaming cases otherwise local	Information Flows (clause 10)	The ID has to be pre-provisioned after signup, but may need to be updated during the subscription lifetime due to changes in the subscribed services.
M2M-Request-ID	**Mcc:** CSE **Mca:** Application Entity	A request initiated by an AE or CSE	**Mcc:** When a request is initiated by a CSE, or handling of a request received by a CSE. **Mca:** When a request is initiated by an AE	Equal to the lifetime of the Request and its corresponding Response	**Mcc:** Global **Mca:** Local or global	Requests and corresponding responses	

Identifier	Assigned by	Assigned to	Assigned during	Lifetime	Uniqueness	Used during	Remarks
External Identifier	Jointly between the Underlying Network provider and M2M SP.	M2M Node belonging to a CSE or ADN-AE that wants to utilize services of the Underlying Network.	Administrative Agreement.	Life of the CSE or ADN-AE.	Local or global, decided by the specific Underlying Network provider	Requests initiated by a CSE over the Mcn reference point, where applicable. Querying the location information of a remote node from the underlying network.	**Pre-Provisioned Mode:** Made available at the Infrastructure Node. **Dynamic Mode:** Made available at M2M device. Conveyed to IN-CSE during CSE or AE Registration.
Underlying Network Identifier	M2M SP	Underlying Networks	Pre-provisioned	Life of the agreement by the M2M SP with the Underlying Network	Local to M2M SP domain	UL Network selection	
Trigger Recipient Identifier	Execution Environment	ASN/MN-CSE or ADN-AE	ASN/MN-CSE or ADN-AE start-up or wake-up	Life of the CSE or ADN-AE	Execution Environment-wide	Device Triggering procedures, where applicable	**Pre-Provisioned Mode:** Made available at Infrastructure Node along with M2M-Ext-ID. **Dynamic Mode:** Made available at M2M device. Conveyed to IN-CSE during CSE or AE Registration along with M2M-Ext-ID.
M2M Service Identifier	M2M Service Provider, Out of Scope	A service defined by the M2M Service Provider which consists of a set of functions defined by the present document.	Out of Scope	Out of Scope	Local to the M2M Service Provider	For M2M Service Subscription	
Role-ID	M2M Service Provider	Application Entities, and one or more CSEs belonging to the same M2M subscriber	Out of scope	Out of scope	Local to M2M SP domain	Access Control Policy	
Token-ID	Token Issuer	Token	Token Assignment	Specified by Token	Global	Dynamic Authorization	
Local-Token-ID	A Hosting CSE making access decisions with the corresponding token	Token	After Hosting CSE has been provided with Token	Specified by Token	Local to the Hosting CSE	Indirect Dynamic Authorization	See clause 11.5.3.

8 Description and Flows of Reference Points

8.1 General Communication Flow Scheme on Mca and Mcc Reference Points

8.1.0 Overview

Procedures involving CSEs and AEs are driven by the exchange of messages across reference points according to the message flows described in this clause.

Depending on the message operation, procedures may manipulate information in a standardized resource structure as described in clause 9. Access and manipulation of the resources is subject to their associated privileges.

8.1.1 Description

Figure 8.1.1-1 shows the general flow that governs the information exchange within a procedure, which is based on the use of Request and Response messages. The message applies to communications such as:

- between an AE and a CSE (Mca reference point); and
- among CSEs (Mcc reference point).

Such communications can be initiated either by the AEs or by the CSEs depending upon the operation in the Request message.

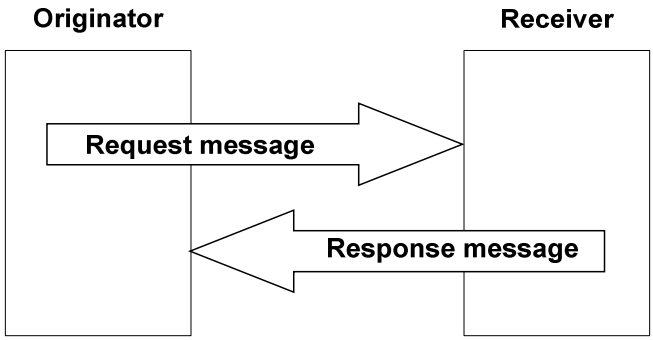


Figure 8.1.1-1: General Flow

8.1.2 Request

Requests over the Mca and Mcc reference points, from an Originator to a Receiver, shall contain mandatory and may contain optional parameters. Certain parameters may be mandatory or optional depending upon the Requested operation. In this clause, the mandatory parameters are detailed first, followed by those that are operation dependent, and then by those that are optional:

- ***To*****:** Address of the target resource or target attribute for the operation. The ***To*** parameter shall conform to clause 9.3.1.

NOTE 1: ***To*** parameter can be known either by pre-provisioning (clause 11.2) or by discovery (clause 10.2.6 for discovery). Discovery of <*CSEBase*> resource is not supported in this release of the document. It is assumed knowledge of <*CSEBase*> resource is by pre-provisioning only.

NOTE 2: The term target resource refers to the resource which is addressed for the specific operation. For example, the ***To*** parameter of a Create operation for a resource <*example*> would be "/m2m.provider.com/exampleBase". The ***To*** parameter for the Retrieve operation of the same resource <*example*> is "/m2m.provider.com/exampleBase/example".

NOTE 3: For Retrieve operation (clause 10.1.3), the ***To*** parameter can be the URI of an attribute to be retrieved.

- ***From*****:** Identifier representing the Originator.

  The ***From*** parameter is used by the Receiver to check the Originator identity for access privilege verification.

- ***Operation*****:** operation to be executed: Create (C), Retrieve (R), Update (U), Delete (D), Notify (N).

  The ***Operation*** parameter shall indicate the operation to be executed at the Receiver:

  - **Create (C):** ***To*** is the address of the target resource where the new resource (parent resource).
  - **Retrieve (R):** an existing ***To*** addressable resource is read and provided back to the Originator.
  - **Update (U):** the content of an existing ***To*** addressable resource is replaced with the new content as in ***Content*** parameter. If some attributes in the ***Content*** parameter do not exist at the target resource, such attributes are created with the assigned values. If some attributes in the ***Content*** parameter are set to NULL, such attributes are deleted from the addressed resource.
  - **Delete (D):** an existing ***To*** addressable resource and all its sub-resources are deleted from the Resource storage.
  - **Notify (N):** information to be sent to the Receiver, processing on the Receiver is not indicated by the Originator.

- ***Request Identifier*****:** request Identifier (see clause 7.1.7).

  Example usage of request identifier includes enabling the correlation between a Request and one of the many received Responses.

Operation dependent Parameters:

- ***Content*****:** resource content to be transferred.

  The ***Content*** parameter shall be present in Request for the following operations:

  - **Create (C):** ***Content*** is the content of the new resource with the resource type ***ResourceType.***
  - **Update (U):** ***Content*** is the content to be replaced in an existing resource. For attributes to be updated at the resource, ***Content*** includes the names of such attributes with their new values. For attributes to be created at the resource, ***Content*** includes names of such attributes with their associated values. For attributes to be deleted at the resource, ***Content*** includes the names of such attributes with their value set to NULL.
  - **Notify (N):** ***Content*** is the notification information.

  The ***Content*** parameter may be present in Request for the following operations:

  - **Retrieve (R):** ***Content*** is the list of attribute names from the resource that needs to be retrieved. The values associated with the attribute names shall be returned.

- ***Resource Type:*** type of resource.

  The ***ResourceType*** parameter shall be present in Request for the following operations:

  - **Create (C):** ***Resource Type*** is the type of the resource to be created.

Optional Parameters:

- ***Role IDs:*** optional, required when role based access control is applied. A list of Role-IDs that are allowed by the service subscription shall be provided otherwise the request is considered not valid.

  The ***Role IDs*** parameter shall be used by the Receiver to check the Access Control privileges of the Originator.

- ***Originating Timestamp*:** optional originating timestamp of when the message was built.

  Example usage of the originating timestamp includes: to measure and enable operation (e.g. message logging, correlation, message prioritization/scheduling, accept performance requests, charging, etc.) and to measure performance (distribution and processing latency, closed loop latency, SLAs, analytics, etc.)

- ***Request Expiration Timestamp*:** optional request message expiration timestamp. The Receiver CSE should handle the request before the time expires. If a Receiver CSE receives a request with ***Request Expiration Timestamp*** with the value indicating a time in the past, then the request shall be rejected.

  Example usage of the request expiration timestamp is to indicate when request messages (including delay-tolerant) should expire and to inform message scheduling/prioritization. When a request with set expiration timestamp demands an operation on a Hosting CSE different than the current Receiver CSE, then the current CSE shall keep trying to deliver the Request to the Hosting CSE until the request expiration timestamp time, in line with provisioned policies.

- ***Result Expiration Timestamp*:** optional result message expiration timestamp. The Receiver CSE should return the result of the request before the time expires.

  Example usage of the result expiration timestamp: An Originator indicates when result messages (including delay-tolerant) should expire and informs message scheduling/prioritization. It can be used to set the maximum allowed total request/result message sequence round trip deadline.

- ***Response Type*:** optional response message type: Indicates what type of response shall be sent to the issued request and when the response shall be sent to the Originator:

  - **nonBlockingRequestSynch:** In case the request is accepted by the Receiver CSE, the Receiver CSE responds, after acceptance, with an Acknowledgement confirming that the Receiver CSE will further process the request. The Receiver CSE includes in the response to an accepted request a reference that can be used to access the status of the request and the result of the requested operation at a later time. Processing of Non-Blocking Requests is defined in clause 8.2.2 and in particular for the synchronous case in clause 8.2.2.2.

  - **nonBlockingRequestAsynch {optional list of notification targets}:** In case the request is accepted by the Receiver CSE, the Receiver CSE shall respond, after acceptance, with an Acknowledgement confirming that the Receiver CSE will further process the request. The result of the requested operation needs to be sent as notification(s) to the notification target(s) provided optionally within this parameter as a list of entities or to the Originator when no notification target list is provided. When an empty notification target list is provided by the Originator, no notification with the result of the requested operation shall be sent at all. Processing of Non-Blocking Requests is defined in clause 8.2.2 and in particular for the asynchronous case in clause 8.2.2.3.

  - **blockingRequest:** In case the request is accepted by the Receiver CSE, the Receiver CSE responds with the result of the requested operation after completion of the requested operation. Processing of Blocking Requests is defined in clause 8.2.1. This is the default behaviour when the *Response Type* parameter is not given the request.

  - **flexBlocking {optional list of notification targets}:** When ***Response Type*** in the request received by the Receiver CSE is set to flexBlocking, it means that the Originator of the request has the capability to accept the following types of responses: nonBlockingRequestSynch, nonBlockingRequestAsynch and blockingRequest.

    The Receiver CSE shall make the decision to respond using blocking or non-blocking based on its own local context (memory, processing capability, etc.) if not defined in the resource handling procedure.

    If the Receiver CSE choose to respond using non-blocking mode or blocking mode, based on the presence of notification targets in the request:

    - If the notification targets are provided in the request and the Receiver CSE is responding, the Receiver CSE shall choose and respond with nonBlockingRequestAsynch, nonBlockingRequestSynch or blockingRequest mode.

    - If notification targets are not provided, the Receiver CSE shall choose and respond with nonBlockingRequestSynch or blockingRequest mode.

- **No Response:** In case the request is accepted by the Receiver CSE or AE, the Receiver CSE or AE does not respond with the result of the requested operation after completion of the requested operation. Note, in this case the Result Content parameter should not be included in the request.

  Example usage of the response type set to *nonBlockingRequestSynch*: An Originator that is optimized to minimize communication time and energy consumption wants to express a Request to the receiver CSE and get an acknowledgement on whether the Request got accepted. After that the Originator may switch into a less power consuming mode and retrieve a Result of the requested Operation at a later time.

  Further example usage of response type set to *nonBlockingRequestSynch:* When the result content is extremely large, or when the result consists of multiple content parts from a target group which are to be aggregated asynchronously over time.

- ***Result Content*:** optional result content: Indicates what are the expected components of the result of the requested operation. This shall be indicated in the ***Result Content*** parameter. Settings of ***Result Content*** depends on the requested operation specified in ***Operation***. This parameter is not applicable when ***Response Type*** has a value of *No Response*. Possible values of ***Result Content*** are:

  - **attributes:** A representation of the targeted resource including all its attributes shall be returned as content, without the address(es) of the child resource(s) or their descendants. For example, if the request is to retrieve a <*container*> resource, the address(es) of the <*contentInstance*> child-resource(s) is not provided. This setting shall be only valid for Create, Retrieve, Update, or Delete operation. If the Originator does not set ***Result Content*** parameter in a Create, Retrieve or Update request message, this setting shall be the default value when the Receiver processes the request message.

  - **modified-attributes:** This setting shall be only valid for a Create or Update operation. A representation of the targeted resource including only attributes that were added, modified or deleted that were not included in the request ***Content*** parameter as well as any attributes which were set to values different from the values specified in the request ***Content*** parameter shall be returned as content, without the address(es) of the child resource(s) or their descendants.

  - **hierarchical-address:** Representation of the address of the created resource. This setting shall only be valid for a Create operation. The address shall be in hierarchical address scheme.

  - **hierarchical-address+attributes:** Representation of the address in hierarchical address scheme and the attributes of the created resource. This setting shall only be valid for a Create operation.

  - **attributes+child-resources:** Representation of the requested resource, along with a nested representation of all of its child resource(s), and their descendants, in line with any provided filter criteria as given in the ***Filter Criteria*** parameter shall be returned as content. If there is no filter criteria parameter in the request message, then all children/descendants are returned along with their attributes. For example, if the request is to retrieve a <*container*> resource that only has <*contentInstance*> children, the attributes of that <*container*> resource and a representation of all of its <*contentInstance*> child-resource(s), including their attributes, are provided.

    The originator may request to limit the maximum number of allowed nesting levels. The originator may also include an offset that indicates the starting point of the direct child resource. The offset shall start at 1. The hosting CSE shall return all direct child resources and their descendants, or up to the maximum nesting level specified in a request subject to maximum size limit that may be imposed by the hosting CSE. The offset, maximum number/size and maximum level shall be specified in ***Filter Criteria*** as *offset, limit,* and *level* condition, respectively, by the Originator.

    The hosting CSE shall list parent resources before their children. This means that the originator of the request will not receive a descendant resource without having received its parents. The hosting CSE shall also ensure that proper nesting representation of all the children is incorporated in its listing for parents and children.

    Nested processing is applicable at every level in the resource tree. If a direct child resource and all its descendants cannot be included in the returned content due to size limitations imposed by the hosting CSE then the direct child resource shall not be included in the response.

    An indication shall be included in the response signalling if the returned content is partial. If the indication is for partial content, the response shall include an offset for the direct child resource where processing can restart for the remaining direct child resources.

This shall be only valid for a Retrieve/Delete operation.

- **child-resources:** A nested representation of the resource's child resource(s) their descendants and their attributes shall be returned as content. The resources that are returned are subject to any filter criteria that are given in the ***Filter Criteria*** parameter (if there are no filter criteria then all children and their descendants are returned). The attributes of the parent resource are not returned, but all the attributes of the children are returned. For example, if the request is to retrieve a <*container*> resource that only has <*contentInstance*> children, only a representation of all of its <*contentInstance*> child-resource(s) is provided.

  The offset, maximum number/size and maximum level shall be specified in ***Filter Criteria*** as *offset*, *limit*, and *level* condition, respectively, by the Originator. Processing of direct child resources, size limitations, maximum nesting level, and offset for the starting of direct child resource processing of **the attributes+child-resources** option shall apply to this option as well.

  This shall be only valid for a Retrieve/Delete operation.

- **attributes+child-resource-references:** Representation of the requested resource, along with the address(es) of the child resource(s), and their descendants shall be returned as content. For example, if the request is to retrieve a <*container*> resource, the <*container*> resource and the address(es) of the <*contentInstance*> child-resource(s) are provided.

  The offset, maximum number/size and maximum level shall be specified in ***Filter Criteria*** as *offset*, *limit*, and *level* condition, respectively, by the Originator. Processing of child resources, size limitations, maximum nesting level, and offset for the starting of child resource processing of **the attributes+child-resources** option shall apply to this option as well.

  This shall be only valid for a Retrieve/Delete operation.

- **child-resource-references:** Address(es) of the child resources and their descendants, without any representation of the actual requested resource shall be returned as content. For example, if the request is to retrieve a <*container*> resource, only the address(es) of the <*contentInstance*> child-resource(s) is provided.

  The offset, maximum number/size and maximum level shall be specified in ***Filter Criteria*** as *offset*, *limit*, and *level* condition, respectively, by the Originator. Processing of child resources, size limitations, maximum nesting level, and offset for the starting of child resource processing of **the attributes+child-resources** option shall apply to this option as well.

  This shall be only valid for a Retrieve/Delete operation.

  This option can be used within the context of resource discovery mechanisms (see clause 10.2.6).

- **nothing:** Nothing shall be returned as operational result content. If the Originator does not set the ***Result Content*** parameter in a Delete request message, this setting shall be the default value when the Receiver processes the request message. This setting shall be valid for a Create, Update, Delete, or Notify operation.

EXAMPLE: If the request is to delete a resource, this setting indicates that the response shall not include any content.

- **original-resource:** Representation of the original resource pointed by the *link* attribute in the announced resource shall be returned as content, without the address(es) of the child resource(s). This shall be only valid for a Retrieve operation where the ***To*** parameter targets the announced resource.

- **semantic-content:** Representation of semantic information that is the result of a semantic query as indicated by the setting of the ***Semantic Query Indicator*** parameter.

  Note that for any of the above options, Discovery access control is applied against discovery related procedures, while Retrieve access control procedures is applied against non-discovery related Retrieve operations.

  Note that the fitter criteria usage governs the purpose of a Retrieve operation.

Table 8.1.2-1: Summary of Result Content Values

Value	Create	Retrieve	Update	Delete	Notify	Retrieve (filterUsage='discovery')
attributes	default	default	default	valid	n/a	n/a
modified-attributes	valid	n/a	valid	n/a	n/a	n/a
hierarchical-address	valid	n/a	n/a	n/a	n/a	n/a
hierarchical-address+attributes	valid	n/a	n/a	n/a	n/a	n/a
attributes+child-resources	n/a	valid	n/a	valid	n/a	n/a
child-resources	n/a	valid	n/a	valid	n/a	n/a
attributes+child-resource-references	n/a	valid	n/a	valid	n/a	n/a
child-resource-references	n/a	valid	n/a	valid	n/a	valid (see note)
nothing	valid	n/a	valid	default	valid	n/a
original-resource	n/a	valid	n/a	n/a	n/a	n/a
semantic-content	n/a	valid	n/a	n/a	n/a	n/a
NOTE: See ETSI TS 118 104 [3] clause 7.5.2 for details.						

- ***Result Persistence*:** optional result persistence: indicates the time for which the response may persist to. The parameter is used in case of non-blocking request where the result attribute of the <request> resource should be kept at the CSE, for example, with the purpose of sharing, tracking and analytics.

  In the case the response of a request is required to be kept in the CSE, for example the procedures of <request> resource, <delivery> resource and <group> resource, the ***Result Persistence*** indicates the time duration for which the CSE keeps the response available after receiving it.

  Example usage of result persistence includes requesting sufficient persistence for analytics to process the response content aggregated asynchronously over time. If a result expiration time is specified, then the result persistence lasts beyond the result expiration time.

- ***Operation Execution Time*:** optional operation execution time: indicates the time when the specified operation ***Operation*** is to be executed by the target CSE. A target CSE shall execute the specified operation of a Request having its operational execution time indicator set, starting at the operational execution time. If the execution time has already passed or if the indicator is not set, then the specified operation shall be immediately executed, unless the request expiration time, if set, has been reached.

  Example usage of operational execution time includes asynchronous distribution of flows, which are to be executed synchronously at the operational execution time.

NOTE 4: Time-based flows could not be supported depending upon time services available at CSEs.

- ***Event Category*:** optional event category: Indicates the event category that should be used to handle this request. Event categories are impacting how Requests to access remotely hosted resources are processed in the CMDH CSF. Selection and scheduling of connections via CMDH are driven by policies that can differentiate event categories.

  Example usage of "event category" set to specific value X: When the request is demanding an operation to be executed on a Hosting CSE that is different from the current Receiver CSE, the request may be stored in the current Receiver CSE that is currently processing the request on the way to the Hosting CSE until it is allowed by provisioned policies for that event category X to use a communication link to reach the next CSE on a path to the Hosting CSE or until the request expiration timestamp is expired.

  The following values for ***Event Category*** shall have a specified pre-defined meaning:

  - ***Event Category* = immediate:** Requests of this category shall be sent as soon as possible and shall not be subject to any further CMDH processing, i.e. the request will not be subject to storing in CMDH buffers when communication over an underlying network is possible. In particular, CMDH processing will respect values for ***Request Expiration Timestamp***, ***Result Expiration Timestamp*** given in the original request and not fill in any default values if they are missing.

- ***Event Category* = bestEffort:** Requests of this category can be stored in CMDH buffers at the discretion of the CSE that is processing the request for an arbitrary time and shall be forwarded via Mcc on a best effort basis. The CSE does not assume any responsibility to meet any time limits for delivering the information to the next CSE. Also the maximum amount of buffered requests for this category is at the discretion of the processing CSE.

- ***Event Category*** = **latest:**

  - If this category is used in a request asking for a CRUD operation on a resource, the following shall apply:
    CRUD requests using this category shall undergo normal CMDH processing as outlined further below in the present document and in ETSI TS 118 104 [3] with a maximum buffer size of one pending request for a specific pair of ***From*** and ***To*** parameters that appear in the request. If a new request message is received by the CSE with a pair of parameters ***From*** and ***To*** that has already been buffered for a pending request, the newer request will replace the buffered older request.

  - If this category is used in a notification request triggered by a subscription, the following shall apply:
    Notification requests triggered by a subscription using this category shall undergo normal CMDH processing as outlined further below in the present document and in ETSI TS 118 104 [3] with a maximum buffer size of one pending notification request per subscription reference that appears in a notification request. If a new notification request is received by the CSE with a subscription reference that has already been buffered for a pending notification request, the newer request will replace the buffered older request.

  - If no further CMDH policies are provisioned for this event category, the forwarding process shall follow the 'bestEffort' rules defined above.

The M2M Service Provider shall be able to provision CMDH policies describing details for the usage of the specific Underlying Network(s) and the applicable rules as defined in the *[cmdhPolicy]* resource type for other ***Event Category*** values not listed above.

- ***Delivery Aggregation*:** optional delivery aggregation on/off: Use CRUD operations of <*delivery*> resources to express forwarding of one or more original requests to the same target CSE(s). When this parameter is not given in the request, the default behaviour is determined per the provisioned CMDH policy if available. If there is no such CMDH policy, then the default value is "aggregation off".

NOTE 5: Since ***Delivery Aggregation*** is optional, there could be a default value to be used when not present in the Request. This parameter could not be exposed to AEs via Mca.

Example usage of delivery aggregation set on: The CSE processing a request shall use aggregation of requests to the same target CSE by requesting CREATE of a <*delivery*> resource on the next CSE on the path to the target CSE.

- ***Group Request Identifier*:** Identifier added to a request when it is fanned out to each member of the group in order to detect loops and avoid duplicated handling of the operation in cases where there are circular references between groups and where there are common members between groups that have a parent-child relationship.

This parameter shall only be added to requests by a Group Hosting CSE, and then only when it is processing requests targeted at a <fanOutPoint> virtual resource that does not already have a Group Request Identifier parameter.

A target CSE shall process any Group Request Identifier that it receives as described in clause 10.2.7.1 step 004.

- ***Group Request Target Members:*** optional group request target members: Indicates subset of members of a group for which fanout is to be executed. Example usage of Group Request Target Members: if fanout operation failed for some of the members then the Originator may use this parameter to execute fanout for failed members of a previous fanout operation.

- ***Filter Criteria*:** optional filter criteria: conditions for filtered operations which are described in table 8.1.2-2. This is used for resource discovery (clause 10.2.6) and general retrieve, update, delete requests (clauses 10.1.3, 10.1.4 and 10.1.5).

The Filter Criteria set includes matching conditions and filter handling conditions. Matching conditions are evaluated against resources and, when true, determine the matched resources which compose the matching result. The filter handling conditions provide additional input used to determine the filtering result (e.g. maximum number of resources to be included in the filtering result). The filtering result may be composed of one or more resources.

Example usage of retrieve requests with filter criteria using *modifiedSince* condition tag: if a target resource is modified since 12:00 then the Hosting CSE will identify it as a matched resource.

- ***Desired Identifier Result Type:*** Optional result format of resource identifiers. This parameter indicates the format of the resource identifiers in the result of operations that can return a list of resource identifiers or Child Resource References. This parameter shall take on one of the following values reflecting the options in clause 9.3.1:

  - Structured identifier format.

  - Unstructured identifier format.

  The absence of the parameter implies that the result shall be in the form of a Structured identifier format.

- ***Token Request Indicator:*** Optional parameter used to indicate that the Originator supports the Token Request procedure, and the Originator may attempt the Token Request procedure if the Receiver provides a ***Token Request Information*** parameter in the response.

- ***Tokens:*** Optional parameter used to transport ESData-protected *Tokens* applicable to the request for use in Indirect Dynamic Authorization.

- ***Token IDs:*** Optional parameter used to transport *Token-IDs* applicable to the request for use in Indirect Dynamic Authorization.

- ***Local Token IDs:*** Optional parameter used to transport Local-Token-IDs applicable to the request for use in Indirect Dynamic Authorization.

- ***Authorization Signature Indicator*****:** Optional parameter used to indicate the capability for creating AuthorRelMapRecord when Originator is an AE. If the Hosting CSE does not support this parameter, the Hosting CSE should ignore it. The details of the AuthorRelMapRecord are described in clause 7.3.2.2 of ETSI TS 118 103 [2].

- ***Authorization Signature*****:** Optional parameter used to transport the signatures for Token(s) or TokenID(s) generated using the certificate of the AE or a MIC generated using a symmetric key shared between the AE and DAS server.

- ***Authorization Relationship Indicator*****:** Optional parameter used to indicate that the relationship between the AE and the Token(s) are maintained in the DAS server.

- ***Semantic Query Indicator:*** Optional parameter used to indicate whether a RETRIEVE request is a semantic query or a semantic resource discovery. If the request contains this parameter with the value set to "TRUE", the request shall be processed as a semantic query based on the SPARQL query statement included in the "*semanticsFilter*" condition tag; other *Filter Criteria* and the following parameters shall be ignored: *Desired Identifier Result Type*, *Delivery Aggregation*. The parameter *Result Content* shall be set to **semantic-content** to indicate that the response message contains the result of a semantic query request. If it is not set or set to "FALSE" the request shall be processed as a semantic resource discovery.

- ***Release Version Indicator:*** This parameter is used to indicate the oneM2M release version that this request message conforms to. Starting with Release 2 this parameter is mandatory. The release version indicated shall apply to all oneM2M defined request parameters and certain types of content carried in the ***Content*** request parameter. Within the ***Content*** request parameter, the release version indicated shall apply to all oneM2M defined elements (e.g. notifications) and resource types with the exception of <*flexContainer*> and <*mgmtObj*> specializations which have their own version implicitly defined by their respective *containerDefinition* and *mgmtSchema* attributes. In addition, the release version indicated does not apply to resource types or specializations defined external to oneM2M.

- ***Vendor Information:*** This optional parameter is available to convey vendor specific information. The use of this parameter is not defined by oneM2M specifications.

Table 8.1.2-2: *Filter Criteria* conditions

Condition tag	Multiplicity	Description
Matching Conditions		
createdBefore	0..1	The *creationTime* attribute of the matched resource is chronologically before the specified value.
createdAfter	0..1	The *creationTime* attribute of the matched resource is chronologically after the specified value.
modifiedSince	0..1	The *lastModifiedTime* attribute of the matched resource is chronologically after the specified value.
unmodifiedSince	0..1	The *lastModifiedTime* attribute of the matched resource is chronologically before the specified value.
stateTagSmaller	0..1	The *stateTag* attribute of the matched resource is smaller than the specified value.
stateTagBigger	0..1	The *stateTag* attribute of the matched resource is bigger than the specified value.
expireBefore	0..1	The *expirationTime* attribute of the matched resource is chronologically before the specified value.
expireAfter	0..1	The *expirationTime* attribute of the matched resource is chronologically after the specified value.
labels	0..1	The *labels* attribute of the matched resource matches the specified value.
labelsQuery	0..1	The value is an expression for the filtering of *labels* attribute of resource when it is of key-value pair format. The expression is about the relationship between label-key and label-value which may include equal to or not equal to, within or not within a specified set etc. For example, label-key equals to label value, or label-key within {label-value1, label-value2}. Details are defined in ETSI TS 118 104 [3].
childLabels	0..1	A child of the matched resource has *labels* attributes matching the specified value. The evaluation is the same as for the *labels* attribute above. Details are defined in ETSI TS 118 104 [3].
parentLabels	0..1	The parent of the matched resource has *labels* attributes matching the specified value. The evaluation is the same as for the *labels* attribute above. Details are defined in ETSI TS 118 104 [3].
resourceType	0..n	The *resourceType* attribute of the matched resource is the same as the specified value. It also allows differentiating between normal and announced resources.
childResourceType	0..n	A child of the matched resource has the *resourceType* attribute the same as the specified value.
parentResourceType	0..1	The parent of the matched resource has the *resourceType* attribute the same as the specified value.
sizeAbove	0..1	The *contentSize* attribute of the <*contentInstance*> matched resource is equal to or greater than the specified value.
sizeBelow	0..1	The *contentSize* attribute of the <*contentInstance*> matched resource is smaller than the specified value.
contentType	0..n	The *contentInfo* attribute of the <*contentInstance*> matched resource matches the specified value.
attribute	0..n	This is an attribute of resource types (clause 9.6). Therefore, a real tag name is variable and depends on its usage and the value of the attribute can have wild card *. E.g. *creator* of container resource type can be used as a filter criteria tag as "creator=Sam", "creator=Sam*", "creator=*Sam".
childAttribute	0..n	A child of the matched resource meets the condition provided. The evaluation of this condition is similar to the *attribute* matching condition above.
parentAttribute	0..n	The parent of the matched resource meets the condition provided. The evaluation of this condition is similar to the *attribute* matching condition above.

Condition tag	Multiplicity	Description
semanticsFilter	0..n	Both semantic resource discovery and semantic query use *semanticsFilter* to specify a query statement that shall be specified in the SPARQL query language [5]. When a CSE receives a RETRIEVE request including a *semanticsFilter*, and the ***Semantic Query Indicator*** parameter is also present in the request, the request shall be processed as a semantic query; otherwise, the request shall be processed as a semantic resource discovery. In the case of semantic resource discovery targeting a specific resource, if the semantic description contained in the <semanticDescriptor> of a child resource matches the semanticFilter, the URI of this child resource will be included in the semantic resource discovery result. In the case of semantic query, given a received semantic query request and its query scope, the SPARQL query statement shall be executed over aggregated semantic information collected from the semantic resource(s) in the query scope and the produced output will be the result of this semantic query. Examples for matching semantic filters in SPARQL to semantic descriptions can be found in oneM2M TR-0007 [i.28].
filterOperation	0..1	Indicates the logical operation (AND/OR) to be used for different condition tags. The default value is logical AND.
contentFilterSyntax	0..1	Indicates the Identifier for syntax to be applied for content-based discovery.
contentFilterQuery	0..1	The query string shall be specified when *contentFilterSyntax* parameter is present.
Filter Handling Conditions		
filterUsage	0..1	Indicates how the filter criteria is used. If provided, possible values are 'discovery' and 'IPEOnDemandDiscovery'. If this parameter is not provided, the Retrieve operation is a generic retrieve operation and the content of the child resources fitting the filter criteria is returned. If *filterUsage* is 'discovery', the Retrieve operation is for resource discovery (clause 10.2.6), i.e. only the addresses of the child resources are returned. If *filterUsage* is 'IPEOnDemandDiscovery', the other filter conditions are sent to the IPE as well as the discovery Originator ID. When the IPE successfully generates new resources matching with the conditions, then the resource address(es) shall be returned. This value shall only be valid for the Retrieve request targeting an <AE> resource that represents the IPE.
limit	0..1	The maximum number of resources to be included in the filtering result. This may be modified by the Hosting CSE. When it is modified, then the new value shall be smaller than the suggested value by the Originator.
level	0..1	The maximum level of resource tree that the Hosting CSE shall perform the operation starting from the target resource (i.e. ***To*** parameter). This shall only be applied for Retrieve operation. The level of the target resource itself is zero and the level of the direct children of the target is one.
offset	0..1	The number of direct child and descendant resources that a Hosting CSE shall skip over and not include within a Retrieve response when processing a Retrieve request to a targeted resource.
applyRelativePath	0..1	This attribute contains a resource tree relative path (e.g. ../tempContainer/LATEST). This condition applies after all the matching conditions have been used (i.e. a matching result has been obtained). The attribute determines the set of resource(s) in the final filtering result. The filtering result is computed by appending the relative path to the path(s) in the matching result. All resources whose Resource-IDs match that combined path(s) shall be returned in the filtering result. If the relative path does not represent a valid resource, the outcome is the same as if no match was found, i.e. there is no corresponding entry in the filtering result.

The rules when multiple matching conditions are used together shall be as follows:

- Different condition tags shall use the "AND/OR" logical operation based on the *filterOperation* specified;

  e.g. *createdBefore* = "time1" AND *unmodifiedSince* = "time2" if *filterOperation* = "AND" or "NULL", or *createdBefore* = "time1" OR *unmodifiedSince* = "time2" if *filterOperation* = "OR".

- Same condition tags shall use the "OR" logical operation, i.e. *filterOperation* does not apply to same conditions.

No mixed AND/OR filter operation will be supported.

Once the Request is delivered, the Receiver shall analyse the Request to determine the target resource.

If the target resource is addressing another M2M Node, the Receiver shall route the request appropriately.

If the target resource is addressing the Receiver, it shall:

- Check the existence of ***To*** addressed resource.
- Identify the resource type by ***Resource Type***.
- Check the privileges for ***From*** Originator to perform the requested operation.
- Perform the requested operation (using ***Content*** content when provided) according to the provided request parameters as described above.
- Depending on the request result content, respond to the Originator with indication of successful or unsuccessful operation results. In some specific cases (e.g. limitation in the binding protocol or based on application indications), the Response could be avoided.

Table 8.1.2-3 summarizes the parameters specified in this clause for the Request message, showing any differences as applied to C, R, U, D or N operations. "M" indicates mandatory, "O" indicates optional, "N/A" indicates "not applicable".

Table 8.1.2-3: Summary of Request Message Parameters

Request message parameter		Operation				
		Create	Retrieve	Update	Delete	Notify
Mandatory	***Operation*** - operation to be executed	M	M	M	M	M
	To - the address of the target resource on the target CSE	M	M	M	M	M
	From - the identifier of the message Originator	O See note 1	M	M	M	M
	Request Identifier - uniquely identifies a Request message	M	M	M	M	M
Operation dependent	***Content*** - to be transferred	M	O	M	N/A	M
	Resource Type - of resource to be created	M	N/A	N/A	N/A	N/A
Optional	***Originating Timestamp*** - when the message was built	O	O	O	O	O
	Request Expiration Timestamp - when the request message expires	O	O	O	O	O
	Result Expiration Timestamp - when the result message expires	O	O	O	O	O
	Operational Execution Time - the time when the specified operation is to be executed by the target CSE	O	O	O	O	O
	Response Type - type of response that shall be sent to the Originator	O	O	O	O	O

Request message parameter		Operation				
		Create	Retrieve	Update	Delete	Notify
	Result Persistence - the duration for which the reference containing the responses is to persist	O	O	O	O	N/A
	Result Content - the expected components of the result	O	O	O	O	N/A
	Event Category - indicates how and when the system should deliver the message	O	O	O	O	O
	Delivery Aggregation - aggregation of requests to the same target CSE is to be used	O	O	O	O	O
	Group Request Identifier - Identifier added to the group request that is to be fanned out to each member of the group	O	O	O	O	O
	Group Request Target Members-indicates subset of members of a group	O	O	O	O	N/A
	Filter Criteria - conditions for filtered retrieve operation	N/A	O	O	O	N/A
	Desired Identifier Result Type - format of resource identifiers returned	N/A	O	N/A	N/A	N/A
	Token Request Indicator - indicating that the Originator may attempt Token Request procedure (for Dynamic Authorization) if initiated by the Receiver	O	O	O	O	O
	Tokens - for use in dynamic authorization	O	O	O	O	O
	Token IDs - for use in dynamic authorization	O	O	O	O	O
	Role IDs - for use in role based access control	O	O	O	O	O
	Local Token IDs - for use in dynamic authorization	O	O	O	O	O
	Authorization Signature Indicator - for use in Authorization Relationship Mapping	O	O	O	O	N/A
	Authorization Signature - for use in Authorization Relationship Mapping	O	O	O	O	N/A
	Authorization Relationship Indicator - for use in Authorization Relationship Mapping	O	O	O	O	N/A
	Semantic Query Indicator - for use in semantic queries	N/A	O	N/A	N/A	N/A
	Release Version Indicator - the oneM2M release version that this request message conforms to.	M See note 2	M See note 2	M See note 2	M See note 2	M See note 2
	Vendor Information	O	O	O	O	O

NOTE 1: *From* parameter is optional in case of an AE CREATE request and mandatory for all other requests.
NOTE 2: ***Release Version Indicator*** parameter is not present for the case when a request is targeting a Rel-1 entity and mandatory for all other cases.

8.1.3 Response

The Response received by the Originator of a Request accessing resources over the Mca and Mcc reference points shall contain mandatory and may contain optional parameters. Certain parameters may be mandatory or optional depending upon the Requested operation (CRUDN) or the mandatory response code. In this clause, the mandatory parameters are detailed first, followed by those that are conditional, and then by those that are optional:

Mandatory Parameters:

- ***Response Status Code*:** response status code: This parameter indicates that a result of the requested operation is successful, unsuccessful, acknowledgement or status of processing such as authorization timeout, etc.:

  - A **successful** code indicates to the Originator that the Requested operation has been executed successfully by the Hosting CSE.

  - An **unsuccessful** code indicates to the Originator that the Requested operation has not been executed successfully by the Hosting CSE.

  - An **acknowledgement** indicates to the Originator that the Request has been received and accepted by the attached CSE, i.e. by the CSE that received the Request from the issuing Originator directly, but the Request operation has not been executed yet. The success or failure of the execution of the Requested operation is to be conveyed later.

  Details of successful, unsuccessful and acknowledge codes are provided in clause 6.6 of ETSI TS 118 104 [3].

- ***Request Identifier*:** Request Identifier. The ***Request Identifier*** in the Response shall match the ***Request Identifier*** in the corresponding Request.

Conditional Parameters:

- ***Content*:** resource content:

  - If ***Response Status Code*** is *successful* then:

    The ***Content*** response parameter may be present in a Create/Update/Delete Response and the information in this ***Content*** response parameter depends on the value of the ***Result Content*** request parameter of the corresponding Request. If the value of the ***Result Content*** request parameter is "nothing" or if the ***Result Content*** request parameter is not present in a Delete Request, the ***Content*** response parameter shall not be present. Otherwise, the ***Content*** response parameter shall be present. in the following cases:

  - The ***Content*** parameter shall be present in a Retrieve Response and the information in this ***Content*** parameter depends on the ***Result Content*** value of the corresponding Retrieve Request. If ***Response Status Code*** is *unsuccessful* then the ***Content*** parameter may be present in a Response to provide more error information.

  - If ***Response Status Code*** is *acknowledgment*, then the ***Content*** parameter:

    - Shall contain the address of a <*request*> resource if the response was an acknowledgement of a non-blocking request and the <*request*> resource type is supported by the Receiver CSE.

    - Is not present otherwise.

- ***Content Status*:** This parameter shall be present in the response to a Retrieve operation when the returned content is partial. More specifically, this parameter takes the value of partial depending on the ***Content*** parameter.

  - If ***Response Code*** is *successful* then and the ***Content*** parameter is present due to the following case:

    - **Retrieve (R):** ***Content*** is the retrieved resource content or aggregated contents of discovered resources and the retrieved content is partial

    Then ***Content Status*** parameter shall be present in the response for a **Retrieve (R)** operation

- ***Content Offset*:** This parameter includes the point where a Hosting CSE left off with processing a Retrieve operation that resulted in a response with partial content (i.e. due to reaching the limit on the number of resources allowed in a response). This parameter shall be expressed as a number which can be used in a subsequent Retrieve request. The parameter shall be used by the Hosting CSE to skip over the specified number of direct child and descendant resources of a targeted resource and retrieve the remaining direct child and descendant resources. Its value depends on the information included in the ***Content Status*** parameter. When a Hosting CSE includes a Content *Offset* parameter within a Retrieve response to indicate partial results, and an originator includes this value within an *offset* ***Filter Criteria*** condition in a subsequent Retrieve request to indicate to the Hosting CSE where to continue processing, the Hosting CSE is not obligated to ensure consistency between any prior partial results it returned and the results it returns for a continued request. For example, a Hosting CSE may receive and process other requests (e.g. creation of new resource or deletion of existing resources) during the interim of when it returns a partial result to a Retrieve request and when it receives a subsequent request with an *offset* ***Filter Criteria*** condition to continue the Retrieve request. If these other requests target the same resources as the Retrieve, this can impact the offset calculations on the Hosting CSE. As a result, the combined set of partial results received by an originator may have duplicate or missing results:

  - If ***Content Status*** parameter is complete, then this parameter shall not be included.
  - If ***Content Status*** parameter is partial, then this shall include the offset where processing can restart for the remaining descendant resources in the resource tree.

  Then Content ***Offset*** parameter shall be present in the response for a **Retrieve (R)** operation.

Optional parameters:

- ***To:*** ID of the Originator or the Transit CSE.
- ***From:*** ID of the Receiver.

The ***To*** and ***From*** parameters can be used in the response for specific protocol bindings (e.g. MQTT):

- ***Originating Timestamp:*** originating timestamp of when the message was built.
- ***Result Expiration Timestamp:*** result expiration timestamp. The Receiver shall echo the result expiration timestamp if set in the Request message, or may set the result expiration timestamp itself.

  Example usage of the Receiver setting the result expiration timestamp is when the value of the delivery time is dependent upon some changing Receiver context e.g. Result message deadline for aircraft position based upon velocity.

- ***Event Category:*** event category: Indicates the event category that should be used to handle this response. The definition of event category is the same as in the case of requests in clause 8.1.2.

  Example usage of "event category" set to specific value X: When the response is targeted to an entity that is different from the Transit CSE currently processing the response message and is not an AE registered with the Transit CSE that is currently processing the response message, the response may be stored in the Transit CSE that is currently processing the response on the way to the destination of the response message until it is allowed by provisioned policies for that event category X to use a communication link to reach the next CSE on a path to the destination of the response message or until the result expiration timestamp is expired.

- ***Token Request Information:*** Optional parameter which may be used for requesting Tokens from Dynamic Authorization Systems.
- ***Assigned Token Identifiers*:** Optional parameter containing the mapping from assigned Local-Token-IDs to corresponding Token-IDs.
- ***Authorization Signature Request Information*:** Optional parameter used to request the ***Authorization Signature(s)*** of the Token(s) from an AE which is the holder of the Token(s).

- ***Release Version Indicator:*** This parameter is used to indicate the oneM2M release version that this response message is compliant with. Starting with Release 2 this parameter is mandatory. The release version indicated shall apply to all oneM2M defined response parameters and certain types of content carried in the ***Content*** response parameter. Within the ***Content*** response parameter, the release version indicated shall apply to all oneM2M defined elements (e.g. notifications) and resource types with the exception of <*flexContainer*> and <*mgmtObj*> specializations which have their own version implicitly defined by their respective *containerDefinition* and *mgmtSchema* attributes. The release version indicated does not apply to resource types or specializations defined external to oneM2M.

- ***Vendor Information:*** This optional parameter is available to convey vendor specific information. The use of this parameter is not defined by oneM2M specifications.

Table 8.1.3-1 summarizes the parameters specified in this clause for the Response messages, showing any differences as applied to successful C, R, U, D or N operations, and unsuccessful operations. "M" indicates mandatory, "O" indicates optional, "N/A" indicates "not applicable".

Table 8.1.3-1: Summary of Response Message Parameters

Response message parameter/success or not	Ack	Successful Operation					Unsuccessful Operation	
		Create	Retrieve	Update	Delete	Notify	Create / Retrieve / Update / Delete	Notify
Response Status Code - successful, unsuccessful, ack	M	M	M	M	M	M	M	M
Request Identifier - uniquely identifies a Request message	M	M	M	M	M	M	M	M
Content - to be transferred	O (Address of *<request>* resource if response is ACK of a non-blocking request)	O (The address and/or the content of the created resource, an address list, or aggregated response primitives) (see notes 3, 4)	M (The retrieved resource content, aggregated contents, an address list or aggregated response primitives) (see notes 3, 4)	O (A complete or partial resource representation, an address list or aggregated response primitives.) (see notes 3, 4)	O (The content actually deleted, an address list or aggregated response primitives) (see notes 3, 4)	O (see note 1, end-to-end security protocol message)	O (Additional error info)	O (see note 1, additional error info secured using ESPrim)
To - the identifier of the Originator or the Transit CSE that sent the corresponding non-blocking request	O	O	O	O	O	O	O	O
From - the identifier of the Receiver	O	O	O	O	O	O	O	O
Originating Timestamp - when the message was built	O	O	O	O	O	O	O	O
Result Expiration Timestamp - when the message expires	O	O	O	O	O	O	O	O
Event Category - what event category shall be used for the response message	O	O	O	O	O	O	O	O
Content Status	N/A	N/A	O	N/A	N/A	N/A	N/A	N/A
Content Offset	N/A	N/A	O	N/A	N/A	N/A	N/A	N/A
Token Request Information	N/A	N/A	N/A	N/A	N/A	N/A	O	O
Assigned Token Identifiers	N/A	O	O	O	O	O	O	O
Authorization Signature Request Information	N/A	N/A	N/A	N/A	N/A	N/A	O	N/A
Release Version Indicator - the oneM2M release version that this response message conforms to	M (see note 2)	M (see note 2)	M (see note 2)	M (see note 2)	M (see note 2)	M (see note 2)	M (see note 2)	M (see note 2)
Vendor Information	O	O	O	O	O	O	O	O

NOTE 1: This parameter is present if the response contains an end-to-end security protocol message. Otherwise this parameter is not applicable.
NOTE 2: ***Release Version Indicator*** parameter is not present for the case when a response is targeting a Rel-1 entity and mandatory for all other cases.
NOTE 3: A resource address list with zero or more resource addresses may be provided for some Discovery-related procedures (see clause 10.2.6 for conditions and details).
NOTE 4: An aggregation of response primitives may be provided for group operations (see clause 10.2.7 for conditions and details) and some Discovery-related procedures (see clause 10.2.6 for conditions and details).

8.2 Procedures for Accessing Resources

8.2.0 Overview

This clause describes the procedures for accessing the resources. The term "hop" in the descriptions here refers to the number of Transit CSEs traversed by a request on its route from the Originator to the Hosting CSE. Traversal implies that the request was forwarded from one CSE to either its Registrar CSE or Registree CSE. For example, when a CSE initiated a request and the Hosting CSE is its Registrar CSE, the hop count is zero.

The Receiver CSE shall forward the received request in the following case:

- The ***To*** parameter in the request contains a CSE-ID and it does not represent the ID of the Receiver CSE.

The Receiver CSE shall handle the received request in the following cases:

- The ***To*** parameter in the request contains a CSE-ID and it represents the ID of the Receiver CSE.
- The ***To*** parameter in the request does not contain a CSE-ID.

All the descriptions and message flows in this clause are illustrative for the direction from a Registree acting as an Originator to a Registrar acting as a Receiver only. The flows from a Registrar CSE to a Registree CSE are symmetric with respect to the one described in this clause. Both the IN-CSE and MN-CSE have the ability to route a received request or response messages to one of their Registrees. If the Hosting CSE is not known by an MN-CSE that receives a request or response message, that MN-CSE shall forward the message to its own Registrar CSE by default.

8.2.1 Accessing Resources in CSEs - Blocking Requests

8.2.1.0 Overview

For the procedures described herein, the addressed resource can be stored in different CSEs. Table 8.2.1.0-1 describes the possible scenarios, where the addressed resource may be on the Registrar CSE or on a CSE located elsewhere in the oneM2M System.

In this clause - for simplicity - it is assumed that the Originator of a Request can always wait long enough to get a Response to the Request after the requested operation has finished. This implies potentially long or unknown blocking times (time for which a pending Request has not been responded to) for the Originator of a Request.

For scenarios that avoid such possibly long blocking times, clause 8.2.2 specifies mechanisms to handle synchronous and asynchronous resource access procedures via returning appropriate references.

Table 8.2.1.0-1: Accessing Resources in different CSEs, from Registree to Registrar CSE

Number of Transit CSEs	Description	Reference
No Hops	• The Originator of the Request accesses a resource. • The Originator of the Request can be an AE or a CSE. • Registrar CSE and Hosting CSE are the same entity. • The Hosting CSE checks the Access Control Privileges for accessing the resource. • Depending on the expected result content, the Hosting CSE responds to the Originator of the Request, either with a success or failure Response.	Figure 8.2.1.0-1
1 Hop	• The Originator of the Request accesses a resource. • The Originator of the Request may be an AE or a CSE. • Registrar CSE and hosting CSEs are different entities. • Registrar CSE forwards the Request to the Hosting CSE if the Registrar CSE is registered with the Hosting CSE, for accessing the resource. • Hosting CSE checks the Access Control Privileges for accessing the resource and depending on the expected result content respond with a success or failure Response.	Figure 8.2.1.0-2
Multi Hops	• The Originator of the Request accesses a resource. • The Originator of the Request may be an AE or a CSE. • Registrar CSE, Transit CSE(s) and the Hosting CSE are different entities. • Registrar CSE: – Forwards the request to a Registree Transit-1 CSE if the Hosting CSE is a descendant of a Registree Transit-1 CSE; or – Forwards the request to its Registrar Transit-1 CSE if the Hosting CSE is not a descendant of any Registree Transit-1 CSE • Transit-N CSE: – Forwards the request to the Hosting CSE if it is registered with the Hosting CSE; or – Forwards the request to a Registree Transit-(N+1) CSE if the Hosting CSE is a descendant of a Registree Transit-(N+1) CSE – Forwards the request to its Registrar Transit-(N+1) CSE if the Hosting CSE is not a descendant of any Registree Transit-(N+1) CSEs and if the Transit-(N-1) CSE is not the Registrar CSE of the Transit-N CSE – Return an error if the Hosting CSE is not a descendant of any Registree Transit-(N+1) CSEs and if the Transit-(N-1) CSE is the Registrar CSE of the Transit-N CSE • In case the Request reaches the IN-CSE, the IN-CSE: – Performs the processing defined under 'Hosting CSE' below if the targeted resource is hosted on IN-CSE – Forwards the request to another IN-CSE if the resource belongs to another M2M SP based on the routing procedure defined in clause 8.2.1.2; or – Forwards the request to the Hosting CSE if the Hosting CSE is registered with the IN-CSE; or – Forwards the request to a Registree Transit-(N+1) CSE if the Hosting CSE is a descendant of a Registree Transit-(N+1) CSE – Return an error if the Hosting CSE is not a descendant of a Registree Transit-(N+1) CSE or the request cannot be forwarded to another IN-CSE in another M2M SP domain • Hosting CSE checks the Access Control Privileges for accessing the resource and depending on the expected result content respond with a success or failure Response.	Figure 8.2.1.0-3

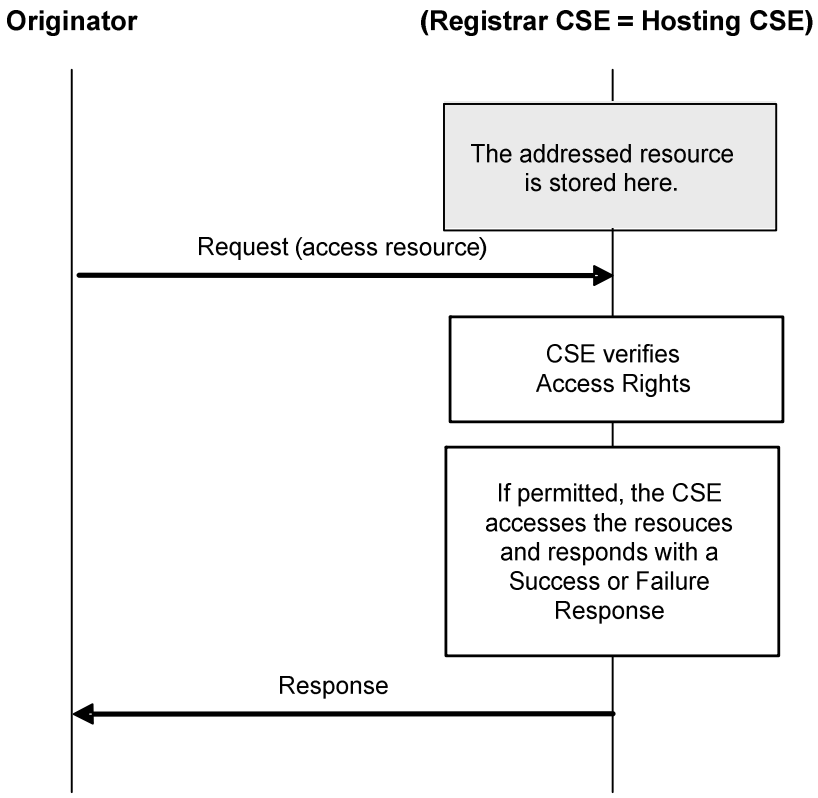


Figure 8.2.1.0-1: Originator accesses a resource on the Registrar CSE (No Hops)

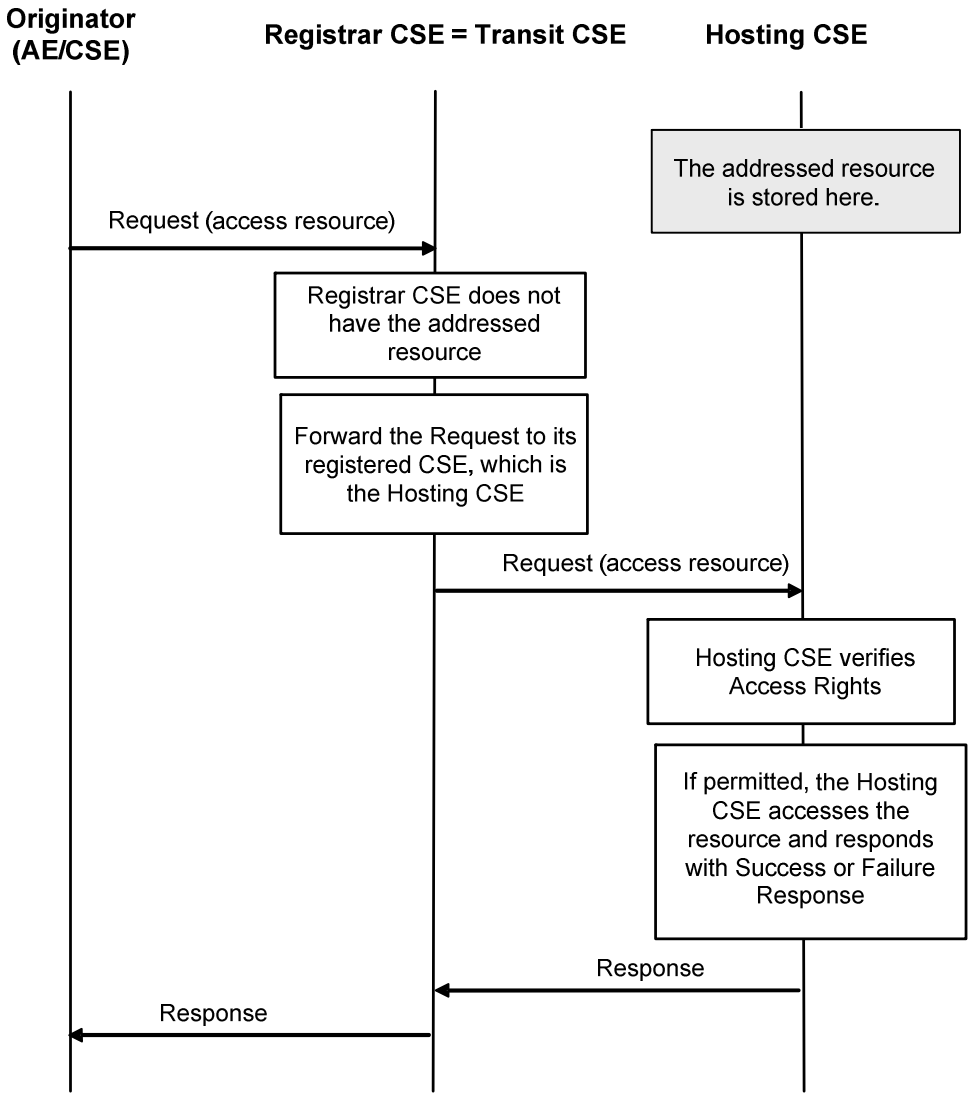


Figure 8.2.1.0-2: AE/CSE accesses a resource at the Hosting CSE (One Hop)

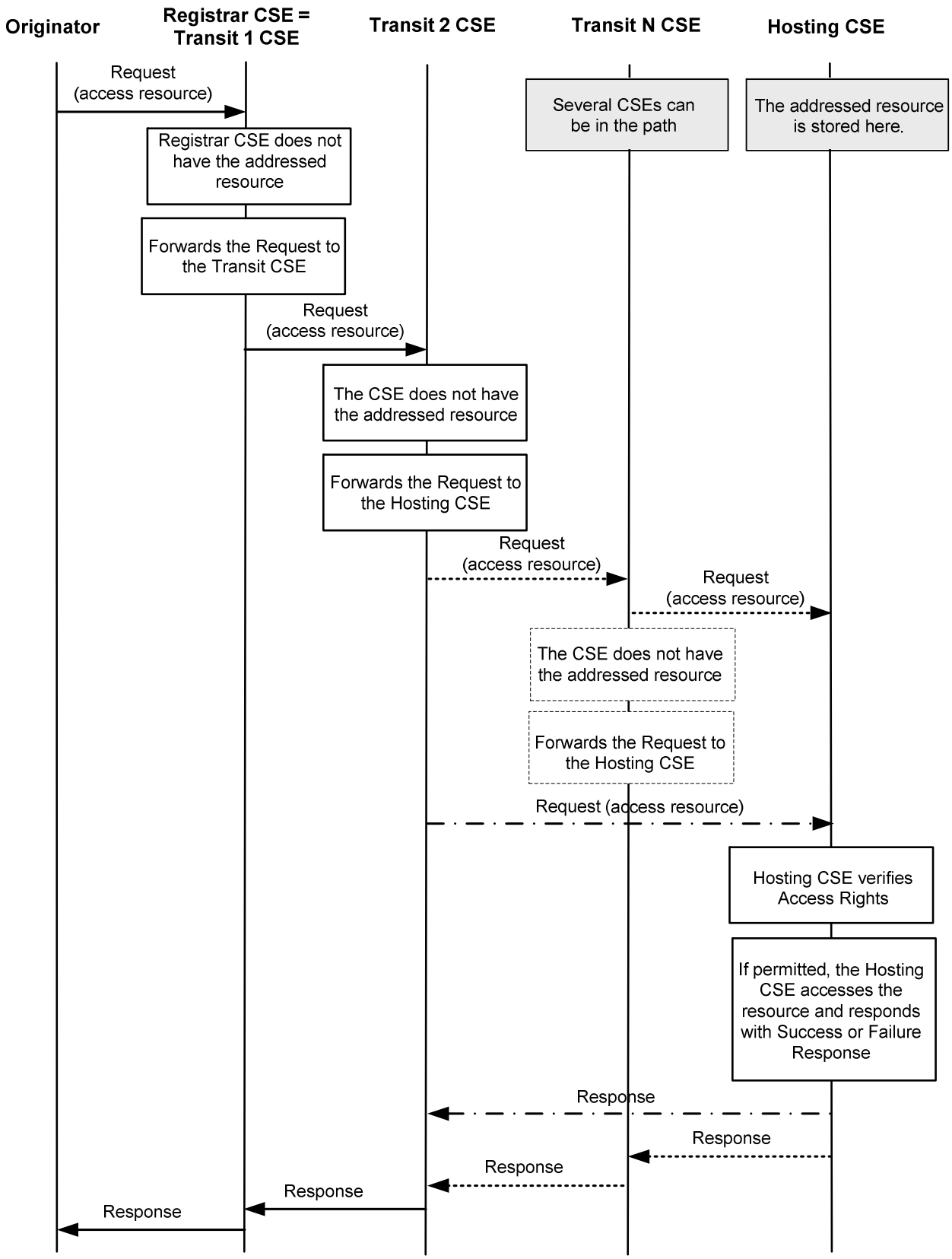


Figure 8.2.1.0-3: Originator accesses a resource at the Hosting CSE (Multi Hops)

8.2.1.1 M2M Requests Routing Policies

A CSE shall route M2M requests targeting another CSE in the same SP domain by forwarding the request to the next hop towards the target CSE by first checking each of its <remoteCSE> resources to determine whether the CSE-ID specified in the ***To*** parameter of the request matches either the *CSE-ID* or *descendantCSEs* attributes of a <remoteCSE> resource. If a match is found, the CSE shall retarget the request to the *pointOfAccess* of the matching <remoteCSE> resource. If a match is not found, and the CSE received the request from an AE or a descendant CSE, and the CSE is not the IN-CSE, then it shall retarget the request to its Registrar CSE. If a match is not found and the CSE is the IN-CSE, then the CSE shall not forward the request and it shall respond with an error. If a match is not found and the CSE is not the IN-CSE and the CSE receives the request from its registrar CSE, then the CSE shall not forward the request and it shall respond with an error. Anytime a CSE re-targets a request to another CSE, it shall keep track of the requestID and the corresponding Originator's ID. This information shall be used to route re-targeted responses back to the Originator.

8.2.1.2 Inter SP Domain M2M Request Routing

If a CSE in the originating SP domain is not the IN-CSE and it receives a request targeting another CSE in a different SP domain, it shall retarget the request to its Registrar CSE. This shall be done by retargeting the requests to the *pointOfAccess* of the <remoteCSE> of its Registrar CSE. If a CSE in the originating SP domain is the IN-CSE and it receives a request targeting another CSE in a different SP domain, the IN-CSE shall routes the request to the IN-CSE in the targeted SP domain using either the DNS-based procedures or the inter-M2M SP registration procedures defined in clause 6.5. For the inter-M2M SP registration based procedure, the IN-CSE in the originating SP domain shall forward the request to the IN-CSE in the target SP domain by checking each of its <remoteCSE> resources to determine whether the SP-ID specified in the ***To*** parameter of the request matches the SP-ID specified in the *CSE-ID* attribute containing a SP-relative CSE-ID. If the IN-CSE finds a match, it shall retarget the request to the *pointOfAccess* of the matching <remoteCSE> resource. If the IN-CSE does not find a match, then it shall not forward the request and it shall respond with an error. An IN-CSE receiving a request from an IN-CSE in another SP domain, shall route the request to the targeted CSE residing in its own domain using the Intra SP Domain routing as described in clause 8.2.1.1 to route the request to the CSE in the targeted SP domain. Anytime a CSE re-targets a request to another CSE in its own SP domain or another SP domain, it shall keep track of the requestID and the corresponding Originator's ID. This information shall be used to route re-targeted responses back to the Originator.

8.2.2 Accessing Resources in CSEs - Non-Blocking Requests

8.2.2.1 Response with Acknowledgement and optional Reference to Request Context and Capturing Result of Requested Operation

In case the Originator of a Request has asked for only a response with an Acknowledgement indicating acceptance of the Request and an optional reference to the context where the result of the requested operation is expected - i.e. when the ***Response Type*** parameter of the request as defined in clause 8.1.2 is set to *nonBlockingRequestSynch* or to *nonBlockingRequestAsynch* - it is necessary to provide a prompt response to the Originator with an Acknowledgement - and in case the <*request*> resource type is supported by the Receiver CSE also, with a reference to an internal resource on the Receiver CSE, so that the Originator can retrieve the status of the request and the outcome of the requested operation at a later time. The details of such an internal resource are defined in clause 9.6.12. In case the <*request*> resource type is supported, the reference is provided in the response to the Request within the ***Content*** parameter of the Response. The abbreviation "Req-Ref" is used for simplicity in the figures of the following clauses.

Two different cases to allow the Originator of a non-blocking request to retrieve the result of a requested operation are defined in the following two clauses.

8.2.2.2 Synchronous Case

In the synchronous case, it is assumed that the Originator of a Request is not able to receive asynchronous messages, i.e. all exchange of information between Originator and Receiver CSE needs to be initiated by the Originator.

In the synchronous case, a Receiver CSE that does not support the <*request*> resource type shall respond an error indicating that is not supported.

In that case the information flow depicted in figure 8.2.2.2-1 is applicable. For the flow depicted in figure 8.2.2.2-1 it is assumed that completion of the requested operation happens before the Originator is trying to retrieve the result of the requested operation with a second Request referring to the "Req-Ref" provided in the Response to the original Request.

Another variation of the information flow for the synchronous case is depicted in figure 8.2.2.2-2. In this variation it is assumed that the requested operation completes after the second request but before the third request sent by the Originator.

Equivalent information flows are valid also for cases where the target resource of the requested operation is not hosted on the Receiver CSE. From an Originator's perspective there is no difference as the later retrieval of the result of a requested operation would always be an exchange of Request/Response messages between the Originator and the Receiver CSE using the reference to the original request.

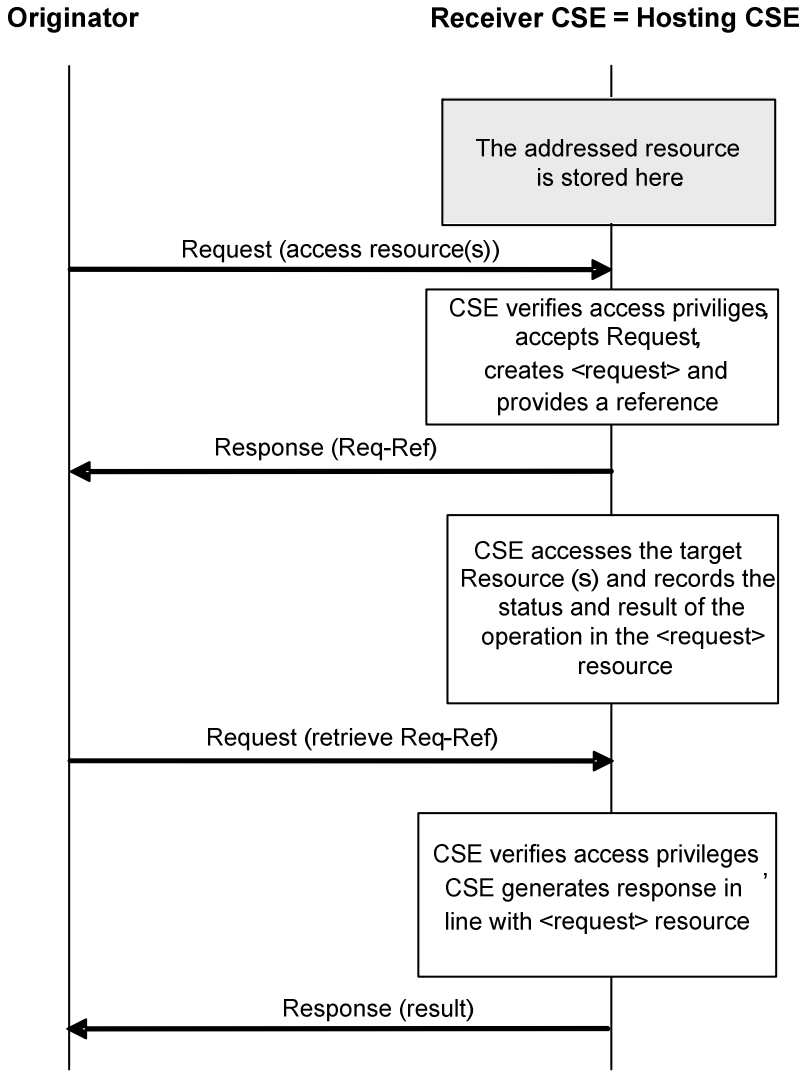


Figure 8.2.2.2-1: Non-blocking access to resource in synchronous mode (Hosting CSE = Receiver CSE), requested operation completed before second request

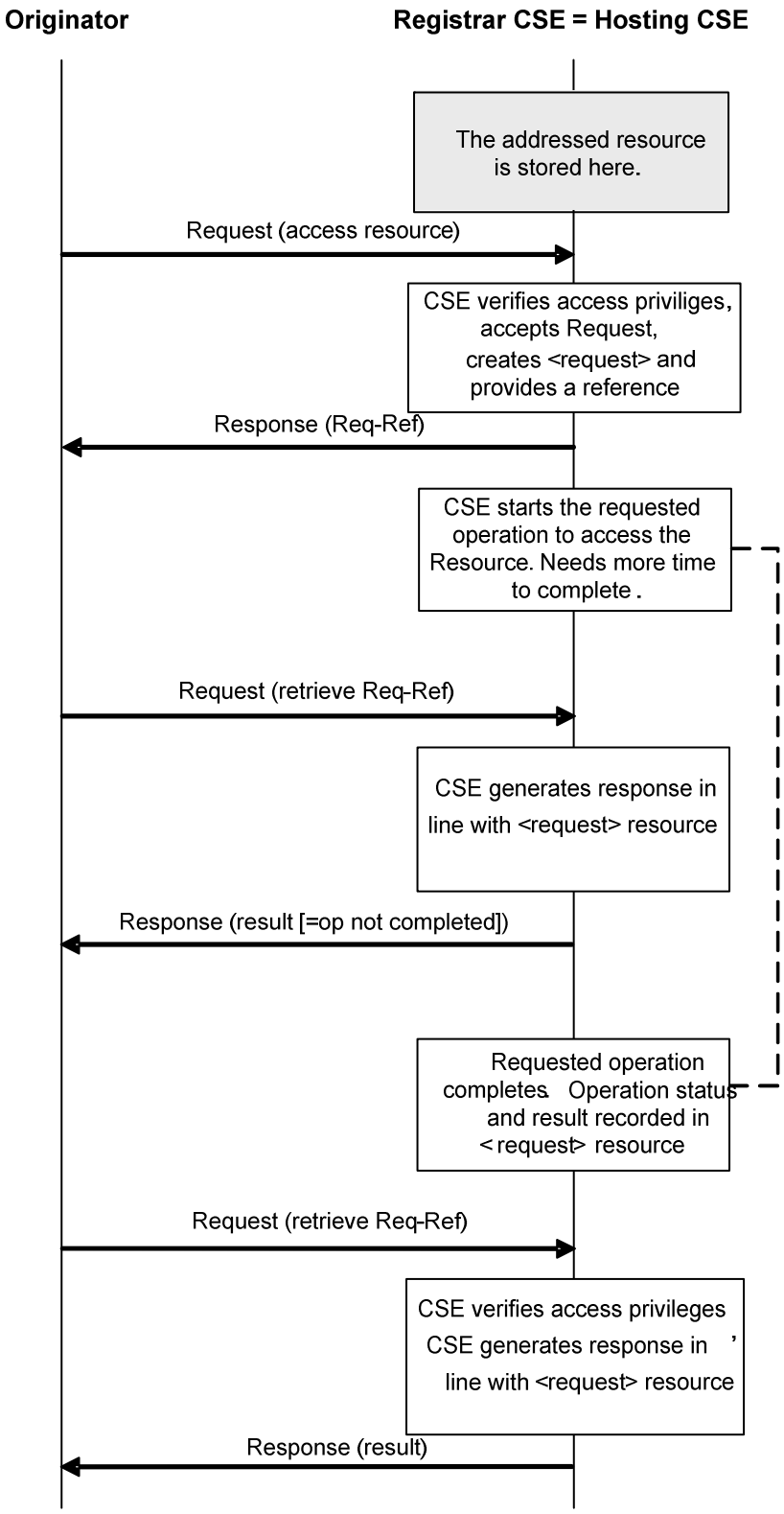


Figure 8.2.2.2-2: Non-blocking access to resource in synchronous mode (Hosting CSE = Receiver CSE), requested operation completed after the second but before the third request

8.2.2.3 Asynchronous Case

In the asynchronous case, it is assumed that the Originator or other entities that need to know about the outcome of a Request are able to receive notification messages, i.e. the CSE carrying out the requested operation may send an unsolicited message to the Originator or to other indicated entities at an arbitrary time to send the status and result of the requested operation to one or more Notification Target(s).

If the Receiver CSE selects to send the NOTIFY in non blocking asynchronous mode, then the Hosting CSE shall request NOTIFY with ***Response Type*** parameter indicating non blocking asynchronous operation with empty target list.

In the asynchronous case, a Receiver CSE that does not support the <*request*> resource type shall respond to an acceptable request with a response containing an Acknowledgement without a reference to a resource containing the context of the request.

In the asynchronous case the exemplary information flow depicted in figure 8.2.2.3-1 is applicable. In this case it is assumed that the Originator of the Request provided two Notification Targets. (the Originator and one other Notification Target) to which notification shall be sent when the result of the requested operation is available or when the request failed.

Equivalent information flows are valid also for cases where the target resource of the requested operation is hosted on the Hosting CSE itself. From an Originator's or Notification Target's perspective there is no difference as the later notification of the result of a requested operation would always be an exchange of request/response messages between the CSE carrying out the requested operation and the Notification Targets using reference to the original Request ID.

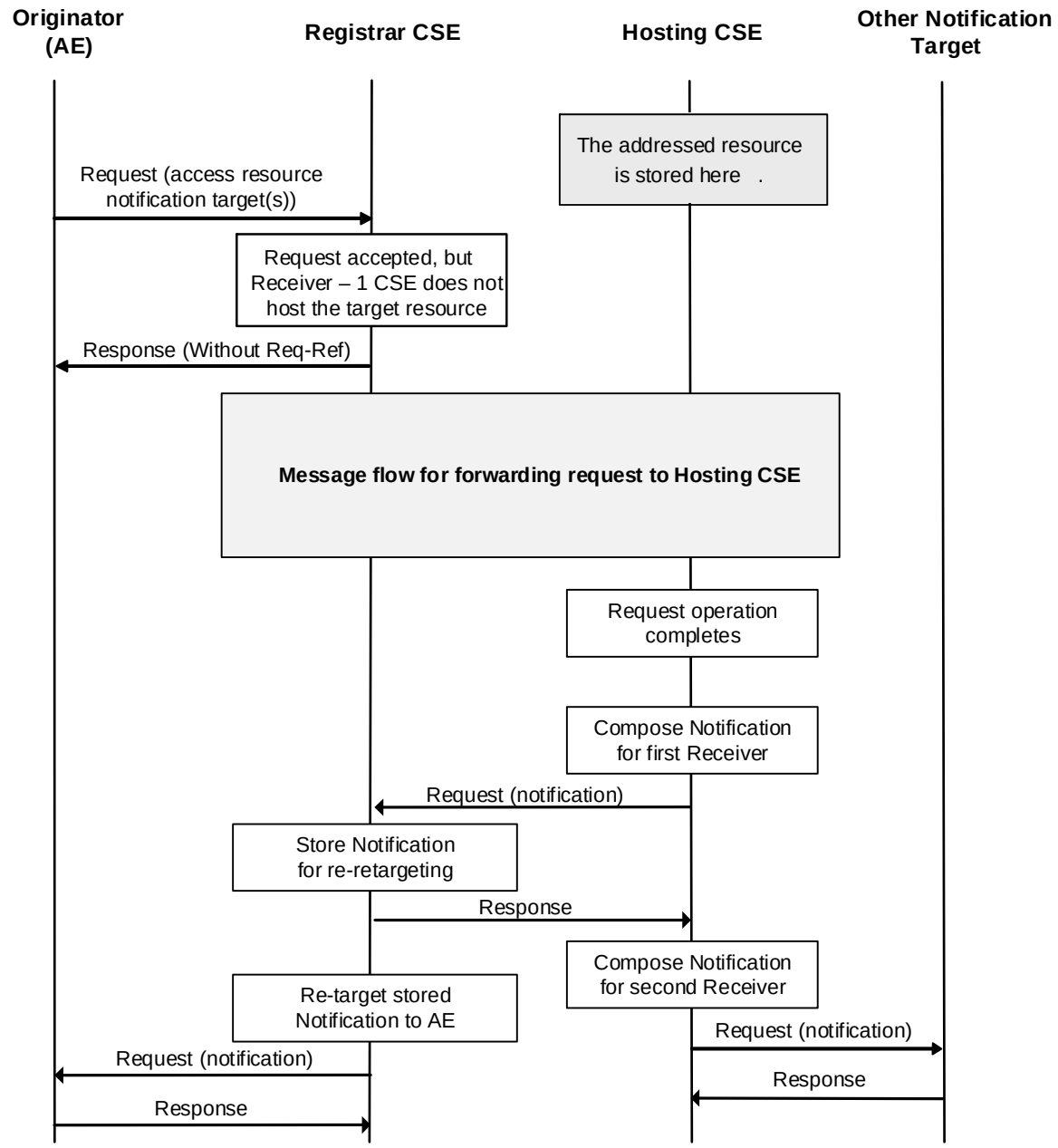


Figure 8.2.2.3-1: Non-blocking access to resource in asynchronous mode (Hosting CSE not equal to Receiver - 1 CSE), Originator provided targets for notification

8.3 Procedures for interaction with Underlying Networks

8.3.1 Introduction

Procedures for interaction with Underlying Networks are used to provide information about the M2M service layer (e.g. communication patterns of oneM2M devices) to the Underlying Network or receive information from the Underlying Network (e.g. reports on issues of the Underlying Network).

Such information enables the Underlying Network to provide means for optimization of M2M traffic and also allows M2M service layer to optimize its services.

8.3.2 Description and Flows on Mcn Reference Point

Communications between the CSEs and the NSEs across the Mcn reference point include:

- the CSE(s) accessing network service functions provided by Underlying Networks; and
- optimizing network service processing for Underlying Networks.

Such services normally are more than just the general transport services.

Communications which pass over the Mcn reference point to Underlying Networks include:

- Messaging services that are widely deployed by Applications and network operators using a number of existing mechanisms.
- Network APIs defined by other SDOs (e.g. OMA and GSMA) are used by network operators for their services.
- Interworking for services and security aspects for MTC (Machine Type Communications) has been defined by 3GPP and 3GPP2.

Examples of service requests from a CSE towards the Underlying Networks are:

- Connection requests with/without QoS requirements.
- Payments, messages, location, bearer information, call control and other network capabilities (e.g. by using GSMA oneAPI, network APIs supporting protocols defined by other SDOs, or proprietary network APIs).
- Device triggering.
- Device management.
- Management information exchange such as charging/accounting records, monitoring and management data exchange.
- Location request.

8.3.3 Device Triggering

8.3.3.1 Definition and scope

Device Triggering is a means by which a node in the infrastructure domain (e.g. IN-CSE) sends information to a node in the field domain (e.g. ASN/MN-CSE or ADN-AE) to perform a specific task, e.g. to wake up the device, to establish communication from the field domain towards the infrastructure domain, or when the IP address for the device is not available or reachable by the infrastructure domain. Triggers are only addressed to and received by ASN/MN-CSEs and ADN-AEs. Triggers may be used to request an ASN/MN-CSE or ADN-AE take some action such as enrol, or as refresh its PoA, or, register, or to request an ADN-AE or a registree AE of the ASN/MN-CSE to perform a CRUD operation.

Underlying Network functionality is used to perform device triggering, for example, using alternate means of communication (e.g. SMS) with the Field Node.

NOTE: Device Triggering is applicable for the entities which are registered with IN-CSE.

Each Underlying Network type may provide a different way of performing a device triggering. For example, 3GPP and 3GPP2 have defined dedicated interfaces for requesting device triggering. The normative references for applicable interfaces are as follows: ETSI TS 123 682 [i.14] and 3GPP2 X.P0068 [i.17]. Access specific mechanisms are covered in ETSI TS 118 126 [15] and annex C.

8.3.3.2 General Procedure for Device Triggering

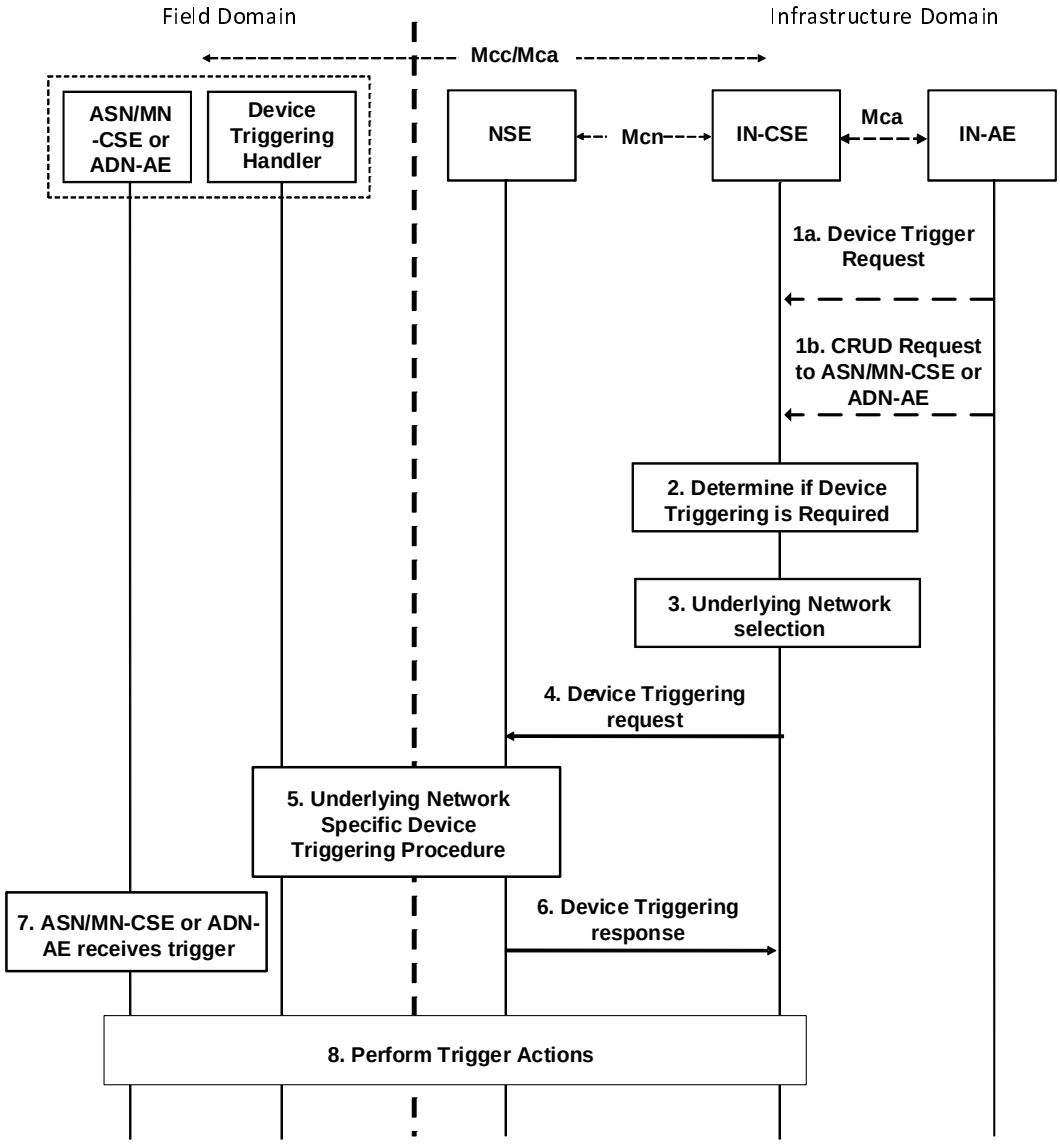
8.3.3.2.0 Overview

This clause covers different scenarios for device triggering.

8.3.3.2.1 Triggering procedure

This case describes the scenario where IN-CSE targets an ASN/MN-CSE or ADN-AE for the Device Triggering request.

Figure 8.3.3.2.1-1 shows the general procedure for Device Triggering and, if required, for establishment of connectivity between an IN-CSE and the Field Node.



NOTE 1: The IN and M2M Device are assumed to be connected through the same Underlying Network.
NOTE 2: The Device Triggering Handler is a functional entity that receives the device triggering request, and it is dependent on the Underlying Network. The Device Triggering Handler is out of scope of the present document.

Figure 8.3.3.2.1-1: General Device Triggering general Procedure

Pre-condition

The ASN/MN-CSE or ADN-AE which is the target of the device triggering may be registered with the IN-CSE, or the IN-CSE may be provisioned with the information necessary to send a trigger to the ASN/MN-CSE or ADN-AE, or an AE can provide the necessary information to the IN-CSE via an AE device trigger request.

Step-1 (Optional): Request to the targeted ASN/MN-CSE or ADN-AE

An AE may issue a device trigger request to an ASN/MN-CSE or ADN-AE by creating or updating a <*triggerRequest*> resource hosted on an IN-CSE. Alternatively, an IN-CSE may initiate a device trigger request to an ASN/MN-CSE or ADN-AE. For example, if an IN-CSE receives an AE request to perform a CRUD operation targeting an ASN/MN-CSE or ADN-AE that is not reachable by the IN-CSE, the IN-CSE may generate a trigger request.

Step-2: Determine if Device Triggering is required

The IN-CSE determines whether or not to send a device trigger to the targeted ASN/MN-CSE or ADN-AE by performing the following ASN/MN-CSE or ADN-AE registration and trigger enable checks.

- If the ASN/MN-CSE or ADN-AE is not registered to the IN-CSE and the purpose of the trigger is to have the ASN/MN-CSE or ADN-AE register to the IN-CSE or enrol to a MEF, then the IN-CSE sends a trigger request without checking whether trigger functionality is enabled for the ASN/MN-CSE or ADN-AE. If the ASN/MN-CSE or ADN-AE is already registered and the purpose of the trigger is to have the ASN/MN-CSE or ADN-AE register to the IN-CSE or enrol to a MEF, then the IN-CSE shall not send a trigger request.

- If the purpose of the trigger is to have the ASN/MN-CSE or ADN-AE establish a connection, update its PoA, or perform a CRUD operation, then the IN-CSE shall first checks whether the ASN/MN-CSE or ADN-AE is registered to the IN-CSE. If the ASN/MN-CSE or ADN-AE is not registered, then the IN-CSE shall not perform the trigger. If registered, the IN-CSE shall check whether the *triggerEnable* attribute of the corresponding ASN/MN-CSE's <*remoteCSE*> or ADN-AE's <*AE*> resource is "TRUE". If *triggerEnable* is "TRUE" the IN-CSE shall send a trigger request to the ASN/MN-CSE or ADN-AE.

Step-3: Underlying network selection

The IN-CSE selects the Underlying Network and the mechanism to deliver the triggering request to the Underlying Network according to the configuration for connected Underlying Networks.

For example, for 3GPP access network IN-CSE may use Tsp, Tsms and GSMA OneAPI; and for 3GPP2 access networks IN-CSE may use Tsp and SMS. However the preferred mechanism is Tsp.

Step-4: Device Triggering request

IN-CSE issues the device triggering request to the selected Underlying Network.

NOTE 1: The Underlying Network dependent Device Triggering procedure for 3GPP and 3GPP2 systems are described in ETSI TS 118 126 [15] and annex C respectively.

Some information provided to the selected Underlying Network for performing device triggering includes:

- M2M-Ext-ID associated with the target ASN/MN-CSE or ADN-AE of the triggering request (see clause 7.1.8).

- Trigger-Recipient-ID associated with the target ASN/MN-CSE or ADN-AE (see clause 7.1.10). For example, when 3GPP Underlying Network is used this identifier could map to Application-Port-ID. If there are multiple ADN-AEs on the node, then they shall use different Trigger-Recipient-IDs.

- IN-CSE ID which could be used by the Underlying Network to authorize the IN-CSE for device triggering.

- Optional Trigger Payload which includes a *triggerPurpose*, and additional payload fields:

  - The *triggerPurpose* field may take the following values:

    - establishConnection - The ASN/MN-CSE or ADN-AE shall interpret this as a request to establish a connection and, if the address of the <remoteCSE> (*triggerInfo Address)* is present in the payload, refresh its PoA.

    - enrolmentRequest - The ASN/MN-CSE or ADN-AE shall interpret this as a request to enroll with a MEF.

    - registrationRequest - The ASN/MN-CSE or ADN-AE shall interpret this as a request to register with a MN/IN-CSE.

  - executeCRUD - The ASN/MN-CSE or ADN-AE shall interpret this as a request to execute a particular CRUD operation. When the trigger recipient is an ASN/MN-CSE, the payload shall indicate which Registree AE of the ASN/MN-CSE is being asked to perform the CRUD operation. The MN/ASN-CSE checks the <AE> resource corresponding to the AE-ID of the ASN/MN-AE that was provided in the trigger payload. It checks if there is a <subscription> to the <AE> resource. It then checks if the eventType attribute of the <subscription> resource indicated that the subscription is for a trigger. If yes, then the MN/ASN-CSE creates the notification and includes the trigger payload in the content of the notification. The notification is sent to the AE and the AE creates a new CRUD request to the IN-CSE as a result of the trigger. It is assumed that an AE, who is targeted with this type of trigger has subscribed to its <AE> resource and is provisioned to know how to interpret the payload content. The CRUD operation shall be performed by the Registree AE as governed by rules and constrains detailed in note 5. When the trigger payload indicates that an ASN/MN-AE is being asked to perform a CRUD operation, its Registrar SN/MN-CSE may establish connectivity with the IN-CSE immediately or it may postpone establishing connectivity with the IN-CSE until the ASN/MN-AE initiates a CRUD request.

- Depending on the *triggerPurpose* field, the rest of the trigger payload may contain:
  - When the *triggerPurpose* field is set to "establishConnection", the payload contains the resource address of the <*remoteCSE*> or <*AE*> where the PoA needs to be updated (*triggerInfoAddress)*. If *triggerInfoAddress* is not provided, the ASN/MN-CSE or ADN-AE assumes that the PoA on its Registrar CSE does not need to be updated.
  - When the *triggerPurpose* field is set to "enrolmentRequest", the payload contains the resource address (*triggerInfoAddress)* of the <*MEFBase*> that the ASN/MN-CSE or ADN-AE should enroll to, the supported protocol bindings that may be used when contacting the <*MEFBase*> and the port number that should be used for each binding.
  - When the *triggerPurpose* field is set to "registrationRequest", the payload contains the resource address (*triggerInfoAddress)* of the <*cseBase*> that the ASN/MN-CSE or ADN-AE should register to, the supported protocol bindings that may be used when contacting the <*cseBase*> and the port number that should be used for each binding.
  - When the *triggerPurpose* field is set to "executeCRUD", the payload provides: the type of CRUDN operation (*triggerInfoOperation*), the address of the resource that the operation should be performed on (*triggerInfoAddress)* and the resource type (*targetedResourceType*). If the trigger recipient is an ASN/MN-CSE, the trigger payload also provides the identity of the ASN/MN-AE that is to perform the CRUD operation (*triggerInfoAeId*).

NOTE 2: The M2M-Ext-ID may be pre-provisioned at the IN-CSE along with the associated CSE-ID or AE-ID, or may be sent at registration, or provided to the IN-CSE by an AE via a trigger request (see clause 7.1.8).

NOTE 3: The above Trigger-Recipient-ID may be pre-provisioned at the IN-CSE along with the associated M2M-Ext-ID, or may be sent at registration, or provided to the IN-CSE by an AE via a trigger request (see clause 7.1.10).

NOTE 4: It is left to Stage 3 to develop the bit encoding for the *triggerPurpose* and *the rest of the payload* fields.

NOTE 5: The following defaults will be used by the trigger recipient to construct the operation requested via executeCRUD.

NOTE 6: The trigger payload sent in the Trigger request should be serialized based on the contentSerialization attribute of the <AE> or <remoteCSE> resource of the targeted entity.

- All triggered CRUD operations are non-blocking, with nonBlockingRequestSynch responses.
- ***Operation, To, Resource type*** *(if mandatory):* set as directed by *the triggerInfoOperation, triggerInfoAddress, and targetedResourceTypes* fields in the trigger payload.
- ***Event Category*** - set to *"immediate"*.
- ***Delivery Aggregation*** - set to *"aggregation off"*.
- ***From, Request Identifier, Originating timestamp Request Expiration, Result Expiration, Operational Execution Time, Result Persistence*** - set as per existing local policies.

- ***All other parameters are Not Present (NP).***

Step-5: Underlying Network Specific Device Triggering procedure

Device Triggering processing procedure is performed between the Underlying Network and the target Node.

Step-6: Device Triggering response

The IN-CSE receives a response for the Device Triggering request via the Mcn reference point.

Step-7: ASN/MN-CSE or ADN-AE Receives Device Trigger

If the trigger had no optional trigger payload, the ASN/MN-CSE or ADN-AE assumes that the purpose of the trigger is to cause the ASN/MN-CSE or ADN-AE to establish connectivity with the IN-CSE. In this case, the address of the IN-CSE is already known to the ASN/MN-CSE or ADN-AE.

If the trigger has an optional trigger payload, the ASN/MN-CSE or ADN-AE uses the triggerPurpose to determine the appropriate action and perform the necessary steps.

Step-8: Perform Trigger Actions

Based on the type of trigger request received, the ASN/MN-CSE or ADN-AE performs the corresponding trigger actions such as establish connectivity with the IN-CSE, enrol with the MEF, register to the IN-CSE, update its PoA, or execute a CRUD request on a specified resource.

8.3.3.2.2 Support for device trigger recall/replace procedure

Figure 8.3.3.2.2-1 shows a procedure for device triggering recall (i.e. cancel a trigger request) and /replace (i.e. update a trigger request) between oneM2M and an Underlying Network.

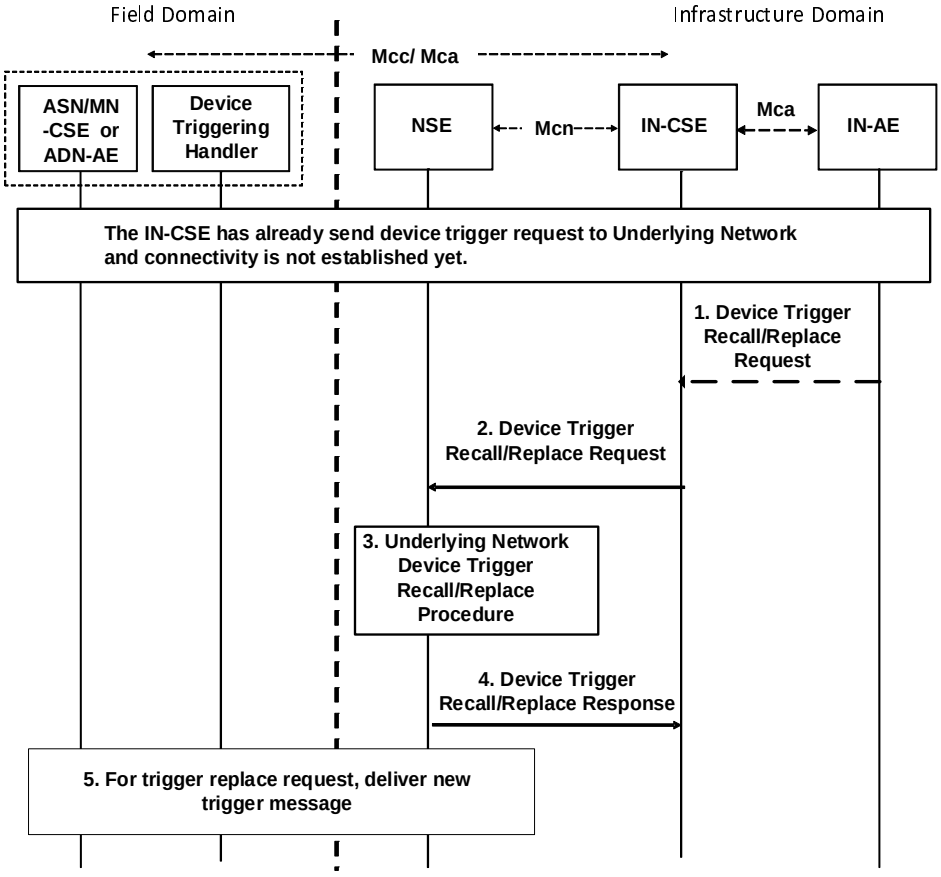


Figure 8.3.3.2.2-1: Device triggering recall/replace procedure

Pre-condition

The IN-CSE has already sent device trigger request to Underlying Network (e.g. 3GPP) and connectivity is not established yet. IN-CSE has already stored the previous device trigger information, e.g. trigger reference number, etc.

Step-1: (Optional): AE Trigger Recall/Replace Request

The AE issues a request to IN-CSE to recall/replace trigger that results in the IN-CSE generating a trigger recall/replace request to the Underlying Network.

Alternatively, the IN-CSE may decide to recall/replace a trigger that the IN-CSE previously initiated. This may be done based on internal policies

Step-2: Device Trigger Recall/Replace request

IN-CSE issues the device trigger Recall/Replace request to the Underlying Network.

In addition to same parameters in the original device trigger request, the following additional parameters for device trigger recall/replace include:

- The old trigger reference number was assigned to the previously submitted trigger message that the IN-CSE wants to recall/replace.
- For trigger replace request, the new trigger reference number which is assigned by the IN-CSE to the newly submitted trigger message.

Step-3: Network Device Trigger Recall/Replace procedure

Device Trigger Recall/Replace procedure is performed in Underlying Network.

Step-4: Device Trigger Recall/Replace response

The IN-CSE receives a response for the Device Trigger Recall/Replace request via the Mcn reference point.

If the IN-CSE receives a success response, the IN-CSE updates the device trigger information as following:

- For device trigger replace success response, the IN-CSE shall store the new trigger reference number replace the old trigger reference number.
- For device trigger recall success response, the IN-CSE shall clear the old trigger reference number.

Step-5: For trigger replace request, deliver new trigger message.

For trigger replace request, the new trigger message will be delivered to the target Node.

8.3.4 Location Request

8.3.4.1 Definition and Scope

Location Request is a means by which a CSE requests the geographical or physical location information of a target CSE or AE hosted in a M2M Node to the location server located in the Underlying Network over Mcn reference point. This clause describes only the case of location request when the attribute *locationSource* is set to Network Based.

8.3.4.2 General Procedure for Location Request

This procedure describes a scenario wherein an AE sends a request to obtain the location information of a target AE or CSE hosted in an M2M Node to the location server NSE, and the location server responses to the CSE with location information.

Figure 8.3.4.2-1 shows the general procedure for Location Request.

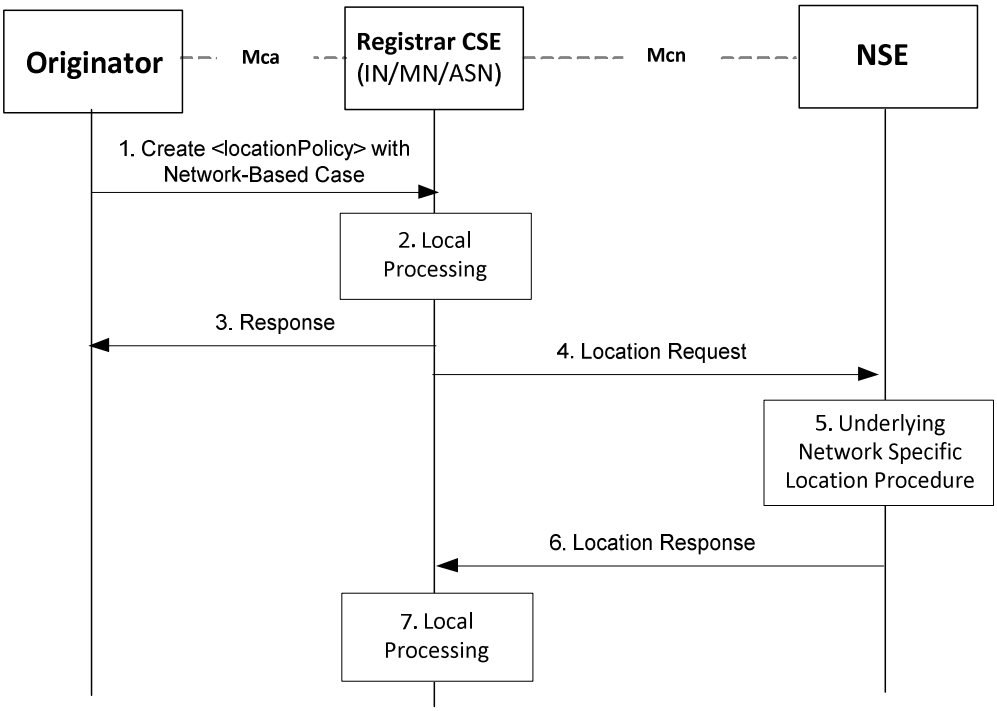


Figure 8.3.4.2-1: General Procedure for Location Request

NOTE 1: Detailed descriptions for step-1 to the step-3 are described in the clause 10.2.11.1.

Step-1: Create <*locationPolicy*>

The Originator requests to CREATE <*locationPolicy*> resource at the Registrar CSE. The *locationSource* attribute of the <*locationPolicy*> resource shall be set to 'Network-Based' and the value for *locationTargetID* and *locationServer* attributes shall be set properly set for the Location Request.

Step-2: Local Processing for creating <*locationPolicy*> resource

After verifying the privileges and the given attributes, the Hosting CSE creates <container> resource where the actual location information is/are stored. Then the Hosting CSE shall create <locationPolicy> resource. The Hosting CSE shall maintain cross-reference between both resources: locationContainerID attribute for <locationPolicy> resource and locationID attribute for <container> resource.

Step-3: Response for creating <*locationPolicy*>

The Registrar CSE shall respond with a Response message.

Step-4: Location Request

The Registrar CSE issues Location Request to the selected Underlying Network. For doing this, the Registrar CSE shall transform the location configuration information received from the Originator into Location Request that is acceptable for the Underlying Network. For example, the Location Request can be one of existing location acquisition protocols such as OMA Mobile Location Protocol [i.5] or OMA RESTful NetAPI for Terminal Location [i.6]. Additionally, the Registrar CSE shall provide default values for other parameters (e.g. required quality of position) in the Location Request according to local policies.

NOTE 2: The Location Request can be triggered by the given conditions, e.g.:

1) when the *locationUpdatePeriod* attribute has expired, or if the *locationUpdatePeriod* attribute is not given from the step-1;

2) the <*locationPolicy*> is created or updated;

3) the linked <*container*> has been retrieved.

4) if the attribute *locationUpdatePeriod* has multiple value and the Hosting CSE of the resource is the target device, the Hosting CSE of the resource may update the location update period by choosing one of the value within the list according to the local context information of the device (velocity, battery level, current range) and its preprovisioned local policy which is out of scope of the present document. The Hosting CSE then issues Location Request with selected value as the update period. Then, if the value switches to another value, step-4,5,6,7shall be repeated using the new period.

Step-5: Performing Location Procedure

The Underlying Network specific procedures are performed. This may involve getting location information from the target device or the network node. These procedures are outside the scope of oneM2M specifications.

Step-6: Location Response

The NSE responds to the Registrar CSE with location information if the Registrar CSE is authorized. If not, the NSE sends an error code back to the Registrar CSE.

Step-7: Local Processing after Location Response

The received response shall be contained in the <container> resource that is related the <*locationPolicy*> resource.

NOTE 3: Please see the clause 10.2.11.2 for detail information.

NOTE 4: For notification regarding the location response towards the Originator, the subscription mechanism is used.

8.3.5 Configuration of Traffic Patterns

8.3.5.1 Purpose of Configuration of Traffic Patterns

M2M devices that have predicable communication behaviour - e.g. in the form of repeating traffic patterns - can profit in terms of reduction of signalling, energy saving, fewer sleep/wake transitions, etc., when their traffic patterns are communicated to the underlying network.

For example, 3GPP devices could use new 3GPP power savings features such as eDRX (extended discontinuous reception) and PSM (Power Saving Mode) on LTE devices.

Also the underlying network can benefit from being informed about a device's traffic patterns by the oneM2M System.

For example, if the IN-CSE knows the device's traffic patterns and transmits them to an underlying 3GPP network, then this information can be used by a 3GPP network to set the device's "Maximum Response Time" (3GPP Term) to tune the UE's DRX and PSM parameters.

Thus the network will benefit because the UE will have fewer sleep/wake transitions and unnecessary signalling in the network can be avoided. Also, if the IN-CSE knows when the device is awake then data can be sent to the device exactly at the time when the device is listening, thus requiring the network to buffer less data for unavailable devices.

The purpose of the Configuration of Traffic Patterns feature is to provide a means to the oneM2M System to provide the Underlying Network optimization information.

The Common Service Entity (CSE) shall use the Mcn interface towards the Underlying Network to provide information on the traffic patterns of a Field Domain Node (ASN or MN) to the underlying network.

To that purpose in the oneM2M System:

- Field Domain Nodes are addressed using the CSE-ID or AE-ID resource identifiers of the corresponding <*AE*> or <*remoteCSE*> resources of the Field Domain Node.
- A group of Field Domain Nodes can be addressed in the oneM2M System by the resource identifier of a corresponding <group> resource.

In the case of a 3GPP network, Field Domain Nodes are identified by the CSE towards the Underlying Network using the M2M-Ext-ID or External Group Identifier.

8.3.5.2 Traffic pattern parameters

Traffic Pattern (TP) parameters can be associated with one or multiple Field Domain Nodes and are defined in table 8.3.5.2-1.

For each Underlying Network, a Field Domain Node can be associated with one or more TP parameters sets that have non-overlapping schedules.

Each parameter set is derived by the CSE from information provided for AEs and CSEs, respectively, using information provided in one item of the *activityPatternElements* attribute (see table 9.6.4-3). Therefore, a set can be derived when the list in the *activityPatternElements* attribute has more than one item. The parameter derivation is described and exemplified in table 8.3.5.2-1.

Table 8.3.5.2-1: Traffic parameter set

TP parameter set	Description	Derivation from *activityPatternElements*
TP Periodic communication indicator	Identifies whether the Node communicates periodically or not, e.g. only on demand.	If periodicity can be derived from the *scheduleElement* of the *activityPatternElements*, the indicator shall be set to TRUE. Otherwise, shall be set to FALSE.
TP Communication duration time	Duration interval time of periodic communication (may be used together with TP Periodic communication indicator). EXAMPLE: 5 minutes.	To be derived from the *scheduleElement* of the *activityPatternElements* as follows: - If a finite communication duration time can be derived, the derived value shall be used. - If only a start time is provided, a maximum value according to set defaults shall be used. - If no start time is provided, the value 0 shall be set.
TP Time period	Interval Time of periodic communication (may be used together with TP Periodic communication indicator). EXAMPLE: Every hour.	If periodicity can be derived from the *scheduleElement* of the *activityPatternElements*, the derived periodicity value shall be set.
TP Scheduled communication time	Time and Day of the week when the Node is available for communication. EXAMPLE: Time: 13:00, Day: Monday.	The start time derived from the current time and the *scheduleElement* of the *activityPatternElements* shall be set.
TP Stationary indication	Identifies whether the Node is stationary or mobile.	The *stationaryIndication* provided in the *activityPatternElements* shall be used. If no *stationaryIndication* is provided this optional parameter is not set.
TP Data size indication	Indicates the expected data size for the pattern.	The value of the *dataSizeIndicator* provided in the *activityPatternElements* shall be used. If no *dataSizeIndicator* is provided this optional parameter is not set.
TP Validity time	The time after which a TP parameter becomes invalid once it had been set.	If an end time can be derived from the *scheduleElement* of the *activityPatternElements,* the end time value shall be used. If no end time can be derived a maximum value according to set defaults shall be used.

EXAMPLE: Consider an evaluation of an *activityPatternElements* attribute as follows:

scheduleElement (with the fields: second, minute, hour, day of month, month, day of week and year) *; 0-30 ; 2; *; Jan-Sept; Tues; 2017

*stationaryIndication: "*Moving"

dataSizeIndicator: 30 kb

The following TP set shall be derived:

- TP Periodic communication indicator: TRUE
- TP Communication duration time: 30 min
- TP Time period: 1 week
- TP Scheduled communication time: Tues, 2:00
- TP Stationary indication: "Moving"
- TP Data size indication: 30 kb
- TP Validity time: 2 months (default maximum)

Note that the IN-CSE may use a single set of TP parameters for an entire group of Field Nodes if the corresponding *activityPatternElements* attributes are identical or if the IN-CSE can derive a common pattern for the group, corresponding to a common *activityPatternElements* attribute. How the parameters corresponding to a common *activityPatternElements* attribute are derived by the IN-CSE is implementation dependent, e.g. by computing the time superset. The parameters of this common *activityPatternElements* attribute are then used as described above to derive a single TP set for the group.

8.3.5.3 General procedure for Configuration of Traffic Patterns

Figure 8.3.5.3-1 depicts a general procedure for configuration of Traffic Patterns.

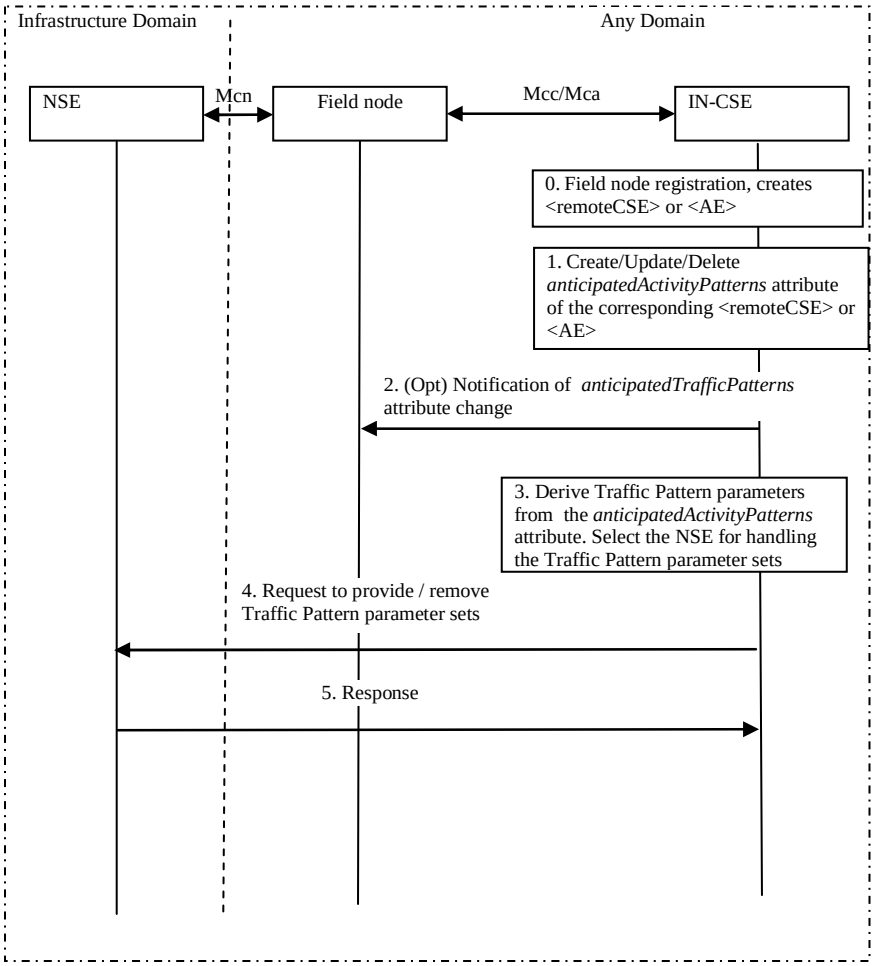


Figure 8.3.5.3-1: General procedure for configuration of Traffic Patterns

Step-0: Field Node registration with IN-CSE

The field node (ADN-AE or ASN/MN-CSE) registers with the IN-CSE. The respective <*AE*> and <*remoteCSE*> resources are created and linked to the corresponding <*node*> resource.

If the IN-CSE uses a single set of TP parameters for an entire group of Field Nodes, it is assumed that they are managed together using a <group> resource and that they are identified in the Underlying Network by a common External Group Identifier. The IN-CSE shall verify that the <*group*> resource membership consists solely of <*AE*> or <*remoteCSE*> resources.

Step-1: Anticipated Communication behaviour of the ADN-AE or ASN/MN-CSE is changed

The anticipated communication behaviour of the ADN-AE or ASN/MN-CSE is changed by updating the *activityPatternElements* attribute of either the <*AE*> or <*remoteCSE*> resource, respectively.

In the group case the anticipated communication behaviour of the group members is changed by updating the *activityPatternElements* attribute using a request targeting the <*fanoutPoint*> virtual resource.

Step-2: (Optional) IN-CSE notifies the Field Node that the communication behaviour has changed

Optionally, the IN-CSE notifies the ADN-AE or ASN/MN-CSE that the anticipated communication schedule has been changed.

Step-3: The IN-CSE derives the TP parameters and the NSE for handling the TP parameter sets

If the IN-CSE selects the NSE by using the network identifier of the Field Domain Node (i.e. the M2M-Ext-ID or External Group Identifier) by which the Field Node can be identified in the NSE (see clause 7.1.8).

The IN-CSE derives the TP parameters as follows:

- For a Field Node hosting one or more AEs represented with a single <node> resource, using the values provided in all the *activityPatternElements* attribute for the <AE>s on this node.
- For a Field Node hosting an ASN or MN, using the values provided by the *activityPatternElements* attribute of the <*remoteCSE*> resource.
- For a group of Field Nodes, using the values provided by the activityPatternElements attribute of each <group> member.

Step-4: Request for the handling of the TP parameter sets

IN-CSE sends a request for handling (i.e. provide or remove) TP parameter sets for the Field Domain Node to the NSE, using the appropriate Mcn protocol. The Mcn can correspond to one of the standard interfaces specified by an external organization, for example, OMA RESTful Network API for Communication Patterns V1.0 [i.31].

The request shall include the corresponding M2M-EXT-ID of a field Node or the External Group Identifier of a group of Field Nodes, and one or more TP parameter set(s) as defined at clause 8.3.5.2.

NOTE 1: If the Underlying Network is 3GPP-compliant, see ETSI TS 118 126 [15] for more details.

Step-5: Response for the handling of the TP parameter sets

The IN-CSE receives the response for the configuration of the TP parameter sets from the NSE.

NOTE 2: If the interaction with NSE in step 3 is unsuccessful, the IN-CSE has the choice to re-try it until successful.

Step-6: The Field Node TP changes are applied

After the notification in step 2, the Field Node (ASN/MN-CSE or ADN-AE) shall utilize the latest values provided by the *activityPatternElements* attribute.

8.4 Connection Request

Connection request service is not defined in the present document.

8.5 Device Management

See clause 6.2.4 for a detailed description on the interaction with a Device Management Server.

9 Resource Management

9.0 Overview

All entities in the oneM2M System, such as AEs, CSEs, data, etc. are represented as resources. A resource structure is specified as a representation of such resources. Such resources are uniquely addressable. Procedures for accessing such resources are also specified.

9.1 General Principles

The following are the general principles for the design of the resource model:

- The "type" of each resource shall be specified. New resource types shall be supported as the need for them is identified.
- The root of the resource structure in a CSE shall be assigned an absolute address. See clause 9.3.1 for additional information.
- The attributes for all resource type shall be specified.
- Each resource type may be instantiated as multiple resources via Create procedure (clause 10.1.2).
- All resources and associated attributes shall be addressable as specified in clause 9.3.1.
- Both hierarchical and non-hierarchical URIs shall be supported by all CSEs.

9.2 Resources

9.2.0 Overview

This clause introduces the resources used in a CSE. A resource scheme is used for modelling the resource structure and associated relationships. Clause 9.5 provides guidelines on how to describe a resource. The present document identifies three categories of resources:

- Normal resources (clause 9.2.1).
- Virtual resources (clause 9.2.2).
- Announced resources (clause 9.2.3).

9.2.1 Normal Resources

Normal resources include the complete set of representations of data which constitutes the base of the information to be managed.

Unless qualified as either "virtual" or "announced", the resource types in the present document are normal resources.

9.2.2 Virtual Resources

A virtual resource is used to trigger processing and/or retrieve results, but they do not have a permanent representation in a CSE.

9.2.3 Announced Resources

An announced resource contains a set of attributes of the original resource. An announced resource is updated automatically by the Hosting CSE of the original resource whenever the original resource changes. The announced resource contains a link to the original resource.

Resource announcement can facilitate resource discovery. The announced resource at a remote CSE can also be used for creating child resources at the remote CSE that are not present as children of the original resource or are not announced children of the original resource.

The following are the resource specification guidelines for resource announcement:

- In order to support announcement of resources, an additional column in the resource template (clause 9.5.1), shall specify the attributes to be announced for inclusion in the associated announced resource type.

- For each announced <*resourceType*>, the addition of suffix "Annc" to the original <*resourceType*> shall be used to indicate its associated announced resource type. For example, resource <*containerAnnc*> shall indicate the announced resource type for <*container*> resource; <*groupAnnc*> shall indicate announced resource type for <*group*> resource, etc.

9.3 Resource Addressing

9.3.1 Generic Principles

An identifier of a resource is a string of characters used to uniquely identify the targeted resource within the scope of a request to access the resources. The scope of a request can be:

- CSE-relative: The request is targeting a resource that resides on the same CSE as the Receiver CSE of the request. In that case a CSE-relative format of a resource identifier can be used to address the resource.
- SP-relative: The request is targeting a resource that resides on a CSE within the same M2M SP domain as the Originator of the request. In that case an SP-relative format of a resource identifier can be used to address the resource.
- Absolute: The request is targeting a resource that resides on a CSE that is within an M2M SP domain that is different from the M2M SP domain of the Originator of the request. In that case the absolute format of a resource identifier shall be used to address the resource. Note that the absolute format of the resource identifier will always be acceptable also in other cases.

A single resource may have more than one resource identifier formats depending on the method and scope that are summarized in table 9.3.1-1.

There are two different methods for identifying a resource within the oneM2M resource structure with three different variants each depending on the scope of the request to access the resource. The ways how the resource identifiers are constructed in each case shall follow.

Table 9.3.1-1 Resource addressing methods

Method	Request Scope		
	CSE-Relative	**SP-Relative**	**Absolute**
Unstructured	Use the 'Unstructured-CSE-relative-Resource-ID' format of the resource identifier as defined in table 7.2-1.	Use the 'SP-relative-Resource-ID' format of the resource identifier constructed with the 'Unstructured-CSE-relative-Resource-ID' as defined in table 7.2-1.	Use the 'Absolute-Resource-ID' format of the resource identifier constructed with the 'Unstructured-CSE-relative-Resource-ID' as defined in table 7.2-1.
Structured	Use the 'Structured-CSE-relative-Resource-ID' format of the resource identifier as defined in table 7.2-1.	Use the 'SP-relative Resource-ID' format of the resource identifier constructed with the 'Structured-CSE-relative-Resource-ID' as defined in table 7.2-1.	Use the 'Absolute-Resource-ID' format of the resource identifier constructed with the 'Structured-CSE-relative-Resource-ID' as defined in table 7.2-1.

These two methods with three request scope variants shall all be supported by a CSE receiving requests.

9.3.2 Addressing an Application Entity

9.3.2.1 Application Entity Addressing

In M2M communication, the goal of M2M addressing is to reach the CSE with which the target AE is registered, and ultimately the target AE on the M2M Node on which the target AE is resident. This principle applies to all Application Entities.

Reachability and routing from/to AEs on M2M Nodes is associated with the CSEs with which these AEs are registered, and the connectivity of such CSEs to the Underlying Networks. Reaching an AE shall be performed through reaching the CSE the AE is registered with. A CSE-PoA (CSE Point of Access) shall provide the set of information needed to reach a CSE from an Underlying Network perspective. Typically a CSE-PoA contains information that is resolved into a network address.

9.3.2.2 Application Entity Reachability

9.3.2.2.1 CSE Point of Access (CSE-PoA)

The CSE-PoA shall be used by the M2M System to communicate with a CSE on an M2M Node. Once communication with a CSE is achieved, an AE registered with that CSE can be reached as long as the AE can be uniquely identified.

The information included in the CSE-PoA as well as the refresh of the CSE-PoA, depends on the characteristics of the Underlying Network and an M2M Node's transport capabilities.

9.3.2.2.2 Locating Application Entities

Locating an AE is a two-step process as follows:

- **Step 1:** There is a need to locate the CSE where the AE is registered. Locating the CSE shall be accomplished as follows:
  - For AEs associated with ASNs/MNs/INs, the CSE-PoA of the ASN-CSE/MN-CSE/IN-CSE where the AE is registered shall be used.
  - For AEs associated with ADNs, the CSE-PoA of the MN-CSE/IN-CSE where the ADN is registered shall be used.
- **Step 2:** The CSE shall locate the appropriate AE using its Application Entity Identifier (AE-ID).

9.3.2.2.3 Usage of CSE-PoA by the M2M System

9.3.2.2.3.0 Overview

The CSE-PoA holds the information used by the M2M System to locate routing information for a CSE. This information shall be provided by the CSE at registration time. However, the routing information related to a CSE (and ultimately to the target AE) in an M2M System depends on the characteristics of the Underlying Network. This impacts the criteria for updating the CSE-PoA by the registered CSE, in addition to the regular CSE registration updates. The information to be conveyed as CSE-PoA needs to support Underlying Network specifics.

CSE-PoA is considered equivalent to the routable addresses of the targeted CSE.

In general the addressing and routing information related to a CSE can be achieved when a static public IP address is assigned to and M2M Node and direct DNS address translation or dynamic DNS address translation is used.

In those circumstances, the CSE-PoA for a registered CSE shall have a URI conforming to IETF RFC 3986 [18] as follows:

- URI = scheme:/fullyqualifieddomainname/path/; or
- URI = scheme://ip-address/path/.

The following clauses specify the information to be conveyed in the CSE-PoA by a registered CSE for various types of Underlying Networks, as well as the criteria for updating the CSE-PoA for the registered CSEs, in addition to the normal CSE registration refresh.

9.3.2.2.3.1 CSE-PoA related to CSEs associated with a Fixed Network

In this case the CSE-PoA for a registered CSE shall have a URI as described above. If the IP address is private, then the address is usually built based on the address of the related PPP protocol which is a public IP address. This in turn is mapped to the corresponding private address.

9.3.2.2.3.2 CSE-PoA related to CSEs associated with Mobile Networks

If the IP address for the registered CSE cannot be reliably used, and cannot be included in the CSE-PoA, then the CSE-PoA for the registered CSE shall include appropriate information as required by the respective Underlying Networks and supported by oneM2M.

Each Underlying Network shall need to specify the means for allowing an M2M SP to fetch the IP address associated with a CSE attaching to that Underlying Network and consequently the information to be included in the CSE-PoA for the registered CSE.

In the event that the M2M SP has connections to multiple Underlying Networks, there is a need to establish a binding between the registered CSE and the associated Underlying Network. That binding may be established through CSEs explicitly identifying the Underlying Network at registration/update time. Otherwise the M2M SP may derive the identity of the Underlying Network, e.g. by using the link, over which the registration arrived, store it and bind it to the registration information.

In the scenarios an M2M Node in mobile networks is not reachable by the previously known IP address and it supports SMS, the originating CSE can make use of SMS for device triggering mechanism to wake up the M2M Node to renew the IP addresses or perform specific functionalities.

To support this option, the CSE-PoA shall, on Mcn interface to the Underlying Networks supporting such an SMS for device triggering mechanism, include identification information of the CSE (such as the external identifier as defined by ETSI TS 123 682 [i.14] in the case of Tsp-based triggering, or MSISDN or any identifier used by triggering network APIs), and send the request to the Underlying Network via the mechanisms supported, such as Tsp, Tsms, Network APIs.

The 3GPP defined interfaces for machine type communication interfaces and example device triggering flows are shown in ETSI TS 118 126 [15].

9.3.2.2.3.3 CSE-PoA to CSEs associated with multiple Underlying Networks

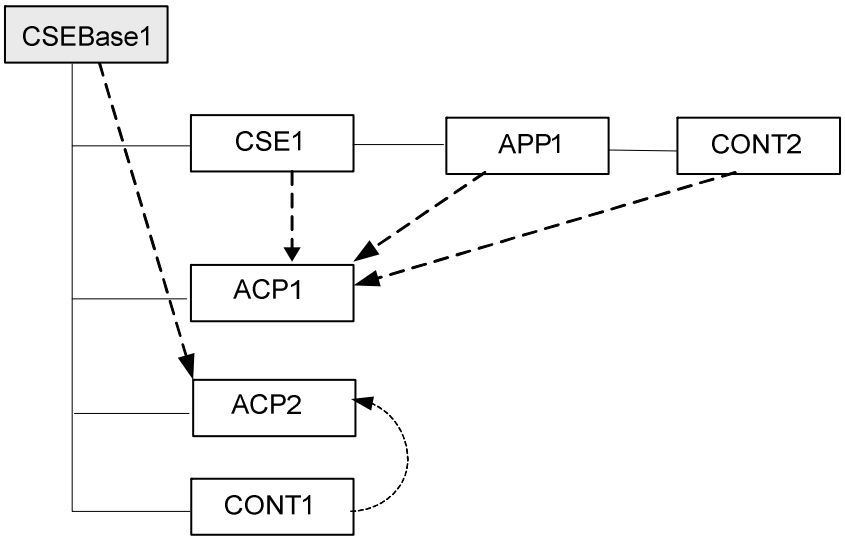
When an M2M Node attaches to a fixed network, the CSE-PoA for a registered CSE shall conform to the procedures associated with the fixed network.

When an M2M Node attaches to a mobile network, the CSE-PoA for a registered CSE shall conform to the procedures associated that mobile network.

If an M2M Node is already attached to an Underlying Network and attaches to another Underlying Network, the CSE may update its PoA information at the remote CSE.

9.4 Resource Structure

9.4.1 Relationships between Resources



NOTE: The resources shown are:
- CSEBase1 is the name of a resource of type <*CSEBase*>.
- CSE1 is the name of a resource of type <*remoteCSE*>.
- APP1 is the name of a resource of type <*AE*>.
- CONT1 and CONT2 are the names of resources of type <*container*>.
- ACP1 and ACP2 are the names of resources of type <*accessControlPolicy*>.

Figure 9.4.1-1: Resource Relationships Example in a CSE

The solid line in figure 9.4.1-1 represents parent-child relation, which is supported by a link (e.g. *parentID*) in the non-hierarchical addressing method, and by the hierarchical addressing method.

Dashed line in figure 9.4.1-1 represents a link i.e. a relationship between the resources (e.g. relationship between the APP1 resource and the ACP1).

Figure 9.4.1-1 provides an example of a resource structure. The represented resources can be addressed by using one of the methods described in clause 9.3.1. Resources in the oneM2M System are linked with each other and they respect the containment relationship. The methods for linking resources are described in clause 9.4.2.

A link shall contain the following information:

- Linked Resource: The target linked resource is given by using the ID of that resource.
- Link Relation: Describes the relationship that the current resource has with the linked resource (only in one direction, i.e. from this resource to the linked resource).

9.4.2 Link Relations

The following link relations are defined.

Table 9.4.2-1: Link Relations

Linked Resource Type (link destination)	Linking Resource Types (link origin)	Linking Method	Description
accessControPolicy	Several (e.g. *node, AE, remoteCSE, container*)	Attribute named *accessControlPolicyIDs*	See clause 9.6.2
node	*CSEBase, remoteCSE, AE*	Attribute named *nodeLink*	See clause 9.6.3 See clause 9.6.4 See clause 9.6.5
CSEBase or *remoteCSE*	*node*	Attribute named *hostedCSELink* OR parent resource of type *CSEBase*	See clause 9.6.18
AE	*node*	*Attribute named hostedAELinks*	See clause 9.6.18
flexContainer	*node*	*Attribute named hostedServiceLinks*	See clause 9.6.18
a parent resource of any resourceType	a child resource of any resourceType	Attribute named *parentID*	See clause 9.6.1.3
a child resource of any resourceType	a parent resource of any resourceType	Child resource itself	See clause 9.6
mgmtObj	*mgmtObj*	Attribute named: *mgmtLink*	See clause 9.6.15
contentInstance	*contentInstance*	Attribute named *contentRef*	See clauses 9.6.7 and 9.6.35
dynamicAuthorizationConsultation	Several *(e.g. node, AE, remoteCSE, container)*	Attribute named: *dynamicAuthorizationConsultationIDs*	See clause 9.6.40

9.5 Resource Type Specification Conventions

9.5.0 Overview

The following conventions are used for the specification of resources.

Resources are specified via a tabular notation and the associated graphical representation as follows:

- The resources are specified in association with a CSE. The resources are the representation in the CSE of the components and elements within the oneM2M System. Other CSEs, AEs, application data representing sensors, commands, etc. are known to the CSE by means of their resource representation. Resource, Child Resource and Attributes are defined in clause 3.1 and are restated below for readability:
  - **Resource:** A Resource is a uniquely addressable entity in oneM2M architecture. A resource is transferred and manipulated using CRUD operations (see clause 10.1). A resource can contain child resource(s) and attribute(s).
  - **Child Resource:** A sub-resource of another resource that is its parent resource. The parent resource contains references to the child resources(s).
  - **Attribute:** Stores information pertaining to the resource itself.
- The set of attributes, which are common to all resources, are not detailed in the graphical representation of a resource.
- Resource names and attribute names are strings in lower case. In case of a composed name, the subsequent word(s) start with a capital letter; e.g. *accessControlPolicy*, *creationTime*, *expirationTime*.

- Resource type names and attribute names are written in *italic* form in the present document.
- A string containing resource type name in *italic* delimited with '<' and '>' e.g. <*resourceType*> is used as an abbreviation referring to the type of a resource. For example, the text "a <*container*> resource" could be used as an abbreviation for "a resource of type *container*".
- A string containing a resource type name delimited with '[' and ']' e.g. *[resourceType]* is an abbreviation referring to a specialization of a resource type.
- Specialization of a resource type is done by defining specific names and descriptions of the attributes that can be specialized from the base resource type. For example, the text "a [*battery*] resource" could be used as an abbreviation for "a resource of type *battery*", where battery is a specialization of base resource type *mgmtObj*.
- A string containing an attribute type name in italic delimited with '[' and ']', e.g. *[objectAttribute]* is used as an abbreviation referring to a type of an attribute that can be specialized. Attributes that can be specialized only occur in resource types that can be specialized.

The resources are specified as shown in figure 9.5.0-1.

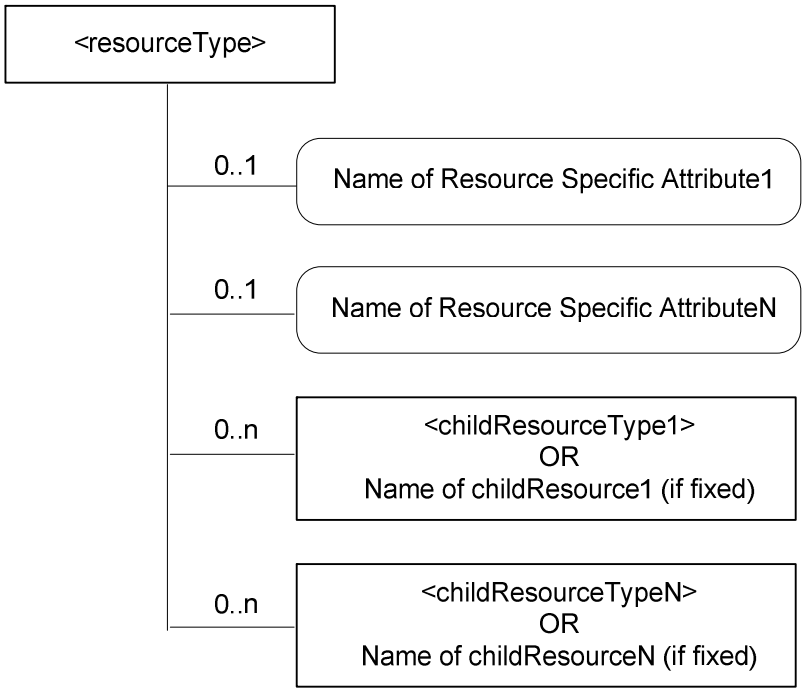


Figure 9.5.0-1: *<resourceType>* representation convention

The resource specification provides the graphical representation for the resource as in figure 9.5.0-1. The graphical representation of a resource shows the multiplicity of the attributes and child resources. The set of attributes, which are common to all resources are not detailed in the graphical representation of a resource. The following graphical representations are used for representing the attributes and child resources:

- Square boxes are used for the resources;
- Square boxes with round corners are used for attributes.

Child resources in a <*resourceType*> are detailed as shown in table 9.5.0-1.

The child resource table for an announce-able <*resourceType*> resource includes an additional column titled '<*resourceTypeAnnc*> Child Resource Types', indicating the type of announced resources. See clause 9.6.26 for further details.

An announced resource may have child resources, and such child resources can be of type "normal" or "announced". Child resources are of type "announced" when the child resources are announced independently of the original resource, as needed by the resource announcing CSE. Child resources are of type "normal" when child resources at the announced resource are created locally by the remote CSE.

Table 9.5.0-1: Child Resources of *<resourceType>*

Child Resources of *<resourceType>*	Child Resource Type	Multiplicity	Description	*<resourceTypeAnnc>* Child Resource Types
<Fill in the name of Child Resource1 if a fixed name is required or [variable] if no fixed name is required>	<Fill in the type of Child Resource1>	<Fill in Multiplicity>	See clause <XRef> <clause> where the type of this child resource is described.	<Fill the child resource type for the announced resource. It can be none or *<crTypeAnnc>* or *<crType>*; where the *<crType>* is the child resource type of the original Child Resource1.
<Fill in the name of Child ResourceN if a fixed name is required or [variable] if no fixed name is required>	<Fill in the type of Child ResourceN>	<Fill in Multiplicity>	See clause <XRef> <clause> where the type of this child resource is described.	<Fill the child resource type for the announced resource. It can be none or *<crTypeAnnc>* or *<crType>;* where the *<crType>* is the child resource type of the original Child ResourceN.

Attributes in a *<resourceType>* are detailed as shown in table 9.5.0-2.

The attributes table for announce-able *<resourceType>* resource includes an additional column titled 'Attributes for *<resourceTypeAnnc>*', indicating the attributes that are to be announced for that *<resourceType>*. See the clause 9.6.26 for further details.

Table 9.5.0-2: Attributes of *<resourceType>* resource

Attributes of *<resourceType>*	Multiplicity	RW/ RO/ WO	Description	*<resourceTypeAnnc>* (MA/OA/NA)
<Fill in name of Common Attribute1>	<Fill in Multiplicity>	<Fill in RW or RO or WO>	Provide description of this attribute - to be moved later to a common attribute clause.	<Fill in MA or OA or NA>
<Fill in name of Common AttributeN>	<Fill in Multiplicity>	<Fill in RW or RO or WO>	Provide description of this attribute - to be moved later to a common attribute clause.	<Fill in MA or OA or NA>
<Fill in name of Resource Specific Attribute1>	<Fill in Multiplicity>	<Fill in RW or RO or WO>	Provide description of this attribute - to be moved later to a central attribute table that also defines the type of the attribute, allowed ranges, etc.	<Fill in MA or OA or NA>
<Fill in name of Resource-Specific AttributeN>	<Fill in Multiplicity>	<Fill in RW or RO or WO>	Provide description of this attribute - to be moved later to a central attribute table that also defines the type of the attribute, allowed ranges, etc.	<Fill in MA or OA or NA>

In case of misalignment of the graphical representation of a resource and the associated tabular representation, tabular representation shall take precedence.

The access modes for *attributes* can assume the following values:

- Read/Write (RW): the value of the attribute is set when the resource is Created or Updated based on information from the Originator (i.e. ***Content*** parameter). Such attributes are allowed for Create/Update/Retrieve operations. Note that such an attribute can be deleted by Update operation.
- Read Only (RO): the value of the attribute is set or can be updated by the Hosting CSE internally. Such an attribute is allowed for Retrieve operation only.

- Write Once (WO): the value of the attribute is set when the resource is Created based on information from the Originator (i.e. ***Content*** parameter). Such an attribute is allowed for Retrieve operation after the creation. Such attribute can thereafter only be updated by hosting CSE internally.

The multiplicity, both for the child resources and the attributes can have the following values:

- A value of "0" indicates that the child resource/attribute shall not be present.
- A value of "1" indicates that the child resource/attribute shall be present.
- A value of "0..1" indicates that the child resource/attribute may be present.
- A value of "0..n" indicates that the child resource/attribute may be present. If present, multiple instances are supported.
- A value of "1..n" indicates that the child resource shall always be present. It has at least one instance and can have multiple instances.
- An attribute multiplicity post-fixed with (L) indicates that it is a list of values.

The attributes for <*resourceTypeAnnc*> in the attribute table can have the following set of values:

- **MA** (Mandatory Announced): The attribute in the original resource is announced to the announced resource. The content of such an announced attribute is the same as the content of the original attribute.
- **OA** (Optional Announced): The attribute in the original resource may be announced to the announced resource depending on the contents of the *announcedAttribute* attribute at the original resource. The content of such an announced attribute is the same as the content of the original attribute.
- **NA** (Not Announced): The original attribute is not announced to the announced resource.

9.5.1 Handling of Unsupported Resources/Attributes/Sub-resources within the M2M System

A CSE shall respond to a received request targeted to it and that includes resource(s), resource attribute(s) or sub-resource(s) that are not supported by it, by sending an appropriate error code back to the request Originator.

When a CSE is not the target entity of a received request, the CSE shall attempt to forward the received request to the targeted entity. If the CSE cannot forward the received request for any reason, it shall respond to the received request by sending an appropriate error code back to the request Originator. The present document includes both mandatory and optional functionalities for interfaces between oneM2M entities. Thus, the functionality implemented for the interfaces may not include all the functionalities specified in the present document.

9.6 Resource Types

9.6.1 Overview

9.6.1.1 Resource Type Summary

Table 9.6.1.1-1 introduces the normal and virtual resource types and their related child or parent resource types. Details of each resource type follow in the remainder of this clause.

Table 9.6.1.1-1 lists each specified ordinary - i.e. not announced - resource type. An addition of suffix "Annc" to the respective resource type identifier indicates the associated announced resource type. Resource types that can occur as child resources of announced resources are summarized in table 9.6.26.1-1.

Among the resource types listed in table 9.6.1.1-1, the following are termed "Content Sharing Resources" in oneM2M Specifications for the purpose of referring to any of those resource types:

- *container;*
- *contentInstance;*
- *flexContainer;*
- *timeSeries;*
- *timeSeriesInstance.*

Table 9.6.1.1-1: Resource Types

Resource Type	Short Description	Child Resource Types	Parent Resource Types	Clause
accessControlPolicy	Stores a representation of privileges. It is associated with resources that shall be accessible to entities external to the Hosting CSE. It controls "who" is allowed to do "what" and the context in which it can be used for accessing resources	*subscription, transaction*	*AE, AEAnnc, remoteCSE, remoteCSEAnnc, CSEBase, CSEBaseAnnc*	9.6.2
AE	Stores information about the AE. It is created as a result of successful registration of an AE with the Registrar CSE	*subscription, container, flexContainer, group, accessControlPolicy, pollingChannel, semanticDescriptor, timeSeries, transaction, transactionMgmt, triggerRequest, crossResourceSubscription, backgroundDataTransfer, semanticMashupInstance, dynamicAuthorizationConsultation, multimediaSession*	*CSEBase*	9.6.5
container	Shares data instances among entities. Used as a mediator that buffers data exchanged between AEs and/or CSEs. The exchange of data between AEs (e.g. an AE on a Node in a field domain and the peer-AE on the infrastructure domain) is abstracted from the need to set up direct connections and allows for scenarios where both entities in the exchange do not have the same reachability schedule	*container, flexContainer, contentInstance, subscription, latest, oldest, semanticDescriptor, timeSeries, transaction*	*AE, AEAnnc, container, containerAnnc, remoteCSE, remoteCSEAnnc, CSEBase, CSEBaseAnnc, flexContainer, flexContainerAnnc*	9.6.6
contentInstance	Represents a data instance in the *<container>* resource	*semanticDescriptor, transaction*	*Container, containerAnnc*	9.6.7
flexContainer	A template which allows to define specialized (customizable) versions of containers with a flexible and lightweight structure	*container, flexContainer, subscription, semanticDescriptor, timeSeries, transaction*	*AE, AEAnnc, container, containerAnnc, flexContainer, flexContainerAnnc, remoteCSE, remoteCSEAnnc, CSEBase, CSEBaseAnnc*	9.6.35

Resource Type	Short Description	Child Resource Types	Parent Resource Types	Clause
CSEBase	The structural root for all the resources that are residing on a CSE. Stores information about the CSE itself	*remoteCSE, remoteCSEAnnc, CSEBaseAnnc, node, AE, container, group, accessControlPolicy, subscription, mgmtCmd, locationPolicy, statsConfig, statsCollect, request, delivery, schedule, notificationTargetPolicy, flexContainer, timeSeries, AEContactList, transaction, transactionMgmt, crossResourceSubscription, backgroundDataTransfer, semanticMashupJobProfile, semanticMashupInstance, m2mServiceSubscriptionProfile , dynamicAuthorizationConsultation, localMulticastGroup, serviceSubscribedAppRule, authorizationPolicy, authorizationDecision, authorizationInformation, ontologyRepository*	*None specified*	9.6.3
delivery	Forwards requests from CSE to CSE	*subscription, transaction*	*CSEBase*	9.6.11
eventConfig	Defines events that trigger statistics collection	*subscription, transaction*	*statsConfig*	9.6.24
execInstance	Contains all execution instances of the same Management Command	*subscription, transaction*	*mgmtCmd*	9.6.17
fanOutPoint (V)	Virtual resource containing target for group request It is used for addressing bulk operations to all the resources that belong to a group	*None specified*	*group*	9.6.14
group	Stores information about resources of the same type that need to be addressed as a Group. Operations addressed to a Group resource shall be executed in a bulk mode for all members belonging to the Group	*fanOutPoint, subscription, semanticFanOutPoint, semanticDescriptor, transaction*	*AE, AEAnnc, remoteCSE, remoteCSEAnnc, CSEBase, CSEBaseAnnc*	9.6.13
latest (V)	Virtual resource that points to most recently created <*contentInstance*> and <*timeSeriesInstance*> child resource within a <*container*> *and a* <*timeSeries*> resource	*None specified*	*container, timeSeries*	9.6.27
locationPolicy	Includes information to obtain and manage geographical location. It is only referenced within a container, the *contentInstances* of the container provide location information	*subscription, transaction*	*CSEBase*	9.6.10

Resource Type	Short Description	Child Resource Types	Parent Resource Types	Clause
mgmtCmd	Management Command resource represents a method to execute management procedures required by existing management protocols	*execInstance, subscription, transaction*	*CSEBase*	9.6.16
mgmtObj	Management Object resource represents management functions that provides an abstraction to be mapped to external management technology. It represents the node and the software installed in the node (see note)	*subscription, transaction, semanticDescriptor*	*node, mgmtObjAnnc*	9.6.15 Annex D
m2mServiceSubscriptionProfile	Data pertaining to the M2M Service Subscription	*serviceSubscribedNode, subscription, transaction*	*CSEBase*	9.6.19
node	Represents specific Node information	*mgmtObj, subscription, semanticDescriptor, schedule, transaction*	*CSEBase*	9.6.18
notificationTargetMgmtPolicyRef	Represents a list of notification targets and the deletion policy	*subscription, transaction*	*subscription*	9.6.31
notificationTargetPolicy	Represents a notification target deletion policy with pre-defined action and deletion rules	*subscription, policyDeletionRules, transaction*	*CSEBase*	9.6.32
notificationTargetSelfReference (V)	Virtual resource used to remove the Notification Target	*None specified*	*subscription*	9.6.34
oldest (V)	Virtual resource that points to first created *<contentInstance>* and *<timeSeriesInstance>* child resource within a *<container> and a <timeSeries>* resource	*None specified*	*container, timeSeries*	9.6.28
pollingChannel	Represent a channel that can be used for a request-unreachable entity	*pollingChannelURI*	*remoteCSE, AE*	9.6.21
pollingChannelURI (V)	Virtual resource used to perform service layer long polling of a resource Hosting CSE by a request-unreachable entity	*None specified*	*pollingChannel*	9.6.22
policyDeletionRules	Represents a set of rules which is associated with notification target removal policy	*subscription, transaction*	*notificationTargetPolicy*	9.6.33

Resource Type	Short Description	Child Resource Types	Parent Resource Types	Clause
remoteCSE	Represents a remote CSE for which there has been a registration procedure with the registrar CSE identified by the CSEBase resource	*container, containerAnnc, contentInstanceAnnc flexContainer, flexContainerAnnc, group, groupAnnc, accessControlPolicy, accessControlPolicyAnnc, subscription, scheduleAnnc, pollingChannel, timeSeries, timeSeriesAnnc, timeSeriesInstanceAnnc, remoteCSEAnnc, mgmtObjAnnc, nodeAnnc, AEAnnc, locationPolicyAnnc, transaction, crossResourceSubscription, backgroundDataTransfer, semanticDescriptorAnnc, semanticMashupJobProfile, semanticMashupJobProfileAnnc, semanticMashupInstance, semanticMashupInstanceAnnc, dynamicAuthorizationConsultation, dynamicAuthorizationConsultationAnnc*	*CSEBase*	9.6.4
request	Expresses/access context of an issued Request	*subscription, transaction*	*CSEBase*	9.6.12
schedule	Contains scheduling information for delivery of messages	*subscription, transaction*	*subscription, CSEBase, node*	9.6.9
serviceSubscribedNode	Node information	*subscription, transaction*	*m2mServiceSubscriptionProfile*	9.6.20
statsCollect	Defines triggers for the IN-CSE to collect statistics for applications	*subscription, transaction*	*CSEBase (in IN-CSE)*	9.6.25
statsConfig	Stores configuration of statistics for applications	*eventConfig, subscription, transaction*	*CSEBase (in IN-CSE)*	9.6.23

Resource Type	Short Description	Child Resource Types	Parent Resource Types	Clause
subscription	Subscription resource represents the subscription information related to a resource. Such a resource shall be a child resource for the subscribed-to resource	*schedule, notificationTargetSelfReference, notificationTargetMgmtPolicyRef, transaction*	*accessControlPolicy, accessControlPolicyAnnc, AE, AEAnnc, container, containerAnnc, CSEBase, delivery, eventConfig, execInstance, group, groupAnnc, locationPolicy, locationPolicyAnnc, mgmtCmd, mgmtObj, mgmtObjAnnc, m2mServiceSubscriptionProfile, node, nodeAnnc, serviceSubscribedNode, remoteCSE, remoteCSEAnnc, CSEBaseAnnc, request, schedule, scheduleAnnc, semanticDescriptor, semanticDescriptorAnnc, statsCollect, statsConfig, flexContainer, flexContainerAnnc, timeSeries, timeSeriesAnnc*	9.6.8
serviceSubscribedAppRule	Represents a rule that defines allowed App-ID and AE-ID combinations that are acceptable for registering an AE on a Registrar CSE	*subscription, transaction*	*CSEBase*	9.6.29
semanticDescriptor	Stores semantic description pertaining to a resource and potentially sub-resources	*subscription, transaction*	*AE, container, contentInstance, group, node, flexContainer, timeSeries, mgmtObj*	9.6.30
semanticFanOutPoint	Virtual resource used as target for semantic discovery aimed at a logical graph distributed over multiple *semanticDescriptor* resources, which belong to the corresponding *group* parent resource		*group*	9.6.14a
dynamicAuthorizationConsultation	Represents consultation information used by a CSE when performing consultation-based dynamic authorization	*subscription, transaction*	*AE, AEAnnc, remoteCSE, remoteCSEAnnc, CSEBase, CSEBaseAnnc*	9.6.40

Resource Type	Short Description	Child Resource Types	Parent Resource Types	Clause
timeSeries	Stores and Shares Time Series Data instances among entities	*timeSeriesInstance, subscription, semanticDescriptor, latest, oldest, transaction*	*AE, AEAnnc, remoteCSE, remoteCSEAnnc, CSEBase, CSEBaseAnnc, container, containerAnnc, flexContainer, flexContainerAnnc*	9.6.36
timeSeriesInstance	Represents a Time Series Data instance in the <*timeSeries*> resource	*transaction*	*timeSeries, timeSeriesAnnc*	9.6.37
role	Represents a role that is assigned to an AE or CSE	*subscription, transaction*	*authorizationInformation*	9.6.38
token	Used for storing a token that is issued to an AE or CSE	*subscription, transaction*	*authorizationInformation*	9.6.39
authorizationDecision	Represents an access control decision point	*subscription, transaction*	*CSEBase*	9.6.41
authorizationPolicy	Represents an access control policy retrieval point	*subscription, transaction*	*CSEBase*	9.6.42
authorizationInformation	Represents an access control information point	*role token subscription, transaction*	*CSEBase*	9.6.43
localMulticastGroup	Stores local multicast group information of member hosting CSE.	*transaction*	*CSEBase*	9.6.44
AEContactList	Contains information about a CSE that has resources that referencing an AE-ID	*AEContactListPerCSE, subscription, transaction*	*CSEBase*	9.6.45
AEContactListPerCSE	Contains information about a CSE that has resources that referencing an AE resource identifier for tracking purposes	*None specified*	*AEContactList*	9.6.46
transactionMgmt		*subscription*	*CSEBase, AE, remoteCSE*	9.6.47
transaction		*None specified*	*All non-virtual resource types with the exception of the following:* *request, delivery, pollingChannel, transactionMgmt, transaction*	9.6.48
triggerRequest	Used by an AE to initiate, replace or recall a device trigger request	*subscription*	*AE*	9.6.49
ontologyRepository	Represents the collection of the managed ontologies and the semantic validation service	*ontology, semanticValidation, subscription*	*CSEBase*	9.6.50
ontology	Store the representation of an ontology	*subscription*	*ontologyRepository*	9.6.51

Resource Type	Short Description	Child Resource Types	Parent Resource Types	Clause
semanticValidation	A virtual resource as the interface to perform semantic validation on the received <semanticDescriptor> resource against the referenced ontology	*None specified*	*ontologyRepository*	9.6.52
semanticMashupJobProfile	Represents the profile and description of a semantic mashup service	*semanticMashupInstance, semanticDescriptor, subscription*	*CSEBase, remoteCSE*	9.6.53
semanitcMashupInstance	Represents a semantic mashup instance	*semanticMashupResult, semanticDescriptor, mashup, subscription*	*semanticMashupJobProfile, AE, remoteCSE, CSEBase*	9.6.54
mashup	A virtual resource use to trigger the calculation and generation of new mashup result	*Not specified*	*semanticMashupInstance*	9.6.55
semanticMashupResult	Represent semantic mashup results	*semanticDescriptor, subscription*	*semanticMashupInstance*	9.6.56
multimediaSession	Stores a representation of a multimedia session information requested by a registering AE	*subscription*	*AE*	9.6.57
crossResourceSubscription	represents the cross-resource subscription information related to multiple subscribed-to resources. Such a resource shall include a list of subscribed-to resources as its attribute, or shall be created as a child resource of a <group> resource where member resources shall be the subscribed-to resources	*schedule, notificationTargetSelfReference, notificationTargetMgmtPolicyRef, transaction*	*CSEBase, remoteCSE, AE*	9.6.58
backgroundDataTransfer	Stores information for a background data transfer request	*None specified*	*AE, remoteCSE, CSEBase*	9.6.60
NOTE: See clause 9.6.12 for a summary of specializations of *<mgmtObj>*.				

9.6.1.2 Resource Type Specializations

9.6.1.2.1 Specializations of <*mgmtObj*>

Table 9.6.1.2.1-1 lists specializations of the <*mgmtObj*> resource type in which the *mgmtDefinition* attribute contains an enumerated value that provides further definition of the resource.

Table 9.6.1.2.1-1: <*mgmtObj*> Specializations

Resource specialization	Short Description	Child Resource Types	Parent Resource Types	Clause
activeCmdhPolicy	Provides a link to the currently active set of CMDH policies	None specified	*node*	D.12.1
areaNwkDeviceInfo	Provides information about the Node in the M2M Area Network	*subscription*	*node*	D.6
areaNwkInfo	Describes the list of Nodes attached behind the MN node and its physical or underlying relation among the nodes in the M2M Area Network	*subscription*	*node*	D.5
battery	Provides the power information of the node (e.g. remaining battery charge)	*subscription*	*node*	D.7
cmdhBuffer	Defines CMDH buffer usage limits	*subscription*	*cmdhPolicy*	D.12.8
cmdhDefaults	Defines CMDH default values	*cmdhDefEcValue, cmdhEcDefParamValues subscription*	*cmdhPolicy*	D.12.2
cmdhEcDefParamValues	Represent a specific set of default values for the CMDH related parameters	*subscription*	*cmdhDefaults*	D.12.4
cmdhDefEcValue	Defines a value for the ***Event Category*** parameter of an incoming request when it is not defined	*subscription*	*cmdhDefaults*	D.12.3
cmdhLimits	Defines limits for CMDH related parameter values	*subscription*	*cmdhPolicy*	D.12.5
cmdhNetworkAccessRules	Defines rules for the usage of underlying networks	*cmdhNwAccessRule, subscription*	*cmdhPolicy*	D.12.6
cmdhNwAccessRule	Defines a rule for the usage of underlying networks	*subscription*	*cmdhNetworkAccessRules*	D.12.7
cmdhPolicy	A set of rules defining which CMDH parameters will be used by default	*cmdhDefaults, cmdhLimits, cmdhNetworkAccessRules, cmdhBuffer, subscription*	*node*	D.12
deviceCapability	Contains information about the capability supported by the Node	*subscription*	*node*	D.9
deviceInfo	Contains information about the identity, manufacturer and model number of the device	*subscription*	*node*	D.8
eventLog	Contains information about the log of events of the Node	*subscription*	*node*	D.11
firmware	Provides information about the firmware of the Node (e.g. name, version)	*subscription*	*node*	D.2
memory	Provides the memory (typically RAM) information of the node (e.g. the amount of total volatile memory)	*subscription*	*node*	D.4
reboot	Used to reboot or reset the Node	*subscription*	*node*	D.10
software	Provides information about the software of the Node	*subscription*	*node*	D.3
registration	To convey the service layer configuration information	*subscription*	*node*	7.1 in [10]
dataCollection	To convey the application configuration information	*subscription*	*node*	7.2 in [10]
authenticationProfile	To convey the configuration information regarding establishing mutually-authenticated secure communications	*subscription*	*node*	7.1 in [10]

Resource specialization	Short Description	Child Resource Types	Parent Resource Types	Clause
myCertFileCred	To configure a certificate or certificate chain	*subscription*	*authenticationProfile*	7.1 in [10]
trustAnchorCred	To identify a trust anchor certificate for validation of certificates	*subscription*	*authenticationProfile*	7.1 in [10]
MAFClientRegCfg	To convey instructions regarding the MAF Client Registration procedure	*subscription*	*authenticationProfile*	7.1 in [10]

9.6.1.2.2 Specializations of <*flexContainer*>

9.6.1.2.2.1 <*flexContainer*> for generic interworking

Generic interworking (specified in ETSI TS 118 112 [6]) provides interworking with many types of non-oneM2M area networks and their devices. It supports the interworking variant "full mapping of the semantic of the non-oneM2M data model to Mca" as indicated in clause F.2.

Generic interworking can be done for non-oneM2M systems for which no oneM2M specified interworking exists. For generic interworking the non-oneM2M data model of the non- oneM2M area network needs to be described in the form of a oneM2M compliant ontology which is derived from the oneM2M base ontology (specified in ETSI TS 118 112 [6]) and that may be available in a formal description language (e.g. OWL).

Table 9.6.1.2.2.1-1 lists specializations of the <*flexContainer*> resource type for generic interworking in which the *containerDefinition* attribute provides further definition of the resource.

Table 9.6.1.2.2.1-1: <*flexContainer*> Specializations for Generic Interworking

Resource specialization	Short Description	Child Resource Types	Parent Resource Types	Clause
genericInterworkingService	This resource type is used to link to a set of input- and outputDataPoints of the service and to provide the parent resource for operationInstance resources of the service	*genericInterworkingService, genericInterworkingOperationInstance, semanticDescriptor, subscription*	*AE, container, flexContainer, genericInterworkingService*	9 in [6]
genericInterworkingOperationInstance	This resource type is used to link to a set of input- and outputDataPoints and a set of operationInputs and operationOutputs of the operationInstance	*semanticDescriptor, subscription*	*genericInterworkingService*	9 in [6]

9.6.1.2.2.2 <*flexContainer*> for AllJoyn interworking

Table 9.6.1.2.2.2-1 contains the list of <flexContainer> specialization resources for oneM2M and AllJoyn interworking defined in ETSI TS 118 121 [7]. This also summarizes parent-child relationship for each specialization resource type.

Table 9.6.1.2.2.2-1: <*flexContainer*> Specializations

Resource specialization	Short Description	Child Resource Types	Parent Resource Types	Clause
svcObjWrapper	AllJoyn service object wrapper	*allJoynApp*	*AE*	B.2 in [7]
svcFwWrapper	AllJoyn service framework wrapper	*n/a*	*AE*	B.3 in [7]
allJoynApp	AllJoyn application	*allJoynSvcObject*	*svcObjWrapper*	B.4 in [7]
allJoynSvcObject	AllJoyn service object	*allJoynInterface*	*allJoynApp*	B.5 in [7]
allJoynInterface	AllJoyn interface	*allJoynMethod, allJoynProperty*	*allJoynSvcObject*	B.6 in [7]
allJoynMethod	AllJoyn method	*allJoynMethodCall*	*allJoynInterface*	B.7 in [7]
allJoynMethodCall	AllJoyn method call	*n/a*	*allJoynMethod*	B.8 in [7]
allJoynProperty	AllJoyn property	*n/a*	*allJoynInterface*	B.9 in [7]

9.6.1.2.2.3 *<flexContainer>* for Home Appliance Information Model

Table 9.6.1.2.2.3-1 contains the list of <flexContainer> specialization resources for oneM2M Home Appliance Information Model (HAIM) defined in ETSI TS 118 123 [8]. This also summarizes parent-child relationship for each specialization resource type.

Table 9.6.1.2.2.3-1: *<flexContainer>* Specializations

Resource specialization	Short Description	Child Resource Types	Parent Resource Types	Clause
alarmSpeaker	HAIM ModuleClass	*subscription*	*n/a*	5.3.1 in [8]
audioVideoInput	HAIM ModuleClass	*subscription*	*television*	5.3.2 in [8]
audioVolume	HAIM ModuleClass	*upVolume, downVolume subscription*	*television*	5.3.3 in [8]
battery	HAIM ModuleClass	*subscription*	*electricVehicleCharger, robotCleaner, storageBattery*	5.3.4 in [8]
binarySwitch	HAIM ModuleClass	*toggle, subscription*	*airConditioner, clothesWasher, electricVehicleCharger, light, microgeneration, oven, refrigerator, robotCleaner, smartElectricMeter, storageBattery, television, waterHeater*	5.3.5 in [8]
bioElectricalImpedanceAnalysis	HAIM ModuleClass	*subscription*	*n/a*	5.3.6 in [8]
boiler	HAIM ModuleClass	*subscription*	*waterHeater*	5.3.7 in [8]
brightness	HAIM ModuleClass	*subscription*	*light*	5.3.8 in [8]
clock	HAIM ModuleClass	*subscription*	*smartElectricMeter, waterHeater*	5.3.9 in [8]
colour	HAIM ModuleClass	*subscription*	*light*	5.3.10 in [8]
colourSaturation	HAIM ModuleClass	*subscription*	*light*	5.3.11 in [8]
doorStatus	HAIM ModuleClass	*subscription*	*refrigerator*	5.3.12 in [8]
electricVehicleConnector	HAIM ModuleClass	*subscription*	*electricVehicleCharger*	5.3.13 in [8]
energyConsumption	HAIM ModuleClass	*subscription*	*smartElectricMeter*	5.3.14 in [8]
energyGeneration	HAIM ModuleClass	*subscription*	*Microgeneration, smartElectricMeter*	5.3.15 in [8]
faultDetection	HAIM ModuleClass	*subscription*	*electricVehicleCharger, light microgeneration, smartElectricMeter, storageBattery, waterHeater*	5.3.16 in [8]
height	HAIM ModuleClass	*subscription*	*n/a*	5.3.17 in [8]
hotWaterSupply	HAIM ModuleClass	*subscription*	*waterHeater*	5.3.18 in [8]
keypad	HAIM ModuleClass	*subscription*	*n/a*	5.3.19 in [8]
motionSensor	HAIM ModuleClass	*subscription*	*n/a*	5.3.20 in [8]
oximeter	HAIM ModuleClass	*subscription*	*n/a*	5.3.21 in [8]
powerSave	HAIM ModuleClass	*subscription*	*refrigerator*	5.3.22 in [8]
pushButton	HAIM ModuleClass	*subscription*	*n/a*	5.3.23 in [8]
recorder	HAIM ModuleClass	*subscription*	*n/a*	5.3.24 in [8]
refrigeration	HAIM ModuleClass	*subscription*	*refrigerator*	5.3.25 in [8]
relativeHumidity	HAIM ModuleClass	*subscription*	*n/a*	5.3.26 in [8]
rinseLevel	HAIM ModuleClass	*subscription*	*clothesWasher*	5.3.27 in [8]
runMode	HAIM ModuleClass	*subscription*	*airConditioner, clothesWasher, electricVehicleCharger, light, microgeneration, oven, robotCleaner, smartElectricMeter, storageBattery, thermostat, waterHeater*	5.3.28 in [8]
signalStrength	HAIM ModuleClass	*subscription*	*n/a*	5.3.29 in [8]

Resource specialization	Short Description	Child Resource Types	Parent Resource Types	Clause
smokeSensor	HAIM ModuleClass	*subscription*	*n/a*	5.3.30 in [8]
spinLevel	HAIM ModuleClass	*subscription*	*clothesWasher*	5.3.31 in [8]
televisionChannel	HAIM ModuleClass	*upChannel, downChannel, subscription*	*television*	5.3.32 in [8]
temperature	HAIM ModuleClass	*subscription*	*airConditioner, clothesWasher, oven, refrigerator, thermostat*	5.3.33 in [8]
temperatureAlarm	HAIM ModuleClass	*subscription*	*n/a*	5.3.34 in [8]
timer	HAIM ModuleClass	*activateClockTimer, deactivateClockTimer, subscription*	*airConditioner, clothesWasher, oven, robotCleaner, thermostat*	5.3.35 in [8]
turbo	HAIM ModuleClass	*subscription*	*airConditioner*	5.3.36 in [8]
waterFlow	HAIM ModuleClass	*subscription*	*clothesWasher*	5.3.37 in [8]
waterLevel	HAIM ModuleClass	*subscription*	*clothesWasher*	5.3.38 in [8]
waterSensor	HAIM ModuleClass	*subscription*	*n/a*	5.3.39 in [8]
weight	HAIM ModuleClass	*subscription*	*n/a*	5.3.40 in [8]
wind	HAIM ModuleClass	*subscription*	*airConditioner*	5.3.41 in [8]
activateClockTimer	HAIM Action	*subscription*	*timer*	C.1 in [8]
deactivateClockTimer	HAIM Action	*subscription*	*timer*	C.2 in [8]
downChannel	HAIM Action	*subscription*	*televisionChannel*	C.3 in [8]
downVolume	HAIM Action	*subscription*	*audioVolume*	C.4 in [8]
toggle	HAIM Action	*subscription*	*binarySwitch*	C.5 in [8]
upChannel	HAIM Action	*subscription*	*televisionChannel*	C.6 in [8]
upVolume	HAIM Action	*subscription*	*audioVolume*	C.7 in [8]
airConditioner	HAIM Device	*binarySwitch, runMode, temperature, timer, turbo, wind, subscription*	*AE*	5.4.1 in [8]
clothesWasher	HAIM Device	*binarySwitch, timer, runMode, temperature, waterLevel, rinseLevel, waterFlow, spinLevel, subscription*	*AE*	5.4.2 in [8]
electricVehicleCharger	HAIM Device	*faultDetection, binarySwitch, runMode, battery, electricVehicleConnector, subscription*	*AE*	5.4.3 in [8]
light	HAIM Device	*faultDetection, binarySwitch, runMode, colour, colourSaturation, brightness, subscription*	*AE*	5.4.4 in [8]
microgeneration	HAIM Device	*faultDetection, binarySwitch, runMode, energyGeneration, subscription*	*AE*	5.4.5 in [8]
oven	HAIM Device	*binarySwitch, runMode, timer, temperature, subscription*	*AE*	5.4.6 in [8]
refrigerator	HAIM Device	*binarySwitch, powerSave, doorStatus, temperature, refrigeration, subscription*	*AE*	5.4.7 in [8]
robotCleaner	HAIM Device	*binarySwitch, runMode, battery, timer, subscription*	*AE*	5.4.8 in [8]

Resource specialization	Short Description	Child Resource Types	Parent Resource Types	Clause
smartElectricMeter	HAIM Device	*faultDetection, binarySwitch, runMode, clock, energyConsumption, energyGeneration, subscription*	*AE*	5.4.9 in [8]
storageBattery	HAIM Device	*faultDetection, binarySwitch, runMode, battery, subscription*	*AE*	5.4.10 in [8]
television	HAIM Device	*binarySwitch, audioVolume, televisionChannel, audioVideoInput, subscription*	*AE*	5.4.11in [8]
thermostat	HAIM Device	*runMode, timer, temperature, subscription*	*AE*	5.4.12 in [8]
waterHeater	HAIM Device	*faultDetection, binarySwitch, runMode, clock, boiler, hotWaterSupply, subscription*	*AE*	5.4.13 in [8]
deviceProperty	HAIM Property	*subscription*	*airConditioner, clothesWasher, electricVehicleCharger, light, microgeneration, oven, refrigerator, robotCleaner, smartElectricMeter, storageBattery, television, thermostat waterHeater*	D.1 in [8]
moduleClassProperty	HAIM Property	*subscription*	*battery, electricVehicleConnector*	D.2 in [8]

9.6.1.3 Commonly Used Attributes

9.6.1.3.0 Overview

Some attributes described herein are present in all <*resourceTypes*>. Such attributes are described in clause 9.6.1.3.1 once in order to avoid duplicating the description for every <*resourceType*> and are referred to as "universal attributes".

Some other attributes described herein are commonly used in multiple, but not all, <*resourceTypes*>. Such attributes are described in clause 9.6.1.3.2 once in order to avoid duplicating the description for every <*resourceType*> that contains it and are referred to as "common attributes".

Remaining attributes are described in the clause specific for that resource type.

9.6.1.3.1 Universal attributes

The following attributes are universal to all resource types which are normal, not virtual or announced. Universal attributes for announced resource types are independently defined in clause 9.6.26.2.

Table 9.6.1.3.1-1: Universal Attributes

Attribute Name	Description
resourceType	Resource Type. This Read Only (assigned at creation time. and then cannot be changed) attribute identifies the type of the resource as specified in clause 9.6. Each resource shall have a *resourceType* attribute.
resourceID	This attribute is an identifier for the resource that is used for 'non-hierarchical addressing method', i.e. this attribute shall contain the 'Unstructured-CSE-relative-Resource-ID' format of a resource ID as defined in table 7.2-1. This attribute shall be provided by the Hosting CSE when it accepts a resource creation procedure. The Hosting CSE shall assign a resourceID which is unique in that CSE.
resourceName	This attribute is the name for the resource that is used for 'hierarchical addressing method' to represent the parent-child relationships of resources. See clause 7.2 for more details. This attribute may be provided by the resource creator. The Hosting CSE shall use a provided *resourceName* as long as it does not already exist among child resources of the targeted parent resource. If the *resourceName* already exists, the Hosting CSE shall reject the request and return an error to the Originator. The Hosting CSE shall assign a *resourceName* if one is not provided by the resource creator.
parentID	This attribute is the *resourceID* of the parent of this resource. The value of this attribute shall be an empty string for the <CSEBase> resource type.
creationTime	Time/date of creation of the resource. This attribute is mandatory for all resources and the value is assigned by the system at the time when the resource is locally created. Such an attribute cannot be changed.
lastModifiedTime	Last modification time/date of the resource. The lastModifiedTime value is set by the Hosting CSE when the resource is created, and the lastModifiedTime value is updated when the resource is updated.

9.6.1.3.2 Common attributes

The following attributes are commonly used in multiple, but not all, resource types which are normal, not virtual or announced. Common attributes for announced resource types are independently defined in clause 9.6.26.3.

NOTE: The list of attributes in table 9.6.1.3.2-1 is not exhaustive.

Table 9.6.1.3.2-1: Common Attributes

Attribute Name	Description
accessControlPolicyIDs	The attribute contains a list of identifiers for <*accessControlPolicy*> resources. The privileges defined in the <*accessControlPolicy*> resources that are referenced determine who is allowed to access the resource containing this attribute for a specific purpose (e.g. Retrieve, Update, Delete, etc.). For an Update operation to a resource, it is forbidden to change the *accessControlPolicyIDs* attribute in the same request to Update other attributes of the targeted resource, i.e. a request to Update the *accessControlPolicyIDs* attribute shall be the only attribute in the UPDATE request. To update the *accessControlPolicyIDs* attribute, a Hosting CSE shall check whether the Originator has Update privilege in any current *selfPrivileges,* of the <*accessControlPolicy*> resources which this attribute references. To update any attribute other than the *accessControlPolicyIDs* attribute, a Hosting CSE shall check whether the Originator has Update privilege in any *privileges,* of the <*accessControlPolicy*> resources which the *accessControlPolicyIDs* attribute references. If a resource type does not have an *accessControlPolicyIDs* attribute definition, then the *accessControlPolicyIDs* for that resource is governed in a different way, for example, the *accessControlPolicy* associated with the parent may apply to a child resource that does not have an *accessControlPolicyIDs* attribute definition, or the privileges for access are fixed by the system. Refer to the corresponding resource type definitions and procedures to see how access control is handled in such cases.

Attribute Name	Description
	If a resource type does have an *accessControlPolicyIDs* attribute definition, but the (optional) *accessControlPolicyIDs* attribute value is not set in a resource instance, then the Hosting CSE shall apply the concept of the default access policy. The default policy shall provide unrestricted access only to the Originator of the successful resource creation request. All other entities shall be denied to access the resource. For that purpose, the Hosting CSE shall keep that Originator information of the resource. Note that how to keep that information is implementation specific. The default access policy is not applied to a resource which has a value assigned to the a*ccessControlPolicyIDs* attribute. All resources are accessible if and only if the privileges (i.e. configured as *privileges* or *selfPrivileges* attribute of <accessControlPolicy> resource) allow it, therefore all resources shall have an associated *accessControlPolicyIDs* attribute, either explicitly (setting the attribute in the resource itself) or implicitly (either by using the parent privileges or the system default policies). Which means that the system shall provide default access privileges in case that the Originator does not provide a specific *accessControlPolicyIDs* during the creation of the resource.
expirationTime	Time/date after which the resource will be deleted by the Hosting CSE. This attribute can be provided by the Originator, and in such a case it will be regarded as a hint to the Hosting CSE on the lifetime of the resource. The Hosting CSE shall configure the *expirationTime* value. If the Hosting CSE configures the new *expirationTime* attribute value rather than the Originator suggested value, the new value can be sent back to the Originator depending on the ***Result Content*** value. The lifetime of the resource can be extended by providing a new value for this attribute in an UPDATE operation. Or by deleting the attribute value, e.g. by updating the attribute with NULL when doing a full UPDATE, in which case the Hosting CSE can decide on a new value. If the Originator does not provide a value in the CREATE operation the system shall assign an appropriate value depending on its local policies and/or M2M service subscription agreements. A resource is known as 'obsolete' when the resource contains the attribute "expirationTime" and the lifetime of this resource has reached the value of this attribute. If the 'obsolete' resource had a reference to an Application Entity Resource ID, the Hosting CSE shall send a NOTIFY request to the IN-CSE, requesting to delete the entry from the <AEContactList> resource.
stateTag	An incremental counter of modification on the resource. When a resource is created, this counter is set to 0, and it will be incremented on every modification of the resource (see notes 1 and 2).
announceTo	This attribute may be included in a CREATE or UPDATE Request in which case it contains a list of addresses/CSE-IDs where the resource is to be announced. For the case that CSE-IDs are provided, the announced-to CSE shall decide the location of the announced resources based on the rules described in clause 9.6.26. For the original resource, this attribute shall only be present if it has been successfully announced to other CSEs. This attribute maintains the list of the resource addresses to the successfully announced resources. Updates on this attribute will trigger new resource announcement or de-announcement. If *announceTo* attribute includes resource address(s), the present document does not provide any means for validating these address(s) for announcement purposes. It is the responsibility of the Hosting-CSE referenced by the resource address(s) to validate the access privileges of the originator of the Request that triggers the announcement.
announcedAttribute	This attributes shall only be present at the original resource if some Optional Announced **(OA)** type attributes have been announced to other CSEs. This attribute maintains the list of the announced Optional Attributes (**OA** type attributes) in the original resource. Updates to this attribute will trigger new attribute announcement if a new attribute is added or de-announcement if the existing attribute is removed.
labels	Tokens used to add meta-information to resources. This attribute is optional. The value of the *labels* attribute is a list of individual labels, each of them being: - Either a standalone label-key, used as a simple "tag", that can be used for example for discovery purposes when looking for particular resources that one can "tag" using that label-key.

Attribute Name	Description
	- Or a composite element made of a label-key and a label-value, separated by a special character defined in ETSI TS 118 104 [3]. The list of allowed characters in a label (and in label-keys and label-values) and separator characters is defined in ETSI TS 118 104 [3], clause 6.3.3.
e2eSecInfo	Present in a resource representing an AE or CSE. Indicates the end-to-end security capabilities supported by the AE or CSE. May indicate supported end-to-end security frameworks. May also contains a certificate or credential identifier used by the AE or CSE. May include random values for use in end-to-end security protocols. The details of this attributes are described in ETSI TS 118 103 [2]. This attribute is optional and if not present it means that the represented entity does not support oneM2M end-to-end security procedures.
dynamicAuthorizationConsultationIDs	This attribute contains a list of identifiers of <*dynamicAuthorizationConsultation*> resources. The information defined in a <*dynamicAuthorizationConsultation*> resource is used by a CSE for initiating consultation-based dynamic authorization requests. Consultation-based dynamic authorization is only performed for a targeted resource if and only if it is linked to an enabled <*dynamicAuthorizationConsultation*> resource. If the attribute is not set or has a value that does not correspond to a valid <*dynamicAuthorizationConsultation*> resource(s), or it refers to an <*dynamicAuthorizationConsultation*> resource(s) that is not reachable, then the *dynamicAuthorizationConsultationIDs* associated with the parent may apply to the child resource if present, or a system default <*dynamicAuthorizationConsultation*> may apply if present.
creator	The AE-ID or CSE-ID of the entity which created the resource containing this attribute.
NOTE 1: In order to enable detection of overflow, the counter needs to be capable of expressing sufficiently long numbers. NOTE 2: This attribute has the scope to allow identifying changes in resources within a time interval that is lower than the one supported by the attribute *lastModifiedTime* (e.g. less than a second or millisecond). This attribute can also be used to avoid race conditions in case of competing modifications.	

9.6.2 Resource Type *accessControlPolicy*

9.6.2.0 Introduction

The Access Control Policies (ACPs) shall be used by the CSE to control access to the resources as specified in the present document and in ETSI TS 118 103 [2].

The ACP is designed to fit different access control models such as access control lists, role or attribute based access control.

The <*accessControlPolicy*> resource is comprised of *privileges* and *selfPrivileges* attributes which represent a set of access control rules defining which entities (defined as *accessControlOriginators*) have the privilege to perform certain operations (defined as *accessControlOperations*) within specified contexts (defined as *accessControlContexts*) and are used by the CSEs in making Access Decision to all or specific parts of the targeted resource (defined as *accessControlObjectDetails*).

In a privilege, each access control rule defines which AE/CSE is allowed for which operation. So for sets of access control rules an operation is permitted if it is permitted by one or more access control rules in the set.

For a resource that is not of <*accessControlPolicy*> resource type, the common attribute *accessControlPolicyIDs* for such resources (defined in table 9.6.1.3.2-1) contains a list of identifiers which link that resource to <*accessControlPolicy*> resources. The CSE Access Decision for such a resource shall follow the evaluation of the set of access control rules expressed by the *privileges* attributes defined in the <*accessControlPolicy*> resources.

The *selfPrivileges* attribute shall represent the set of access control rules for the <*accessControlPolicy*> resource itself.

The CSE Access Decision for <*accessControlPolicy*> resource shall follow the evaluation of the set of access control rules expressed by the *selfPrivileges* attributes defined in the <*accessControlPolicy*> resource itself.

Logically an authorization system may comprise four sub-functions: enforcing access control decision, making access control decision, providing access control policies and providing access control information (e.g. roles). As specified in ETSI TS 118 103 [2], these sub-functions are modelled as policy enforcement point (PEP), Policy Decision Point (PDP), Policy Retrieval Point (PRP) and Policy Information Point (PIP) respectively. In a oneM2M System these authorization sub-functions may coexist in one CSE or may be distributed in different CSEs in different combinations.

In the <*accessControlPolicy*> resource three operational attributes are defined for holding the information about where to find the distributed authorization sub-functions. These attributes are: *authorizationDecisionResourceIDs*, *authorizationPolicyResourceIDs* and *authorizationInformationResourceIDs*.

The *authorizationDecisionResourceIDs* attribute contains a list of addresses of <*authorizationDecision*> resources. Each <*authorizationDecision*> resource represents a PDP to which an access control decision request shall be sent in order to obtain an access control decision. See clause 9.6.41 for further details of <*authorizationDecision*> resource type.

The *authorizationPolicyResourceIDs* attribute contains a list of addresses of <*authorizationPolicy*> resources. Each <*authorizationPolicy*> resource represents a PRP to which an access control policy request shall be sent in order to obtain access control policies. See clause 9.6.42 for further details of <*authorizationPolicy*> resource type.

The *authorizationInformationResourceIDs* attribute contains a list of addresses of <*authorizationInformation*> resources. Each <*authorizationInformation*> resource represents a PIP to which an access control information request shall be sent in order to obtain requested access control information (e.g. role and/or token) for making an access control decision. See clause 9.6.43 for further details of <*authorizationInformation*> resource type.

When processing a request to a targeted resource, the Hosting CSE shall progress through the different types of authorization (if supported) as described in clause 10.2.3.1.

The applicability of the *authorizationDecisionResourceIDs*, *authorizationPolicyResourceIDs* and *authorizationInformationResourceIDs* attributes for the distributed authorization depends on the deployment form of authorization sub-functions:

- In the case the *privileges* attribute is not NULL, the access control rules in the *privileges* attribute shall be used for access control, and the *authorizationDecisionResourceIDs*, *authorizationPolicyResourceIDs* and *authorizationInformationResourceIDs* attributes shall not be present.
- In the case the *privileges* attribute is NULL, how to process further depends on which authorization method is adopted. In the case distributed authorization method is supported, *authorizationDecisionResourceIDs* or *authorizationPolicyResourceIDs* attribute shall be considered for obtaining access control decision or access control policies from another CSE. However, *authorizationDecisionResourceIDs* and *authorizationPolicyResourceIDs* attributes shall not be present at the same time.
- In case the *authorizationInformationResourceIDs* attribute is present, the access control information request (e.g. for role information) related to the access control policy specified in the *privileges* attribute shall be sent to one of the addresses listed in this attribute.

The details of distributed authorization procedures are described in ETSI TS 118 103 [2].

The <*accessControlPolicy*> resource shall contain the child resource specified in table 9.6.2.0-1.

Table 9.6.2.0-1: Child resources of <*accessControlPolicy*> resource

Child Resources of <*accessControlPolicy*>	Child Resource Type	Multiplicity	Description	<*accessControlPolicyAnnc*> Child Resource Types
[variable]	<*subscription*>	0..n	See clause 9.6.8	<*subscription*>
[variable]	<*transaction*>	0..n	See clause 9.6.48	<*transaction*>

The <*accessControlPolicy*> resource shall contain the attributes specified in table 9.6.2.0-2.

Table 9.6.2.0-2: Attributes of *<accessControlPolicy>* resource

Attributes of *<accessControlPolicy>*	Multiplicity	RW/ RO/ WO	Description	*<accessControlPolicyAnnc>* Attributes
resourceType	1	RO	See clause 9.6.1.3.	NA
resourceID	1	RO	See clause 9.6.1.3.	NA
resourceName	1	WO	See clause 9.6.1.3.	NA
parentID	1	RO	See clause 9.6.1.3.	NA
expirationTime	1	RW	See clause 9.6.1.3.	MA
labels	0..1(L)	RW	See clause 9.6.1.3.	MA
creationTime	1	RO	See clause 9.6.1.3.	NA
lastModifiedTime	1	RO	See clause 9.6.1.3.	NA
announceTo	0..1 (L)	RW	See clause 9.6.1.3.	NA
announcedAttribute	0..1 (L)	RW	See clause 9.6.1.3.	NA
privileges	1	RW	A set of access control rules that applies to resources referencing this <*accessControlPolicy*> resource using the *accessControlPolicyID* attribute.	MA
selfPrivileges	1	RW	A set of access control rules that apply to the <*accessControlPolicy*> resource itself and *accessControlPolicyIDs* attribute of any other resource which is linked to this <accessControlPolicy> resource.	MA
authorizationDecisionResourceIDs	0..1 (L)	RW	A list of addresses of <*authorizationDecision*> resources. See clause 9.6.41 for further details.	MA
authorizationPolicyResourceIDs	0..1 (L)	RW	A list of addresses of <*authorizationPolicy*> resources. See clause 9.6.42 for further details.	MA
authorizationInformationResourceIDs	0..1 (L)	RW	A list of addresses of <*authorizationInformation*> resources. See clause 9.6.43 for further details.	MA

The set of access control rules represented in *privileges* and *selfPrivileges* attributes are comprised of 4-tuples (*accessControlOriginators*, *accessControlContexts*, *accessControlOperations*, *accessControlObjectDetails*) with parameters shown in table 9.6.2.0-3 which are further described in the following clauses.

If the *privileges* attribute contains no 4-tuples, then this represents an empty set of the access control rules.

The *selfPrivileges* attribute shall contain at least one tuple.

The CSE access granting mechanism shall follow the procedure described in ETSI TS 118 103 [2], clause 7.1 (Access Control Mechanism).

Table 9.6.2.0-3: Parameters in access-control-rule-tuples

Name	Description
accessControlOriginators	See clause 9.6.2.1
accessControlContexts	See clause 9.6.2.2
accessControlOperations	See clause 9.6.2.3
accessControlObjectDetails	See clause 9.6.2.4
accessControlAuthenticationFlag	See clause 9.6.2.5

9.6.2.1 *accessControlOriginators*

The *accessControlOriginators* is a mandatory parameter in an access-control-rule-tuple. It represents the set of Originators that shall be allowed to use this access control rule. The set of Originators is described as a list of parameters, where the types of the parameter can vary within the list. Table 9.6.2.1-1 describes the supported types of parameters in *accessControlOriginators*. The following Originator privilege types shall be considered for access control policy check by the CSE.

Table 9.6.2.1-1: Types of Parameters in *accessControlOriginators*

Name	Description
domain	A SP domain or SP sub-domain.
originatorID	CSE-ID, AE-ID or the resource-ID of a <group> resource that contains the AE or CSE that represents the Originator.
all	Any Originators are allowed to access the resource within the *accessControlOriginators* constraints.
Role-ID	A Role Identifier as defined in clause 7.1.14.

When the *originatorID* is the resource-ID of a <group> resource which contains <AE> or <remoteCSE> as member, the Hosting CSE of the resource shall check if the originator of the request matches one of the members in the memberIDs attribute of the <group> resource (e.g. by retrieving the <group> resource). If the <group> resource cannot be retrieved or does not exist, the request shall be rejected.

9.6.2.2 *accessControlContexts*

The *accessControlContexts* is an optional parameter in an access-control-rule-tuple that contains a list, where each element of the list, when present, represents a context that is permitted to use this access control rule. Each request context is described by a set of parameters, where the types of the parameters can vary within the set. Table 9.6.2.2-1 describes the supported types of parameters in *accessControlContexts*.

The following Originator *accessControlContexts* shall be considered for access control policy check by the CSE.

Table 9.6.2.2-1: Types of Parameters in *accessControlContexts*

Name	Description
accessControlTimeWindow	Represents a time window constraint which is compared against the time that the request is received at the Hosting CSE.
accessControlLocationRegion	Represents a location region constraint which is compared against the location of the Originator of the request.
accessControlIpIPAddress	Represents an IP address constraint or IP address block constraint which is compared against the IP address of the Originator of the request.

9.6.2.3 *accessControlOperations*

The *accessControlOperations* is a mandatory parameter in an access-control-rule-tuple that represents the set of operations that are authorized using this access control rule. Table 9.6.2.3-1 describes the supported set of operations that are authorized by *accessControlOperations*.

The following *accessControlOperations* shall be considered for access control policy check by the CSE.

Table 9.6.2.3-1: Types of parameters in *accessControlOperations*

Name	Description
RETRIEVE	Privilege to retrieve the content of an addressed resource
CREATE	Privilege to create a child resource
UPDATE	Privilege to update the content of an addressed resource
DELETE	Privilege to delete an addressed resource
DISCOVER	Privilege to discover the resource
NOTIFY	Privilege to receive a notification

9.6.2.4 accessControlObjectDetails

The *accessControlObjectDetails* is an optional parameter of an access control rule. It specifies a subset of child resource types of the targeted resource to which the access control rule applies. If an access control rule includes *accessControlObjectDetails*, then *childResourceType* shall be specified. An access control rule which does not include any *accessControlObjectDetails* parameters applies to the child resource types of the target resource. The *accessControlObjectDetails* parameter shall consist of the elements listed in table 9.6.2.4-1. Child resource types listed in the *childResourceType* component are subject of access control for the Create operation only. Once a child resource is created, the Access Control Policies assigned directly to it apply. The *resourceType* and *specialization* element are optional. If either the *resourceType* or *specialization* element is present in *accessControlObjectDetails*, the CSE shall match the type of resource or specialization of the targeted resource with the value specified in the *resourceType* or *specialization* element. Further checking of *childResourceType* shall be done only if the *resourceType* or *specialization* match occurs. However, if the *resourceType* and *specialization* elements are not provided, only *childResourceType* match shall be performed.

Table 9.6.2.4-1: Types of Parameters in *accessControlObjectDetails*

Name	Description
resourceType	Identifier of the resource type to which this access control rule applies.
specialization	When the *resourceType* is *mgmtObj* or *flexContainer*, the identifier of the specialization as defined by *mgmtDefinition* or *containerDefinition* attribute, respectively, shall be specified.
childResourceType	List of child resource types and/or the identifier of the specialization. The identifier of the specialization shall be specified when the *resourceType* is *mgmtObj* or *flexContainer*.

9.6.2.5 *accessControlAuthenticationFlag*

The *accessControlAuthenticationFlag* is an optional parameter in an access-control-rule-tuple: if the value is TRUE, then the access control rule applies only if the Originator is considered to be authenticated by the Hosting CSE; if the value is FALSE, then the access control rule applies whether or not the Originator is considered to be authenticated by the Hosting CSE. Clause 7.1.2 in ETSI TS 118 103 [2] describes the criteria used to determine if the Originator is considered to be authenticated by the Hosting CSE.

If the *accessControlAuthenticationFlag* parameter is not present, then the value is assumed to be FALSE.

9.6.3 Resource Type *CSEBase*

A <*CSEBase*> resource shall represent a CSE. The <*CSEBase*> resource shall be the root for all resources that are residing in the CSE. A CSE shall be represented by only one <*CSEBase*> resource.

The <*CSEBase*> resource shall contain the child resources specified in table 9.6.3-1.

Table 9.6.3-1: Child resources of <*CSEBase*> resource

Child Resources of <*CSEBase*>	Child Resource Type	Multiplicity	Description
[variable]	*<CSEBaseAnnc>*	0..n	Announced variant of <CSEBase>. Resource with CSE-specific information for a CSE that intends to announce resources to another CSE
[variable]	*<remoteCSE>*	0..n	See clause 9.6.4
[variable]	*<remoteCSEAnnc>*	0..n	Announced variant of *<remoteCSE>*. Resource with CSE-specific information for a CSE that announced itself to another CSE with which it does not have a registration relationship
[variable]	*<node>*	0..n	See clause 9.6.18
[variable]	*<AE>*	0..n	See clause 9.6.5
[variable]	*<container>*	0..n	See clause 9.6.6
[variable]	*<flexContainer>*	0..n	See clause 9.6.35
[variable]	*<group>*	0..n	See clause 9.6.13
[variable]	*<accessControlPolicy>*	0..n	See clause 9.6.2
[variable]	*<subscription>*	0..n	See clause 9.6.8
[variable]	*<mgmtCmd>*	0..n	See clause 9.6.16
[variable]	*<locationPolicy>*	0..n	See clause 9.6.10
[variable]	*<statsConfig>*	0..n	See clause 9.6.23
[variable]	*<statsCollect>*	0..n	See clause 9.6.25
[variable]	*<request>*	0..n	See clause 9.6.12
[variable]	*<delivery>*	0..n	See clause 9.6.11
[variable]	*<schedule>*	0..1	This resource defines the reachability schedule information of the entity. The absence of this resource implies the entity is always reachable. See clause 9.6.9
[variable]	*<role>*	0..n	See clause 9.6.38
[variable]	*<token>*	0..n	See clause 9.6.39
[variable]	*<m2mServiceSubscriptionProfile>*	0..n	See clause 9.6.19
[variable]	*<serviceSubscribedAppRule>*	0..n	See clause 9.6.29
[variable]	*<notificationTargetPolicy>*	0..n	See clause 9.6.32
[variable]	*<dynamicAuthorizationConsultation>*	0..n	See clause 9.6.40
[variable]	*<timeSeries>*	0..n	See clause 9.6.36
[variable]	*<authorizationDecision>*	0..n	See clause 9.6.41
[variable]	*<authorizationPolicy>*	0..n	See clause 9.6.42
[variable]	*<authorizationInformation>*	0..n	See clause 9.6.43
[variable]	*<localMulticastGroup>*	0..n	See clause 9.6.44
[variable]	*<transactionMgmt>*	0..n	See clause 9.6.47
[variable]	*<transaction>*	0..n	See clause 9.6.48
[variable]	*<ontologyRepository>*	0..1	See clause 9.6.50
[variable]	*<semanticMashupJobProfile>*	0..n	See clause 9.6.53
[variable]	*<semanticMashupInstance>*	0..n	See clause 9.6.54
[variable]	*<AEContactList>*	0..n	See clause 9.6.45

The <*CSEBase*> resource shall contain the attributes specified in table 9.6.3-2.

Table 9.6.3-2: Attributes of <*CSEBase*> resource

Attributes of <*CSEBase*>	Multiplicity	RW/ RO/ WO	Description	<*CSEBaseAnnc*> Attributes
resourceType	1	RO	See clause 9.6.1.3.	NA
resourceID	1	RO	See clause 9.6.1.3.	NA
resourceName	1	RO	See clause 9.6.1.3.	NA
parentID	1	RO	See clause 9.6.1.3. Shall be an empty string.	NA
creationTime	1	RO	See clause 9.6.1.3.	NA
lastModifiedTime	1	RO	See clause 9.6.1.3.	NA
accessControlPolicyIDs	0..1 (L)	RO	See clause 9.6.1.3.	MA

Attributes of <*CSEBase*>	Multiplicity	RW/ RO/ WO	Description	<*CSEBaseAnnc*> Attributes
labels	0..1 (L)	RO	See clause 9.6.1.3.	MA
announceTo	0..1 (L)	RO	See clause 9.6.1.3.	NA
announcedAttribute	0..1 (L)	RO	See clause 9.6.1.3.	NA
dynamicAuthorizationConsultationIDs	0..1 (L)	RO	See clause 9.6.1.3.	OA
owner	0..1	RW	See clause 9.6.1.3	NA
location	0..1	RW	See clause 9.6.1.3.	OA
cseType	0..1	RO	Indicates the type of CSE represented by the created resource: • Mandatory for an IN-CSE, hence multiplicity (1). • Its presence is subject to SP configuration in case of an ASN-CSE or a MN-CSE.	OA
CSE-ID	1	RO	The CSE identifier in SP-relative CSE-ID format (clause 7.2).	OA
supportedResourceType	1 (L)	RO	List of the resource types which are supported in the CSE. This attribute contains subset of resource types listed in clause 9.2.	NA
pointOfAccess	1 (L)	RO	Represents the list of physical addresses to be used by remote CSEs to connect to this CSE (e.g. IP address, FQDN). This attribute is exposed to its Registree.	OA
nodeLink	0..1	RO	The *resource identifier* of a <*node*> resource that stores the node specific information of the node on which the CSE represented by this <*CSEBase*> resource resides.	OA
notificationCongestionPolicy	0..1	RO	This attribute applies to CSEs generating subscription notifications. It specifies the rule which is applied when the storage of notifications for each subscriber (an AE or CSE) reaches the maximum storage limit for notifications for that subscriber. E.g. Delete stored notifications of lower *notificationStoragePriority* to make space for new notifications of higher *notificationStoragePriority*, or delete stored notifications of older *creationTime* to make space for new notifications when all notifications are of the same *notificationStoragePriority*.	OA
contentSerialization	0..1 (L)	RO	The list of supported serializations of the ***Content*** primitive parameter for receiving a request from its registrants. (e.g. XML, JSON). The list shall be ordered so that the most preferred format comes first.	OA
e2eSecInfo	0..1	RO	See clause 9.6.1.3.	MA
supportedReleaseVersions	0..1 (L)	RO	List of oneM2M release versions which are supported by the CSE. Starting with Release 2, this attribute is mandatory for a CSE. For CSEs compliant to older releases, this attribute is optional. For CSEs that do not include this attribute, the default release version shall be Release 1.	MA
currentTime	0..1	RO	When the CSE receives a retrieve request targeting this resource or attribute, the CSE samples its current time (e.g. makes an OS call to get the system time) and respond with the value in this attribute. An Originator retrieving this attribute can use this time value to adjust and synchronize its time value to the time value of this CSE.	OA

9.6.4 Resource Type *remoteCSE*

A <*remoteCSE*> resource shall represent a Registree CSE that is registered to the Registrar CSE. <*remoteCSE*> resources shall be located directly under the <*CSEBase*> resource of Registrar CSE.

Similarly <*remoteCSE*> resource shall also represent a Registrar CSE. <*remoteCSE*> resource shall be located directly under the <*CSEBase*> resource of Registree CSE.

For example, when CSE1 (Registree CSE) registers with CSE2 (Registrar CSE), there will be two <*remoteCSE*> resources created: one in CSE1: <*CSEBase1*>/<*remoteCSE2*> and one in CSE2: <*CSEBase2*>/<*remoteCSE1*>.

Note that the creation of the two resources does not imply mutual registration. The <*CSEBase1*>/<*remoteCSE2*> does not mean CSE2 registered with CSE1 in the example above.

The <*remoteCSE*> resource shall contain the child resources specified in table 9.6.4-1.

Table 9.6.4-1: Child resources of *<remoteCSE>* resource

Child Resources of *<remoteCSE>*	Child Resource Type	Multiplicity	Description	*<remoteCSEAnnc>* Child Resource Types
[variable]	*<container>*	0..n	See clause 9.6.6	*<container>*
[variable]	*<containerAnnc>*	0..n	Announced variant of *<container>.* See clause 9.6.6	*<containerAnnc>*
[variable]	*<flexContainer>*	0..n	See clause 9.6.35	*<flexContainer>*
[variable]	*<flexContaineAnnc>*	0..n	Announced variant of <flexC*ontainer>.* See clause 9.6.35	*<flexContainerAnnc>*
[variable]	*<group>*	0..n	See clause 9.6.13	*<group>*
[variable]	*<groupAnnc>*	0..n	Announced variant of *<group>.* See clause 9.6.13	*<groupAnnc>*
[variable]	*<accessControlPolicy>*	0..n	See clause 9.6.2	*<accessControlPolicy>*
[variable]	*<accessControlPolicyAnnc>*	0..n	Announced variant of *<accessControlPolicy>.* See clause 9.6.2	*<accessControlPolicyAnnc>*
[variable]	*<subscription>*	0..n	See clause 9.6.8	*<subscription>*
[variable]	*<pollingChannel>*	0..1	See clause 9.6.21. If *requestReachability* is FALSE, the CSE that created this *<remoteCSE>* resource should create a *<pollingChannel>* resource and perform long polling. The *<pollingChannel>* shall be utilized by the parent resource	*None*
[variable]	*<nodeAnnc>*	0..n	Announced variant of *<node>.* This announced resource is associated with a <node> resource that is hosted on a CSE which is represented by the parent *<remoteCSE>* or *<remoteCSEAnnc>* resource. See clause 9.6.18 for *<node>*	*<nodeAnnc>*
[variable]	*<dynamicAuthorizationConsultation>*	0..n	See clause 9.6.40	
[variable]	*<timeSeries>*	0..n	See clause 9.6.36	*<timeSeries>*
[variable]	*<timeSeriesAnnc>*	0..n	Announced variant of *<timeSeries>.* See clause 9.6.36	*<timeSeriesAnnc>*
[variable]	*<AEAnnc>*	0..n	Announced variant of *<AE>.* See clause 9.6.5	*<AEAnnc>*
[variable]	*<locationPolicyAnnc>*	0..n	Announced variant of *<locationPolicy>.* See clause 9.6.10	*<locationPolicyAnnc>*
[variable]	*<transactionMgmt>*	0..n	See clause 9.6.47	*<transactionMgmt>*
[variable]	*<transaction>*	0..n	See clause 9.6.48	*<transaction>*

Child Resources of *<remoteCSE>*	Child Resource Type	Multiplicity	Description	*<remoteCSEAnnc>* Child Resource Types
[variable]	*<ontologyRepositoryAnnc>*	0..1	Announced variant of *<ontologyRepository>.* See clause 9.6.50	*<ontologyRepositoryAnnc>*
[variable]	*<semanticMashupJobProfile>*	0..n	See clause 9.6.53	*<semanticMashupJobProfile>*
[variable]	*<semanticMashupJobProfileAnnc>*	0..n	Announced variant of *<semanticMashupJobProfile>.* See clause 9.6.53	*<semanticMashupJobProfileAnnc>*
[variable]	*<semanticMashupInstance>*	0..n	See clause 9.6.54	*<semanticMashupInstance>*
[variable]	*<semanticMashupInstanceAnnc>*	0..n	Announced variant of *<semanticMashupInstance>.* See clause 9.6.54.	*<semanticMashupInstanceAnnc>*

The <remoteCSE> resource shall contain the attributes specified in table 9.6.4-2.

Table 9.6.4-2: Attributes of *<remoteCSE>* resource

Attributes of *<remoteCSE>*	Multiplicity	RW/ RO/ WO	Description	*<remoteCSEAnnc>* Attributes
resourceType	1	RO	See clause 9.6.1.3.	NA
resourceID	1	RO	See clause 9.6.1.3.	NA
resourceName	1	WO	See clause 9.6.1.3.	NA
parentID	1	RO	See clause 9.6.1.3.	NA
creationTime	1	RO	See clause 9.6.1.3.	NA
lastModifiedTime	1	RO	See clause 9.6.1.3.	NA
expirationTime	1	RW	See clause 9.6.1.3.	MA
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.	MA
labels	0..1 (L)	RW	See clause 9.6.1.3.	MA
announceTo	0..1 (L)	RW	See clause 9.6.1.3.	NA
announcedAttribute	0..1 (L)	RW	See clause 9.6.1.3.	NA
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.	OA
cseType	0..1	WO	Indicates the type of CSE represented by the created resource. • Mandatory for an IN-CSE, hence multiplicity (1). • Its presence is subject to SP configuration in case of an ASN-CSE or a MN-CSE.	OA
pointOfAccess	0..1 (L)	RW	For request-reachable remote CSE it represents the list of physical addresses to be used to connect to it (e.g. IP address, FQDN). If this information is not provided and <pollingChannel> resource does exist, the CSE should use *<pollingChannel>* resource. Then the Hosting CSE can forward a request to the CSE without using the PoA.	OA
CSEBase	1	WO	The address of the *<CSEBase>* resource represented by this *<remoteCSE>* resource.	OA
CSE-ID	1	WO	The CSE identifier of the remote CSE represented by this *<remoteCSE>* resource in SP-relative CSE-ID format (clause 7.2).	OA

Attributes of *<remoteCSE>*	Multiplicity	RW/ RO/ WO	Description	*<remoteCSEAnnc>* Attributes
M2M-Ext-ID	0..1	RW	Supported when Registrar is IN-CSE. See clause 7.1.8 where this attribute is described. This attribute is used only for the case of dynamic association of M2M-Ext-ID and CSE-ID.	NA
Trigger-Recipient-ID	0..1	RW	Supported when Registrar is IN-CSE. See clause 7.1.10 where this attribute is described. This attribute is used only for the case of dynamic association of M2M-Ext-ID and CSE-ID.	NA
requestReachability	1	RW	This attribute is an indication of static capability of the CSE that created this *<remoteCSE>* resource. If the CSE can receive requests originated at or forwarded by its registrar CSE, this attribute is set to "TRUE" otherwise "FALSE" (see note 1).	OA
nodeLink	0..1	RW	The *resource identifier* of a *<node>* resource that stores the node specific information of the node on which the CSE represented by this *<remoteCSE>* resource resides.	OA
contentSerialization	0..1 (L)	RW	The list of supported serializations of the ***Content*** primitive parameter for receiving a request (e.g. XML, JSON). The list shall be ordered so that the most preferred format comes first.	OA
e2eSecInfo	0..1	RW	See clause 9.6.1.3.	MA
triggerReferenceNumber	0..1	RW	This is to identify device trigger procedure request. This attribute is used only for device trigger and assigned by the IN-CSE.	NA
descendantCSEs	0..1(L)	RW	This attribute contains a list of identifiers of descendent CSEs of the Registree CSE represented by this <remoteCSE> resource. A descendant CSE is a CSE that either registers to the CSE represented by this <remoteCSE>, or registers to another CSE which is a descendant CSE of this <remoteCSE>. The Registree CSE represented by this <remoteCSE> shall configure this attribute with a list of descendent CSEs upon creation of the <remoteCSE> resource. The Registree CSE shall update this attribute whenever a new descendent CSE either registers or de-registers. The Registree CSE shall detect when a descendent CSE registers or de-registers by monitoring its <remoteCSE> resources and the descendent CSEs attribute(s) of these <remoteCSE> resources. For a <remoteCSE> resource representing a Registrar CSE this attribute shall not be set.	OA
multicastCapability	0..1	RW	Indicates the oneM2M node multicast Capability, pre-defined values are: • MBMS • IP	OA

Attributes of <*remoteCSE*>	Multiplicity	RW/ RO/ WO	Description	<*remoteCSEAnnc*> Attributes
externalGroupID	0..1	RW	Supported when Registrar CSE is an IN-CSE. It is used by an M2M Service Provider (M2M SP) when services targeted to a group of M2M Devices are requested from the Underlying Network. It is assumed to be a globally unique ID exposed by the underlying network to identify a group of M2M Devices (e.g. ASN, MN) for group related services.	OA
triggerEnable	0..1	RW	When set to "TRUE", trigger requests may be sent to the CSE represented by this <*remoteCSE*> resource. When set to "FALSE" trigger requests shall not be sent to this CSE.	OA
activityPatternElements	0..1(L)	RW	This attribute describes the anticipated availability of the CSE for communications. See further description below and table 9.6.4-3.	OA
supportedReleaseVersions	0..1(L)	RW	The oneM2M release versions supported by the CSE represented by this <*remoteCSE*> resource. Starting with Release 2, this attribute is mandatory for a CSE. For CSEs compliant to older releases, this attribute is optional. For CSEs that do not include this attribute, the default release version shall be Release 1.	MA
NOTE-1: Even if this attribute is set to "FALSE", it is not meant that the CSE is always unreachable by its registrees. E.g. if the CSE and its registrees are behind the same NAT, then the CSE can receive requests from its registrees. See also *pollingChannel* description in clause 9.6.21. NOTE-2: For the case of a response, this attribute is applicable if the corresponding request does not contain the serialization format of the *Content* request parameter to allow a CSE to determine the proper serialization format to use in the response.				

The set of activity patterns represented in the *activityPatternElements* attribute describes the anticipated availability of the CSE for communications. The set provides the anticipated activity timing pattern, and may provide additional information about the anticipated mobility status and expected data size to be exchanged. Each *activityPatternElements* item is comprised of triples (*scheduleElement*, *stationaryIndication*, *dataSizeIndicator*) with parameters shown and described in table 9.6.4-3.

Table 9.6.4-3: Parameters in *activityPatternElements* triple

Name	Description
scheduleElement	See clause 9.6.9. This parameter shall be composed from seven fields of second, minute, hour, day of month, month, day of week and year. This is a mandatory parameter in the triple. This parameter indicates the times when the entity is available to send and receive primitives.
stationaryIndication	It indicates the field node as 'Stationary (Stopping)' or 'Mobile (Moving)' for the traffic pattern. The default value is NULL, denoting that no *stationaryIndication* is provided.
dataSizeIndicator	It indicates the expected data size for the traffic pattern. The default value is NULL, denoting that no *dataSizeIndicator* is provided.

9.6.5 Resource Type *AE*

An <*AE*> resource shall represent information about an Application Entity registered to a CSE.

The <*AE*> resource shall contain the child resources specified in table 9.6.5-1.

Table 9.6.5-1: Child resources of <*AE*> resource

Child Resources of <*AE*>	Child Resource Type	Multiplicity	Description	<*AEAnnc*> Child Resource Types
[variable]	<*semanticDescriptor*>	0..n	See clause 9.6.30	<*semanticDescriptor*>, <*semanticDescriptorAnnc*>
[variable]	<*subscription*>	0..n	See clause 9.6.8	<*subscription*>
[variable]	<*container*>	0..n	See clause 9.6.6	<*container*> <*containerAnnc*>
[variable]	<*flexContainer*>	0..n	See clause 9.6.35	<*flexContainer*> <*flexContainerAnnc*>
[variable]	<*group*>	0..n	See clause 9.6.13	<*group*> <*groupAnnc*>
[variable]	<*accessControlPolicy*>	0..n	See clause 9.6.2	<*accessControlPolicy*> <*accessControlPolicyAnnc*>
[variable]	<*pollingChannel*>	0..1	See clause 9.6.21 When the AE is request-unreachable, the AE should create this <*pollingChannel*> resource and perform long polling. The <*pollingChannel*> shall be utilized by the parent resource	*None*
[variable]	<*dynamicAuthorizationConsultation*>	0..n	See clause 9.6.40	*None*
[variable]	<*timeSeries*>	0..n	See clause 9.6.36	<*timeSeries*> <*timeSeriesAnnc*>
[variable]	<*transactionMgmt*>	0..n	See clause 9.6.47	<*transactionMgmt*>
[variable]	<*transaction*>	0..n	See clause 9.6.48	<*transaction*>
[variable]	<*triggerRequest*>	0..n	See clause 9.6.49	*None*
[variable]	<*multimediaSession*>	0..n	See Clause 9.6.57. This resources holds information describing the established multimedia session	*None*
[variable]	<*semanticMashupInstance*>	0..n	See clause 9.6.54	<*semanticMashupInstance*> <*semanticMashupInstanceAnnc*>

The <*AE*> resource shall contain the attributes specified in table 9.6.5-2.

Table 9.6.5-2: Attributes of <*AE*> resource

Attributes of <*AE*>	Multiplicity	RW/ RO/ WO	Description	<*AEAnnc*> Attributes
resourceType	1	RO	See clause 9.6.1.3.	NA
resourceID	1	RO	See clause 9.6.1.3. Contains the AE-ID-Stem of the AE (see clause 7.2 on identifier formats and clause 10.2.2.2 for AE registration procedure).	NA
resourceName	1	WO	See clause 9.6.1.3.	NA
parentID	1	RO	See clause 9.6.1.3.	NA
expirationTime	1	RW	See clause 9.6.1.3.	MA
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.	MA
creationTime	1	RO	See clause 9.6.1.3.	NA
lastModifiedTime	1	RO	See clause 9.6.1.3.	NA
labels	0..1 (L)	RW	See clause 9.6.1.3.	MA
announceTo	0..1 (L)	RW	See clause 9.6.1.3.	NA
announcedAttribute	0..1 (L)	RW	See clause 9.6.1.3.	NA
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.	OA
appName	0..1	RW	The name of the application, as declared by the application developer (e.g. "HeatingMonitoring"). Several sibling resources may share the *appName*.	OA

Attributes of <*AE*>	Multiplicity	RW/ RO/ WO	Description	<*AEAnnc*> Attributes
App-ID	1	WO	The identifier of the Application (see clause 7.1.3).	OA
AE-ID	1	RO	The identifier of the Application Entity (see clause 7.1.2).	OA
M2M-Ext-ID	0..1	RW	Supported when Registrar is IN-CSE. See clause 7.1.8 where this attribute is described. This attribute is used only for the case of dynamic association of M2M-Ext-ID and AE-ID.	NA
trigger-Recipient-ID	0..1	RW	Supported when Registrar is IN-CSE. See clause 7.1.10 where this attribute is described. This attribute is used only for the case of dynamic association of M2M-Ext-ID and AE-ID.	NA
triggerReferenceNumber	0..1	RW	This is to identify device trigger procedure request. This attribute is used only for device trigger and assigned by the IN-CSE.	NA
pointOfAccess	0..1 (L)	RW	The list of addresses for communicating with the registered Application Entity over Mca reference point via the transport services provided by Underlying Network (e.g. IP address, FQDN, URI). This attribute shall be accessible only by the AE and the Hosting CSE. If this information is not provided and the <pollingChannel> resource does exist, the AE should use <*pollingChannel*> resource. Then the Hosting CSE can forward a request to the AE without using the PoA.	OA
registrationStatus	0..1	RW	Denotes status of the AE registration. If ACTIVE, the <AE> resource and all its child resources may be discoverable. If INACTIVE, the <AE> resource and all its child resources shall not be discoverable. Set to ACTIVE during a AE registration or re-registration. When an AE changes its registration point, the registration at the old registration point is set to INACTIVE.	OA
trackRegistrationPoints	0..1	RW	Denotes if the Application Entity requests that its Registration Points be tracked. If TRUE, AE requests to be tracked as it changes its Registration Points. If FALSE, the AE requests not to be tracked as it changes its Registration Points.	OA
ontologyRef	0..1	RW	A URI of the ontology used to represent the information that is managed and understood by the AE.	OA
requestReachability	1	RW	This attribute is an indication of static capability of the AE that created this <*AE*> resource. If the AE can receive requests originated at or forwarded by its registrar CSE, this attribute is set to "TRUE" otherwise "FALSE".	OA
nodeLink	0..1	RW	The *resource identifier* of a <*node*> resource that stores the node specific information of the node on which the AE represented by this <*AE*> resource resides.	OA
contentSerialization	0..1 (L)	RW	The list of supported serializations of the ***Content*** primitive parameter for receiving a request and a response from its registrar CSE. (e.g. XML, JSON, CBOR). The list shall be ordered so that the most preferred format comes first.	OA
e2eSecInfo	0..1	RW	See clause 9.6.1.3.	MA

Attributes of <*AE*>	Multiplicity	RW/ RO/ WO	Description	<*AEAnnc*> Attributes
activityPatternElements	0..1(L)	RW	This attribute describes the anticipated availability of the AE for communications. See further description below and table 9.6.4-3.	OA
triggerEnable	0..1	RW	When set to "TRUE", trigger requests may be sent to the AE represented by this <*AE*> resource. When set to "FALSE" trigger requests shall not be sent to this AE.	OA
sessionCapabilities	0..1 (L)	RW	The list of supported session media types (e.g. audio, video, image) and supported session protocols (e.g. RTP, RTP/AVP) as defined by session parameters as defined by the IETF IANA Session Descriptor Protocol (SDP) Parameter Registry.	OA
supportedReleaseVersions	0..1(L)	RW	The oneM2M release versions supported by the Registree AE represented by this <*AE*> resource. Starting with Release 2, this attribute is mandatory for an AE. For AEs compliant to older releases, this attribute is optional. For AEs that do not include this attribute, the default release version shall be Release 1.	MA
NOTE: For the case of a response, this attribute is applicable if the corresponding request does not contain the serialization format of the *Content* request parameter to allow a CSE to determine the proper serialization format to use in the response.				

The set of activity patterns represented in the *activityPatternElements* attribute describes the anticipated availability of the AE for communications. The set provides the anticipated activity timing pattern and might provide additional information about the anticipated mobility status and expected data size to be exchanged. Each *activityPatternElements* item is comprised of triples (*scheduleElement*, *stationaryIndication*, *datasizeIndicator*) with parameters shown and described in table 9.6.4-3.

9.6.6 Resource Type *container*

The <*container*> resource represents a container for data instances. It is used to share information with other entities and potentially to track the data. A <*container*> resource has no associated content. It has only attributes and child resources.

The <*container*> resource shall contain the child resources specified in table 9.6.6-1.

Table 9.6.6-1: Child resources of <*container*> resource

Child Resources of <*container*>	Child Resource Type	Multiplicity	Description	<*containerAnnc*> Child Resource Types
[variable]	<*semanticDescriptor*>	0..n	See clause 9.6.30	<*semanticDescriptor*>, <*semanticDescriptorAnnc*>
[variable]	<*contentInstance*>	0..n	See clause 9.6.7	<*contentInstance*>, <*contentInstanceAnnc*>
[variable]	<*subscription*>	0..n	See clause 9.6.8	<*subscription*>
[variable]	<*container*>	0..n	See clause 9.6.6	<*container*> <*containerAnnc*>
[variable]	<*flexContainer*>	0..n	See clause 9.6.35	<*flexContainer*> <*flexContainerAnnc*>
[variable]	<*timeSeries*>	0..n	See clause 9.6.36	<*timeSeries*>, <*timeSeriesAnnc*>
la	<*latest*>	1	See clause 9.6.27	*None*
ol	<*oldest*>	1	See clause 9.6.28	*None*
[variable]	<*transaction*>	0..n	See clause 9.6.48	<*transaction*>

The <*container*> resource shall contain the attributes specified in table 9.6.6-2.

Table 9.6.6-2: Attribute of <*container*> resource

Attributes of <*container*>	Multiplicity	RW/ RO/ WO	Description	<*containerAnnc*> Attributes
resourceType	1	RO	See clause 9.6.1.3.	NA
resourceID	1	RO	See clause 9.6.1.3.	NA
resourceName	1	WO	See clause 9.6.1.3.	NA
parentID	1	RO	See clause 9.6.1.3.	NA
expirationTime	1	RW	See clause 9.6.1.3.	MA
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.	MA
labels	0..1 (L)	RW	See clause 9.6.1.3.	MA
creationTime	1	RO	See clause 9.6.1.3.	NA
lastModifiedTime	1	RO	See clause 9.6.1.3.	NA
stateTag	1	RO	See clause 9.6.1.3.	NA
announceTo	0..1 (L)	RW	See clause 9.6.1.3.	NA
announcedAttribute	0..1 (L)	RW	See clause 9.6.1.3.	NA
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.	OA
creator	0..1	RO	See clause 9.6.1.3.	NA
maxNrOfInstances	0..1	RW	Maximum number of direct child <*contentInstance*> resources in the <*container*> resource.	OA
maxByteSize	0..1	RW	Maximum size in bytes of data (i.e. *content* attribute of a <*contentInstance*> resource) that is allocated for the <*container*> resource for all direct child <*contentInstance*> resources in the <*container*> resource.	OA
maxInstanceAge	0..1	RW	Maximum age of a direct child <*contentInstance*> resource in the <*container*> *resource*. The value is expressed in seconds.	OA
currentNrOfInstances	1	RO	Current number of direct child <*contentInstance*> resource in the <*container*> resource. It is limited by the *maxNrOfInstances*. The *currentNrOfInstances* attribute of the <container> resource shall be updated on successful creation or deletion of direct child <contentInstance> resource of <container> resource.	NA
currentByteSize	1	RO	Current size in bytes of data (i.e. *content* attribute of a <*contentInstance*> resource) stored in all direct child <*contentInstance*> resources of a <*container*> resource. This is the summation of *contentSize* attribute values of the <*contentInstance*> resources. It is limited by the*maxByteSize*. The *currentByteSize* attribute of the <container> resource shall be updated on successful creation of deletion of direct child <contentInstance> resource of <container> resource.	NA
locationID	0..1	RO	An ID of the resource where the attributes/policies that define how location information are obtained and managed. This attribute is defined only when the <*container*> resource is used for containing location information.	OA
ontologyRef	0..1	RW	A reference (URI) of the ontology used to represent the information that is stored in the child <*contentInstance*> resources of the present <*container*> resource (see note).	OA

Attributes of <*container*>	Multiplicity	RW/ RO/ WO	Description	<*containerAnnc*> Attributes
disableRetrieval	0..1	RW	Boolean value to control RETRIE/UPDATE/DELETE operation on the child <*contentInstance*> resource. When the value is set to 'TRUE', RETRIEVE/DELETE/UPDATE operations for child <contentInstance> shall be rejected at all times. When the value is updated from 'TRUE' to 'FALSE', all existing <contentInstance> are deleted immediately. When the value is set to 'FALSE', all operations are permitted on the <contentInstance> resource as per existing procedures.	OA
NOTE: The access to this URI is out of scope of oneM2M.				

9.6.7 Resource Type *contentInstance*

The <*contentInstance*> resource represents a data instance in the <*container*> resource. The content of the *contentInstance* can be encrypted.

Unlike other resources, the <*contentInstance*> resource shall not be modified once created. This pertains to its attributes, but not to the creation of child resources. An AE shall be able to delete a *contentInstance* resource explicitly or it may be deleted by the platform based on policies. If the platform has policies for *contentInstance* retention, these shall be represented by the attributes *maxByteSize*, *maxNrOfInstances* and/or *maxInstanceAge* attributes in the <*container*> resource. If multiple policies are in effect, the strictest policy shall apply.

The <*contentInstance*> resource inherits the same access control policies of the parent <*container*> resource, and does not have its own *accessControlPolicyIDs* attribute.

The <*contentInstance*> resource shall contain the child resources specified in table 9.6.7-1.

Table 9.6.7-1: Child resources of <*contentInstance*> resource

Child Resources of <*contentInstance*>	Child Resource Type	Multiplicity	Description	<*contentInstanceAnnc*> Child Resource Types
[variable]	<*semanticDescriptor*>	0..n	See clause 9.6.30	<*semanticDescriptor*>, <*semanticDescriptorAnnc*>
[variable]	<*transaction*>	0..n	See clause 9.6.48	<*transaction*>

The <contentInstance> resource shall contain the attributes specified in table 9.6.7-2.

Table 9.6.7-2: Attributes of <*contentInstance*> resource

Attributes of <*contentInstance*>	Multiplicity	RW/ RO/ WO	Description	<*contentInstance Annc*> Attributes
resourceType	1	RO	See clause 9.6.1.3.	NA
resourceID	1	RO	See clause 9.6.1.3.	NA
resourceName	1	WO	See clause 9.6.1.3.	NA
parentID	1	RO	See clause 9.6.1.3.	NA
labels	0..1 (L)	WO	See clause 9.6.1.3.	MA
expirationTime	1	WO	See clause 9.6.1.3.	NA
creationTime	1	RO	See clause 9.6.1.3.	NA
lastModifiedTime	1	RO	See clause 9.6.1.3.	NA

Attributes of <*contentInstance*>	Multiplicity	RW/ RO/ WO	Description	<*contentInstance Annc*> Attributes
stateTag	1	RO	See clause 9.6.1.3. The *stateTag* attribute of the parent resource should be incremented first and copied into this *stateTag* attribute when a new instance is added to the parent resource.	NA
announceTo	0..1 (L)	WO	See clause 9.6.1.3.	NA
announcedAttribute	0..1 (L)	WO	See clause 9.6.1.3.	NA
creator	0..1	RO	See clause 9.6.1.3.	NA
contentInfo	0..1	WO	This attribute contains information to understand the content of *content* attribute. It shall be composed of two mandatory components consisting of Internet Media Type (as defined in the IETF RFC 6838 [i.36]) and an encoding type. In addition, an optional content security component may also be included. The format of this attribute is defined in ETSI TS 118 104 [3]. This attribute should be used so that AEs can understand the content. If the value of *contentInfo* is a supported representation of semantic information, as defined in ETSI TS 118 104 [3], the value of *content* shall be handled as semantic information with respect to the supported semantic oneM2M functionalities.	OA
contentSize	1	RO	Size in bytes of the *content* attribute.	NA
contentRef	0..1	WO	This attribute contains a list of name-value pairs. Each entry expresses and associative reference to a <contentInstance> resource. The name of the entry indicates the relationship and the value of the entry the indicates reference (URI) to the resource.	OA
ontologyRef	0..1	WO	A reference (URI) of the ontology used to represent the information that is stored in the *contentInstances* resources of the <*container*> *resource*. If this attribute is not present, the *contentInstance* resource inherits the *ontologyRef* from the parent <*container*> resource if present (see note).	OA
content	1	WO	Actual content of a *contentInstance*. This content may be opaque data for understandable with the help of the *contentInfo*. This may, for example, be an image taken by a security camera, or a temperature measurement taken by a temperature sensor.	OA
NOTE: Access to this URI is out of scope of oneM2M.				

9.6.8 Resource Type *subscription*

The <*subscription*> resource contains subscription information for its subscribed-to resource.

A subscription to a resource allows an entity in the oneM2M architecture to be notified about changes of the subscribed-to resource. The <*subscription*> resource shall represent a subscription to a subscribed-to resource. In order to establish a subscription, a <*subscription*> resource shall be created as a child resource of the subscribed-to resource. The <*subscription*> child resource contains information about the exact scope of the subscription and targets to be notified. For example, a <*container*> resource having a <*subscription*> resource as a child resource (see clause 9.6.6) shall result in notification(s) of target(s) configured in the <*subscription*> child resource when changes to the parent <*container*> resource matching with notification event criteria described by the child <*subscription*> resource occur. A <*subscription*> resource shall be deleted when the parent subscribed-to resource is deleted.

In general, an Originator shall be able to create a resource of <*subscription*> resource type when the Originator has RETRIEVE privilege to the subscribed-to resource. The Originator which creates a <*subscription*> resource becomes the resource subscriber.

A <subscription> resource can be configured to implement a blocking "UPDATE" to a resource or attributes of a resource whereby a notification is sent to the notification target to respond with the result of the "UPDATE" request.

Each <*subscription*> may include notification policies that specify which, when, and how notifications are sent. These notification policies may work in conjunction with CMDH policies.

When a <*subscription*> resource is deleted, a Notify request shall be sent to the target indicated by the attribute *subscriberURI* if it is provided by the Subscriber.

The <*subscription*> resource shall contain the child resources specified in table 9.6.8-1.

Table 9.6.8-1: Child resources of <*subscription*> resource

Child Resources of <*subscription*>	Child Resource Type	Multiplicity	Description
notificationSchedule	<*schedule*>	0..1	In the context of the <*subscription*> resource, the *notificationSchedule* specifies when notifications may be sent by the Hosting CSE to the *notificationURI(s).* See clause 9.6.9.
[variable]	<*notificationTargetMgmtPolicyRef*>	0..n	See 9.6.31 for this type of resource.
nstr	<*notificationTargetSelfReference*>	1	See 9.6.34 for this type of resource.
[variable]	<*transaction*>	0..n	See clause 9.6.48.

The <*subscription*> resource shall contain the attributes specified in table 9.6.8-2.

Table 9.6.8-2: Attributes of <*subscription*> resource

Attributes of <*subscription*>	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
labels	0..1 (L)	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.
creator	0..1	WO	See clause 9.6.1.3.

Attributes of <*subscription*>	Multiplicity	RW/ RO/ WO	Description
eventNotificationCriteria	0..1	RW	This attribute (notification policy) indicates the event criteria for which a notification is to be generated. When no *eventNotificationCriteria* attribute is present in a <*subscription*> resource, the Hosting CSE shall trigger notifications for this subscription when any of the attributes of the subscribed-to resource is modified.
expirationCounter	0..1	RW	This attribute (notification policy) indicates that the subscriber wants to set the life of this subscription to a limit of a maximum number of notifications. When the number of notifications sent reaches the count of this counter, the <*subscription*> resource shall be deleted, regardless of any other policy.
notificationURI	1 (L)	RW	This attribute shall be configured as a list consisting of one or more targets that the Hosting CSE shall send notifications to. A target shall be formatted as a oneM2M compliant Resource-ID as defined in clause 7.2 or as an identifier compliant with a oneM2M supported protocol binding (e.g. http, coap, mqtt). If a target is formatted as a oneM2M compliant Resource-ID, then the target shall be formatted as a structured or unstructured CSE-Relative-Resource-ID, SP-Relative-Resource-ID, and/or Absolute-Resource-ID of an <*AE*> or <CSEBase> resource. A Hosting CSE shall use this information to determine proper pointOfAccess, requestReachability and/or pollingChannel information needed to send a notification to the target. The following is an example. • /CSE0001/AE0001 For a target that is formatted as an identifier compliant with a oneM2M supported protocol binding, the details of this format are defined by the respective oneM2M protocol specification. The following is an example of an HTTP URI compliant with oneM2M HTTP protocol binding. • https://172.25.30.25:7000/notification/handler For a subscription to a <fanoutpoint> resource, if <subscription> resource in request contains a notificationForwardingURI, then the group hosting CSE shall configure the *notificationURI* of the fanout subscription request with an address specified by the Group Hosting CSE that can be used by the Group Hosting CSE to receive aggregated notifications. A notification serialization type may be appended to each notification target configured in this list. The Hosting CSE shall serialize notifications and send it to a notification target based on this serialization type indicator. Possible serialization types are defined in ETSI TS 118 104 [3] (e.g. XML, JSON or CBOR). If a notification serialization type is not appended to a notification target, a default shall apply based on the Hosting CSE local policy. The syntax for appending a serialization type to a notification target shall use the "?" delimiter character as shown in the below examples. • http://mydomain/notificationHandler?ct=json • CSE02/base/ae2?ct=xml
groupID	0..1	RW	The ID of a <*group*> resource in case the subscription is made through a group. This attribute may be used in the ***Filter Criteria*** to discover all subscription resources created via a <fanOutPoint> resource to a specific groupID.
notificationForwardingURI	0..1(L)	RW	The attribute shall be present only for group related subscriptions. If the subscriber intends the Group Hosting CSE to aggregate the notifications, the attribute shall be set identical to the *notificationURI* attribute. It shall be used by Group Hosting CSE for forwarding aggregated notifications. See clauses 10.2.7.10 and 10.2.7.11.

Attributes of <*subscription*>	Multiplicity	RW/ RO/ WO	Description
batchNotify	0..1	RW	This attribute (notification policy) indicates that the subscription originator wants to receive batches of notifications rather than receiving them one at a time. This attribute includes: the number of notifications to be batched for delivery and the duration. When only the number is specified by the subscription originator, the Hosting CSE shall set the default duration given by M2M Service Provider. If *batchNotify* is used simultaneously with *latestNotify*, only the latest notification shall be sent and have the ***Event Category*** set to "latest".
rateLimit	0..1	RW	This attribute (notification policy) indicates that the subscriber wants to limit the rate at which it receives notifications. This attribute expresses the subscriber's notification policy and includes two values: a maximum number of events that may be sent within some duration, and the *rateLimit* window duration. When the number of generated notifications within the *rateLimit* window duration exceeds the maximum number, notification events are temporarily stored, until the end of the window duration, when the sending of notification events restarts in the next window duration. The sending of notification events continues as long as the maximum number of notification events is not exceeded during the window duration. The *rateLimit* policy may be used simultaneously with other notification policies.
preSubscriptionNotify	0..1	WO	This attribute (notification policy) indicates that the subscriber wants to be sent notifications for events that were generated prior to the creation of this subscription. This attribute has a value of the number of prior notification events requested. If up-to-date caching of retained events is supported on the Hosting CSE and contains the subscribed events, then prior notification events will be sent up to the number requested. The *preSubscriptionNotify* policy may be used simultaneously with any other notification policy.
pendingNotification	0..1	RW	This attribute (notification policy), if set, indicates how missed notifications due to a period of no connectivity are handled (according to the reachability and notification schedules). The possible values for *pendingNotification are*: • "sendLatest"; • "sendAllPending". This policy depends upon caching of retained notifications on the hosted CSE. When this attribute is set to "sendLatest", only the last notification shall be sent and it shall have the ***Event Category*** set to "latest". If this attribute is not present, the Hosting CSE sends no missed notifications. This policy applies to all notifications regardless of the selected delivery policy (*batchNotify*, *latestNotify*, etc.) Note that unreachability due to reasons other than scheduling is not covered by this policy.
notificationStoragePriority	0..1	RW	Indicates that the subscriber wants to set a priority for this subscription relative to other subscriptions belonging to this same subscriber. This attribute sets a number within the priority range. When storage of notifications exceeds the allocated size, this policy is used as an input with the storage congestion policy (*notificationCongestionPolicy*) specified in clause 9.6.3 to determine which stored and generated notifications to drop and which ones to retain.
latestNotify	0..1	RW	This attribute (notification policy) indicates if the subscriber wants only the latest notification. If multiple notifications of this subscription are buffered, and if the value of this attribute is set to true, then only the last notification shall be sent and it shall have the ***Event Category*** value set to "latest".

Attributes of <*subscription*>	Multiplicity	RW/ RO/ WO	Description
notificationContentType	1	RW	Indicates a notification content type that shall be contained in notifications. The allowed values are: • "modified attributes"; • "all attributes"; • "ID" of the resource indicated in the *notificationEventType* condition. • Trigger Payload For a list of the default and allowed values of *notificationContentType* for each of the supported values of *notificationEventType* refer to table 9.6.8-4.
notificationEventCat	0..1	RW	This attribute (notification policy) indicates the subscriber's requested ***Event Category*** to be used for notification messages generated by this subscription.
subscriberURI	0..1	WO	This attribute shall be configured with the target of the subscriber. The target is used by the Hosting CSE to determine where to send a notification when the subscription is deleted. A target shall be formatted as a oneM2M compliant Resource-ID as defined in clause 7.2 or as an identifier compliant with one of the oneM2M supported protocol bindings (the detailed format of which are defined by each respective oneM2M protocol binding specification).
associatedCrossResourceSub	0..1	RW	This attribute lists *the identifier of* <*crossResourceSubscription*> resources where this <*subscription*> is involved in.

Table 9.6.8-3 describes the *eventNotificationCriteria* conditions.

Table 9.6.8-3: *eventNotificationCriteria* conditions

Condition tag	Multiplicity	Matching condition
createdBefore	0..1	The *creationTime* attribute of the resource is chronologically before the specified value.
createdAfter	0..1	The *creationTime* attribute of the resource is chronologically after the specified value.
modifiedSince	0..1	The *lastModifiedTime* attribute of the resource is chronologically after the specified value.
unmodifiedSince	0..1	The *lastModifiedTime* attribute of the resource is chronologically before the specified value.
stateTagSmaller	0..1	The *stateTag* attribute of the resource is smaller than the specified value.
stateTagBigger	0..1	The *stateTag* attribute of the resource is bigger than the specified value.
expireBefore	0..1	The *expirationTime* attribute of the resource is chronologically before the specified value.
expireAfter	0..1	The *expirationTime* attribute of the resource is chronologically after the specified value.
sizeAbove	0..1	The *contentSize* attribute of the <*contentInstance*> resource is equal to or greater than the specified value.
sizeBelow	0..1	The *contentSize* attribute of the <*contentInstance*> resource is smaller than the specified value.
notificationEventType	0..6	The type of event that shall trigger a notification. If multiple *notificationEventType* tags are present, a notification shall be triggered if any of the configured events occur. Note that not all permutations of event type are meaningful. Possible notification event type values are: A. Update to attributes of the subscribed-to resource B. Deletion of the subscribed-to resource, C. Creation of a direct child of the subscribed-to resource, D. Deletion of a direct child of the subscribed-to resource

Condition tag	Multiplicity	Matching condition
		E. An attempt to retrieve a <*contentInstance*> direct-child-resource of a subscribed-to <*container*> resource is performed while this <*contentInstance*> child resource is an obsolete resource or the reference used for retrieving this resource is not assigned. This retrieval is performed by a RETRIEVE request targeting the subscribed-to resource with the Result Content parameter set to either "child-resources" or "attributes+child-resources". This value for the *eventNotificationType* tag implies that the subscribed-to resource shall be an <*container*> resource. Otherwise this setting is not valid. F. Trigger Received targeting the MN/ASN-AE associated with the <AE> parent resource. This implies that the subscribed-to resource shall be an <*AE*> resource instance. Otherwise this setting is not valid. G. Update to attributes of the subscribed-to resource with blocking of the triggering UPDATE operation. For this *eventNotificationType* value setting, only one single Notification Target shall be present in the *notificationURI* attribute - see *notificationURI* attribute definition. This value for the *eventNotificationType* tag shall not be combined with any other *eventNotificationType* tag value. This value for *notificationEventType* establishes a subscription that is triggered for the same events as for the value "Update to attributes of the subscribed-to resource". However, upon occurrence of a triggering UPDATE operation that has been validated and results in an authorized UPDATE operation, the triggering UPDATE operation shall be blocked by the Hosting CSE until a notification request was sent out and a corresponding response message was received or a timeout happens. When the response status code of the notification response message indicates a successful notification reception in combination with a successful notification action taken by the Notification Target entity, the triggering UPDATE operation shall be completed with a successful update of the targeted attribute(s). If the notification response message indicates an unsuccessful notification reception or a successful notification reception with unsuccessful notification action by the targeted entity or times out, the blocked UPDATE operation shall be completed with no success and no change of the targeted attribute(s). For any subscribed-to resource there shall exist a maximum of one subscription with this setting of *notificationEventType*. All other notification policies shall not be allowed when this setting of *notificationEventType* is used. The *notificationContentType* shall be "modified attributes". When an UPDATE operation has been blocked due to triggering this type of notification, any other occurring UPDATE or DELETE requests to the same resource shall be handled only after the blocked UPDATE operation has been completed.

Condition tag	Multiplicity	Matching condition
		The other conditions in *eventNotificationCriteria* conditions apply within the scope of the selected *notificationEventType.* For example, if notificationEventType is "Creation of a direct child of the subscribed-to resource" then other *eventNotificationCriteria* conditions is applied to the direct child resources of the subscribed-to resource. If this condition is not specified, the default value is "Update to attributes of the subscribed-to resource". This default value shall apply only if *operationMonitor* is not present in the resource. The notion of "obsolete resource" is defined in clause 9.6.1.3.2 (Common attributes).
operationMonitor	0..n	The operations and/or the Originators accessing the subscribed-to resource matches with the specified value. It allows monitoring which operation and/or which Originator is attempting to the access subscribed-to resource regardless of whether the operation is performed. This feature is useful to detect AEs that send requests to a subscribed-to resource and that result in a successful or failure response. Possible arguments are operation(s) (e.g.: CREATE, RETRIEVE, UPDATE, DELETE, NOTIFY) and/or Originator identifier(s). If a set of Originator identifier(s) is included in this tag and no operations are listed, any operations initiated from any of the indicated Originator(s) shall trigger a notification. If a set of operation(s) is included in this tag and no Originator identifier, any of the listed operations shall trigger a notification. If both, a set of Originator identifiers and a set of operations are listed, then any of the listed operations initiated from any of the listed Originators shall trigger the notification.
attribute	0..n	A list of attribute names of a subscribed-to-resource. This list is only applicable when *notificationEventType* has a value of "Update to attributes of the subscribed-to resource". or "Update to attributes of the subscribed-to resource with blocking of the triggering UPDATE operation". If this list is present, then it is used to specify a subset of a subscribed-to resource's attributes for which updates shall result in a notification. If ANY attribute specified on this list is updated, then a notification shall be generated. If an attribute that is not specified in this list is updated, then a notification shall not be generated. If this list is not presented, then the default attribute list is the full set of a subscribed-to resource's attributes. If ANY attribute of a subscribed-to resource is updated, then a notification shall be generated.
childResourceType	0.. 1 (L)	A list of resource types. This list is only applicable when *notificationEventType* has a value of "Creation of a direct child of the subscribed-to resource" or "Deletion of a direct child of the subscribed-to resource". If this list is present, then it is used to specify a subset of resource type for direct child resource of which creation or deletion shall result in a notification. If ANY resource type specified on this list is created or deleted, then a notification shall be generated. If a resource type that is not specified in this list is created or deleted, then a notification shall not be generated. If this list is not present, then the default resource type list is the full set of a direct child resource.

Condition tag	Multiplicity	Matching condition
missingData	0..1	The *missingData* includes two values: a minimum specified missing number of the Time Series Data within the specified window duration, and the window duration. The condition only applies to subscribed-to resources of type <*timeSeries*>. If this attribute is modified by an UPDATE the associated timer/counter are stopped and restarted with the new values. The first detected missing data point starts the timer associated with the window duration. The window duration is restarted upon its expiry until such time as the entire subscription is terminated or not refreshed. More details about NOTIFICATIONS related to data reporting is found in clause 10.2.4.29.
filterOperation	0..1	Indicates the logical operation (AND/OR) to be used for the condition tags *createdBefore, createdAfter, modifiedSince, unmodifiedSince, stateTagSmaller, stateTagBigger, expireBefore, expireAfter, sizeAbove, sizeBelow*. The default value is logical AND.

The rules when multiple conditions are used together shall be as follows:

- Different condition tags shall use the "AND/OR" logical operation based on the *filterOperation* specified;
- Same condition tags shall use the "OR" logical operation.

No mixed AND/OR filter operation will be supported.

Table 9.6.8-4 defines the default and allowed values of *notificationContentType* for each of the supported values of *notificationEventType*.

Table 9.6.8-4: Default and allowed values of *notificationContentType*

notificationEventType / notificationContentType	A	B	C	D	E	F	G
"modified attributes"	valid	n/a	n/a	n/a	n/a	n/a	valid (default)
"all attributes"	valid (default)	valid (default)	valid (default)	valid (default)	valid (default)	n/a	n/a
"ID" of the resource indicated in the *notificationEventType* condition	valid	valid	valid	valid	valid	n/a	n/a
"Trigger Payload"	n/a	n/a	n/a	n/a	n/a	valid (default)	n/a

9.6.9 Resource Type *schedule*

The <*schedule*> resource contains scheduling information. The usage of the <*schedule*> resource is slightly different depending on the associated resource type, as follows:

- A child <*schedule*> resource of the <*node*> resource shall indicate the time periods when the node can communicate via the Underlying Network. If multiple Underlying Networks are supported, for each there can be a maximum of one <schedule> resources. One <schedule> resource may be used for multiple Underlying Networks.

  The *mgmtLink* attribute of the <*cmdhNwAccessRule*> child of a <node> resource shall link to a <schedule> resource, child of the same <node> resource.

NOTE: The node will obey the communication schedule indicated for the Underlying Network. If the schedule information is modified, the node will ensure that the change of schedule is detected e.g. via external DM, subscription/notification mechanisms, polling, etc.

- A child <*schedule*> resource of the <*CSEBase*> resource shall indicate the anticipated time periods when the CSE is available for processing.
- A child <*schedule*> resource of the <*subscription*> resource shall indicate the time periods when the notifications can be sent to the notification targets.

The <schedule> resource shall contain the child resource specified in table 9.6.9-1.

Table 9.6.9-1: Child resources of <*schedule*> resource

Child Resources of <*schedule*>	Child Resource Type	Multiplicity	Description	<*scheduleAnnc*> Child Resource Types
[variable]	<*subscription*>	0..n	See clause 9.6.8	None
[variable]	<*transaction*>	0..n	See clause 9.6.48	<*transaction*>

The <*schedule*> resource shall contain the attributes specified in table 9.6.9-2.

Table 9.6.9-2: Attributes of <*schedule*> resource

Attributes of <*schedule*>	Multiplicity	RW/ RO/ WO	Description	<*scheduleAnnc*> Attributes
resourceType	1	RO	See clause 9.6.1.3.	NA
resourceID	1	RO	See clause 9.6.1.3.	NA
resourceName	1	WO	See clause 9.6.1.3.	NA
parentID	1	RO	See clause 9.6.1.3.	NA
expirationTime	1	RW	See clause 9.6.1.3.	MA
creationTime	1	RO	See clause 9.6.1.3.	NA
lastModifiedTime	1	RO	See clause 9.6.1.3.	NA
labels	0..1 (L)	RW	See clause 9.6.1.3.	MA
announceTo	0..1 (L)	RW	See clause 9.6.1.3.	NA
announcedAttribute	0..1 (L)	RW	See clause 9.6.1.3.	NA
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3	NA
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.	NA
scheduleElement	1 (L)	RW	Each item of the *scheduleElement* list shall be composed from seven fields of second, minute, hour, day of month, month, day of week and year.	OA
networkCoordinated	0..1	RW	Indicates if IN-CSE shall perform schedule coordination with an Underlying Network. This attribute is only applicable when <schedule> is a child resource of <node>. The supported values are: • True: The IN-CSE shall perform schedule coordination. • False: The IN-CSE may not perform schedule coordination. See note.	OA
NOTE: The schedule coordination is also subject to IN-CSE local policy.				

9.6.10 Resource Type *locationPolicy*

The <*locationPolicy*> resource represents the method for obtaining and managing geographical location information of an M2M Node.

The actual location or event result (in case Geo-fence-Based method) information shall be stored in a <*contentInstance*> resource which is a child resource of the <*container*> resource. The <*container*> resource includes the *locationID* attribute which holds the ID of this <*locationPolicy*> resource. A CSE can obtain location information based on the attributes defined under <*locationPolicy*> resource, and store the location information in the target <*container*> resource.

Based on the *locationSource* attribute, the method for obtaining location information of an M2M Node can be differentiated. The methods for obtaining location information shall be as follows:

- **Network-based method:** where the CSE on behalf of the AE obtains the target M2M Node's location information from an Underlying Network.
- **Device-based method:** where the ASN is equipped with any location capable modules or technologies (e.g. GPS) and is able to position itself.
- **Sharing-based method:** where the ADN has no GPS nor an Underlying Network connectivity. Its location information can be retrieved from either the associated ASN or a MN.

NOTE: Geographical location information could include more than longitude and latitude.

Figure 9.6.10-1 shows the graphical information regarding the event types for the Geo-fence feature defined as the *geofenceEventCriteria* attribute. The time difference between t_1 and t_2 described in the figure below is defined by *locationUpdatePeriod* attribute.

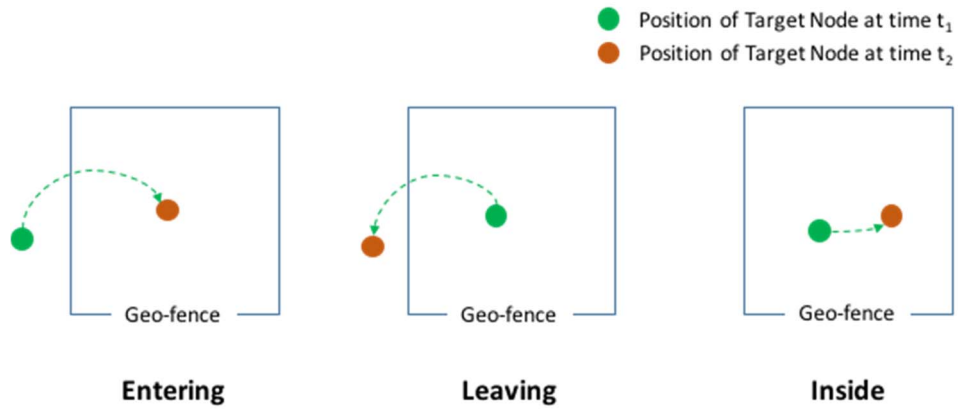


Figure 9.6.10-1: The Event Types for the Geo-fence

The <*locationPolicy*> resource shall contain the child resources specified in table 9.6.10-1.

Table 9.6.10-1: Child resources of <*locationPolicy*> resource

Child Resources of <*locationPolicy*>	Child Resource Type	Multiplicity	Description	<*locationPolicyAnnc*> Child Resource Types
[variable]	<*subscription*>	0..n	See clause 9.6.8	*None*
[variable]	<*transaction*>	0..n	See clause 9.6.48	<*transaction*>

The <*locationPolicy*> resource shall contain the attributes specified in table 9.6.10-2.

Table 9.6.10-2: Attributes of <*locationPolicy*> resource

Attributes of <*locationPolicy*>	Multiplicity	RW/ RO/ WO	Description	<*locationPolicyAnnc*> Attributes
resourceType	1	RO	See clause 9.6.1.3.	NA
resourceID	1	RO	See clause 9.6.1.3.	NA
resourceName	1	WO	See clause 9.6.1.3.	NA
parentID	1	RO	See clause 9.6.1.3.	NA
expirationTime	1	RW	See clause 9.6.1.3.	MA
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.	MA
creationTime	1	RO	See clause 9.6.1.3.	NA
lastModifiedTime	1	RO	See clause 9.6.1.3.	NA
labels	0..1 (L)	RW	See clause 9.6.1.3.	MA
announceTo	0..1 (L)	RW	See clause 9.6.1.3.	NA
announcedAttribute	0..1 (L)	RW	See clause 9.6.1.3.	NA
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.	OA

Attributes of <*locationPolicy*>	Multiplicity	RW/ RO/ WO	Description	<*locationPolicyAnnc*> Attributes
locationSource	1	WO	Indicates the source of location information: • Network Based; • Device Based; • Sharing Based.	OA
locationInformationType	1	RW	Indicate the types of location information: • Position fix (e.g. longitude and latitude); • Geo-fence event (e.g. entering and leaving).	OA
locationUpdatePeriod	0..1(L)	RW	Indicates the period for updating location information. If the value is marked '0' or not defined and *locationUpdateEventCriteria* is absent, location information is updated only when a retrieval request to the <*latest*> child resource of the <*container*> indicated by *locationContainerID* is detected. If the attribute has more than one value and the hosting CSE of the resource is the target device, the value could be selected within the listed values depending on device's local context information (e.g. velocity, status of battery, range of the location etc.). Zero ('0') shall not be stored with non zero value(s). When the value is read, the first value in the list is the current active update period.	OA
locationTargetID	0..1	WO	The identifier to be used for retrieving the location information of a remote Node and this attribute is only used for the case that location information is provided by a location server. For example, when the remote Node is a 3GPP UE, *locationTargetID* could be M2M-Ext-ID or MSISDN.	OA
locationServer	0..1	WO	Indicates the identity of the location or Geo-fence server. This attribute is only used in that case location information is provided by a location server or Geo-fence server.	OA
locationContainerID	1	RO	ID of the <*container*> resource where the actual location information or event result of a M2M Node is stored.	OA

Attributes of <*locationPolicy*>	Multiplicity	RW/ RO/ WO	Description	<*locationPolicyAnnc*> Attributes
locationContainerName	0..1	WO	A name of the <*container*> resource where the actual location information of a M2M Node is stored. If it is not assigned, the Hosting CSE automatically assigns a name of the resource (see note).	OA
locationStatus	1	RO	Contains the information on the current status of the location request (e.g. location server fault).	OA
geographicalTargetArea	0..1	RW	Indicates area information where the Geo-fence feature is applied. The area is a polygon and represented as a list of geographical coordinates that define the perimeter corner points of the polygon.	OA
geofenceEventCriteria	0..1	RW	Indicate the event type of Geo-fence feature: • Entering; • Leaving; • Inside; • Outside.	OA
authID	0..1	RW	Indicates the identity of the application which retrieves the location information of a remote node. This attribute is only used by the Location Server in the Underlying Network to verify whether the application is authorized to request the location information. See clause 7.1.8 where this attribute is described.	OA
retrieveLastKnownLocation	0..1	RW	Indicates if the Hosting CSE shall retrieve the last known location when the Hosting CSE fails to retrieve the latest location. This attribute shall only be applicable when the *locationSource* is Network Based. The supported values are: • True: Shall retrieve the last known location. • False: Shall not retrieve the last known location.	OA
locationUpdateEventCriteria	0..1	RW	Indicates the type of event that shall result in a location update. This attribute shall only be applicable when locationUpdatePeriod is zero or not defined and *locationSource* is Network Based. The supported values are: • LocationChange	OA
NOTE: The created <*container*> resource related to this policy shall be stored only in the Hosting CSE.				

9.6.11 Resource Type *delivery*

When a CSE is requested to initiate an operation (CRUDN) targeting resources on another CSE, then it needs to do scheduling and execution of delivery of data from the source CSE to the target CSE in line with the provisioned policies. It shall be in one of the following ways:

- using delivery aggregation (***Delivery Aggregation*** information set to ON); or
- forwarding the original request as a separate request on the Mcc reference point without changes.

In order to be able to initiate and manage the execution of data delivery in a resource-based manner, resource type <*delivery*> is defined. This resource type shall be used for forwarding requests from one CSE to another CSE when the ***Delivery Aggregation*** parameter in the request is set to ON. If the ***Delivery Aggregation*** parameter is set to OFF, the original request shall be forwarded without change to the next CSE, i.e. without the use of <*delivery*> resource. If the ***Delivery Aggregation*** parameter is not present, the latter method shall be used.

Operations to Retrieve, Update or Delete a <*delivery*> resource shall allow authorized entities to inquire the status of a delivery, change delivery attributes or cancel a delivery.

As defined in clause 10.2.4, <*delivery*> resource can only be created by a CSE. A request for the creation of a <*delivery*> resource can only be issued to a registrar or registree CSE from a registree or registrar CSE with a direct registration relationship among each other (i.e. no transit CSE). <*delivery*> resource is deleted on successful delivery of the data in the *aggregatedRequest* attribute to the next hop CSE.

The parent of a <*delivery*> resource is the <*CSEBase*> resource of the CSE that accepted the request for the creation of the <*delivery*> resource.

The <*delivery*> resource shall contain the child resource specified in table 9.6.11-1.

Table 9.6.11-1: Child resources of *<delivery>* resource

Child Resources of *<delivery>*	Child Resource Type	Multiplicity	Description
[variable]	*<subscription>*	0..n	See clause 9.6.8

The <*delivery*> resource shall contain the attributes specified in table 9.6.11-2.

Table 9.6.11-2: Attributes of <*delivery*> resource

Attributes of <*delivery*>	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
labels	0..1 (L)	RW	See clause 9.6.1.3.
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.
source	1	WO	The CSE-ID of the CSE that initiated the delivery process represented by this <delivery> resource.
target	1	WO	CSE-ID that defines the Hosting CSE for delivering the data contained in the *aggregatedRequest* attribute.
lifespan	1	RW	Defines the time limit when the delivery of the information in the *aggregatedRequest* attribute needs to complete. If the *lifespan* expires before successful delivery, no further attempts to deliver the information in the *aggregatedRequest* attribute need to be executed. If the delivery fails, a feedback may be expected by the source CSE depending on options reflected in the *deliveryMetaData* attribute. The *lifespan* attribute of a <*delivery*> resource shall be set consistent with the ***Request Expiration Timestamp*** parameters of the set of original requests contained in the *aggregatedRequest* attribute, i.e. *lifespan* shall not extend beyond the earliest expiring ***Request Expiration Timestamp*** parameter in the set of the original requests contained in the *aggregatedRequest* attribute.
eventCat	1	RW	Defines the category of the event that triggered the delivery request represented by this <*delivery*> resource.
deliveryMetaData	1	RW	Contains meta information on the delivery process represented by this <*delivery*> resource, such as delivery status, delivery options, tracing information, etc.
aggregatedRequest	1	WO	Attribute containing the request(s) to be delivered to the Hosting CSE. This represents one or more original requests that were targeting the same Hosting CSE.

9.6.12 Resource Type *request*

Creation of a <*request*> resource shall only be done on a Receiver CSE implicitly when a Registree AE or a Registree/Registrar CSE issues a request to the Receiver CSE targeting any other resource type or requesting a notification in non-blocking mode. Creation of a <*request*> resource is only permitted by the Receiver CSE after reception of a valid which contains the ***Response Type*** parameter that is set to *'nonBlockingReqeustSynch'* or *'nonBlockingRequestAsynch'*.

When a CSE is requested to initiate an operation for which the result should be available to the Originator by ***Request Expiration Timestamp*** information of the request set to *'nonBlockingReqeustSynch'* or *'nonBlockingRequestAsynch'*, the Receiver CSE which received the request directly from the Originator shall provide a reference of the created <*request*> resource back to the Originator so that the Originator can access attributes of the <*request*> resource at a later time - for instance in order to retrieve the result of an operation that was taking a longer time. If the Receiver CSE supports <request> resource type, the reference that shall be given back to the Originator as part of the acknowledgment that is the address of the <*request*> resource. The Originator (or any other authorized entity depending on access control) can access the request status and the requested operation result through it.

The <*request*> resource may be deleted by the CSE that is hosting it when the expiration time of the <*request*> resource is reached. So after the expiration time of a <*request*> resource is reached it cannot be assumed that particular <*request*> resource is still accessible. A <*request*> resource may also get deleted earlier than the expiration time, when the result of the requested operation (if any result was requested at all) has been sent back to the Originator.

For the purpose of providing a standardized structure for expressing and accessing the context of a previously issued request, the resource type <*request*> is defined. The parent resource of a <*request*> resource shall be the <*CSEBase*> resource of the Hosting CSE.

The <*request*> resource shall contain the child resources specified in table 9.6.12-1.

Table 9.6.12-1: Child resources of <*request*> resource

Child Resources of <*request*>	Child Resource Type	Multiplicity	Description
[variable]	<*subscription*>	0..n	See clause 9.6.8

The <*request*> resource shall contain the attributes specified in table 9.6.12-2.

Table 9.6.12-2: Attributes of <*request*> resource

Attributes of <*request*>	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3. The value of the *expirationTime* is chosen by the CSE dependent on the ***Request Expiration Timestamp***, ***Result Expiration Timestamp***, ***Result Persistence*** and ***Operation Execution Time*** parameters associated with the original request.
parentID	1	RO	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RO	See clause 9.6.1.3.
labels	0..1 (L)	RO	See clause 9.6.1.3.
dynamicAuthorizationConsultationIDs	0..1 (L)	RO	See clause 9.6.1.3.
operation	1	RO	It contains the value of the parameter ***Operation*** in the original request message.
target	1	RO	It contains the value of the parameter ***To*** in the original request message.
originator	1	RO	It contains the value of the parameter ***From*** in the original request message.
requestID	1	RO	It contains the value of the parameter ***Request Identifier*** in the original request message.
metaInformation	1	RO	Meta information about the request. The content of this attribute is equivalent to information in any other optional parameters described in clause 8.1.
primitiveContent	0..1	RO	Contains the content that is carried in the ***Content*** parameter of the original request message.
requestStatus	1	RO	Contains information on the current status of the Request, e.g. "accepted and pending".
operationResult	1	RO	Contains the result of the originally requested operation in line with the ***Result Content*** parameter associated with the original request.

All operations on <*request*> resources except for the CREATE operations - which shall only be triggered implicitly are controlled by the access control policy.

9.6.13 Resource Type *group*

The <*group*> resource represents a group of resources of the same or mixed types. The <*group*> resource can be used to do bulk manipulations on the resources represented by the *memberIDs* attribute. The <*group*> resource contains an attribute that represents the members of the group and the <*fanOutPoint*> virtual resource that enables generic operations to be applied to all the resources represented by those members. By grouping <*semanticDescriptor*> resources across which a semantic description is distributed, another virtual resource (<*semanticFanOutPoint*>) enables semantic discovery procedures to be applied across the full logical tree in the description.

Members of a <group> resource may support unicast or multicast communication. In case multiple members of a <group> resource support multicast communications and they share the same multicast address, those members form a multicast group as a sub-set of the group. There may be multiple multicast groups corresponded to one group since the members of the group may use different multicast mechanisms (e.g. 3GPP MBMS vs. IP multicast) and different multicast addresses.

The <*group*> resource shall contain the child resources specified in table 9.6.13-1.

Table 9.6.13-1: Child resources of <group> resource

Child Resources of <*group*>	Child Resource Type	Multiplicity	Description	<*groupAnnc*> Child Resource Types
[variable]	<*semanticDescriptor*>	*0..n*	See clause 9.6.30	<*semanticDescriptor*>, <*semanticDescriptorAnnc*>
[variable]	<*subscription*>	0..n	See clause 9.6.8	<*subscription*>
fopt	<*fanOutPoint*>	1	See clause 9.6.14	*None*
sfop	<*semanticFanOutPoint*>	0..1	See clause 9.6.14a	*None*
[variable]	<*transaction*>	0..n	See clause 9.6.48	<*transaction*>

The <*group*> resource shall contain the attributes specified in table 9.6.13-2.

Table 9.6.13-2: Attributes of <*group*> resource

Attributes of <*group*>	Multiplicity	RW/ RO/ WO	Description	<*groupAnnc*> Attributes
resourceType	1	RO	See clause 9.6.1.3.	NA
resourceID	1	RO	See clause 9.6.1.3.	NA
resourceName	1	WO	See clause 9.6.1.3.	NA
parentID	1	RO	See clause 9.6.1.3.	NA
expirationTime	1	RW	See clause 9.6.1.3.	MA
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.	MA
labels	0..1 (L)	RW	See clause 9.6.1.3.	MA
creationTime	1	RO	See clause 9.6.1.3.	NA
lastModifiedTime	1	RO	See clause 9.6.1.3.	NA
announceTo	0..1 (L)	RW	See clause 9.6.1.3.	NA
announcedAttribute	0..1 (L)	RW	See clause 9.6.1.3.	NA
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.	OA
creator	0..1	RO	See clause 9.6.1.3.	NA
memberType	1	WO	It is the resource type of the member resources of the group, if all member resources (including the member resources in any sub-groups) are of the same type. Otherwise, it is of type 'mixed'.	OA
specializationType	0..1	WO	When the *memberType* attribute value is set to "mgmtObj" this *specializationType* may be set to the specialization defined by the *mgmtDefinition* attribute. When the *memberType* attribute value is set to "flexContainer", this *specializationType* may be set to the specialization defined by the *containerDefinition* attribute.	OA

Attributes of <*group*>	Multiplicity	RW/ RO/ WO	Description	<*groupAnnc*> Attributes
currentNrOfMembers	1	RO	Current number of members in a group. It shall not be larger than *maxNrOfMembers*.	OA
maxNrOfMembers	1	RW	Maximum number of members in the <*group*>.	OA
memberIDs	1 (L)	RW	List of member resource IDs referred to in the remaining of the present document as *memberID*. Each ID (*memberID*) should refer to a member resource or a (sub-) <*group*> resource of the <*group*> if *memberID* is suffixed with *"/fopt"*. A <group> resource with an empty member list is allowed.	OA
membersAccessControlPolicyIDs	0..1 (L)	RW	List of IDs of the <*accessControlPolicy*> resources defining who is allowed to access the <*fanOutPoint*> and <*semanticFanOutPoint*> virtual resources.	OA
memberTypeValidated	0..1	RO	Denotes if the resource types of all members' resources of the group have been validated by the Hosting CSE. In the case that the *memberType* attribute of the <*group*> resource is not 'mixed', then this attribute shall be set.	OA
consistencyStrategy	1	WO	This attribute determines how to deal with the <*group*> resource if the *memberType* validation fails. Its possible values are • ABANDON_MEMBER • ABANDON_GROUP • SET_MIXED Which means delete the inconsistent member if the attribute is ABANDON_MEMBER; delete the group if the attribute is ABANDON_GROUP; set the *memberType* to "mixed" if the attribute is SET_MIXED. If it is not given by the Originator at the creation procedure, default is "ABANDON_MEMBER"	OA
groupName	0..1	RW	Human readable name of the <*group*>.	OA
semanticSupportIndicator	0..1	RO	Indicator of support for sematic discovery functionality via <semanticFanOutPoint>.	OA
notifyAggregation	0..1	RW	This attribute specifies the number of messages and/or the duration that the group hosting CSE will aggregate notification messages when the subscriptions created specify aggregation of notifications i.e. specifying the notificationForwardingURI of the original <subscription> resource.	OA

9.6.14 Resource Type *fanOutPoint*

The <*fanOutPoint*> resource is a virtual resource because it does not have a representation. It is the child resource of a <*group*> resource. Whenever a request is sent to the <*fanOutPoint*> resource, the request is fanned out to each of the members of the <*group*> resource indicated by the *membersIDs* attribute of the <*group*> resource. The responses (to the request) from each member are then aggregated and returned to the Originator. A timer should be set for the aggregation. The responses are aggregated if all the responses expected have been received or when the timer expires. The responses received after the time expires should be discarded. If the ***Result Expiration Timestamp*** parameter is received from the Originator, the timer should be set to enforce this parameter, otherwise, the timer is set based on the local policy configured at the Hosting CSE.

The <*fanOutPoint*> resource does not have a resource representation by itself and consequently it does not have an *accessControlPolicyIDs* attribute. The <*accessControlPolicy*> resource used for access control policy validation is indicated by the *membersAccessControlPolicyIDs* attribute in the parent <*group*> resource.

9.6.14a Resource Type *semanticFanOutPoint*

The <*semanticFanOutPoint*> resource is a virtual resource because it does not have a representation. It is the child resource of a <*group*> resource with members of type <*semanticDescriptor*> or <contentInstance>. In the former case the semantic information is contained in the *descriptor* attribute, in the latter case semantic information can be contained in the *content* attribute. It is allowed to reference a <latest> resource representing the most recent <contentInstance> in a container.

Whenever a semantic discovery request is sent to the <*semanticFanOutPoint*> resource the host uses the *memberIDs* attribute of the parent <*group*> resource to retrieve all the related resources. If there are resources stored on different CSEs, individual RETRIEVE requests are sent to each CSE for retrieving the external resources. In case no semantic information is contained, the resource is not considered. All semantic resources are accessed based on the respective access control policies.

Once all of the related resources with semantic information have been retrieved, the semantic content of each is added to the content on which the SPARQL request is being executed. The full/enlarged content subject to the SPARQL request is provided to the SPARQL engine for processing.

The <*semanticFanOutPoint*> resource uses *membersAccessControlPolicyIDs* attribute in the parent <*group*> resource for access control policy validation.

9.6.15 Resource Type *mgmtObj*

The <*mgmtObj*> resource contains management data which represents individual M2M management functions. It represents a general structure to map to technology specific data model e.g. OMA DM [i.3], BBF TR-069 [i.2] and LWM2M [i.4]. Each instance of <*mgmtObj*> resource shall be mapped to single technology specific protocol.

The <*mgmtObj*> resource shall contain the child resource specified in table 9.6.15-1.

Table 9.6.15-1: Child resources of <*mgmtObj*> resource

Child Resources of <*mgmtObj*>	Child Resource Type	Multiplicity	Description	<*mgmtObjAnnc*> Child Resource Type
[variable]	<*subscription*>	0..n	See clause 9.6.8	<*subscription*>
[variable]	<*semanticDescriptor*>	0..n	See clause 9.6.30	<*semanticDescriptor*>, <*semanticDescriptorAnnc*>
[variable]	<*transaction*>	0..n	See clause 9.6.48	<*transaction*>

The <*mgmtObj*> resource shall contain the attributes specified in table 9.6.15-2.

Table 9.6.15-2: Attributes of <*mgmtObj*> resource

Attributes of <*mgmtObj*>	Multiplicity	RW/ RO/ WO	Description	<*mgmtObjAnnc*> Attributes
resourceType	1	RO	See clause 9.6.1.3.	NA
resourceID	1	RO	See clause 9.6.1.3.	NA
resourceName	1	WO	See clause 9.6.1.3.	NA
parentID	1	RO	See clause 9.6.1.3.	NA
expirationTime	1	RW	See clause 9.6.1.3.	MA
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.	MA
creationTime	1	RO	See clause 9.6.1.3.	NA
lastModifiedTime	1	RO	See clause 9.6.1.3.	NA
labels	0..1 (L)	RW	See clause 9.6.1.3.	MA
announceTo	0..1 (L)	RW	See clause 9.6.1.3.	NA
announcedAttribute	0..1 (L)	RW	See clause 9.6.1.3.	NA
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.	OA

Attributes of <*mgmtObj*>	Multiplicity	RW/ RO/ WO	Description	<*mgmtObjAnnc*> Attributes
mgmtDefinition	1	WO	Specifies the type of <*mgmtObj*> resource e.g. software, firmware, memory. The list of the value of the attribute can be seen in annex D.	MA
mgmtSchema	0..1	WO	Contains a URI to the <*mgmtObj*> schema definition which shall be used by the Hosting CSE to validate the syntax of incoming primitives targeting this <*mgmtObj*> resource. This URI may refer to a oneM2M specified <*mgmtObj*> definition as well as other <mgmtObj> definitions.	MA
objectIDs	0..1 (L)	WO	Contains the list URNs that uniquely identify the technology specific data model objects used for this <*mgmtObj*> resource as well as the managed function and version it represents. This attribute shall be provided during the creation of the <*mgmtObj*> resource and shall not be modifiable afterwards. If the <*mgmtObj*> resource is mapped to multiple technology specific data model objects, this attribute shall list all URNs for each mapped technology specific data model objects. This is mandatory for the <*mgmtObj*>, for which the data model is not specified by oneM2M but mapped from technology specific data model.	OA
objectPaths	0..1 (L)	WO	Contains the list of local paths of the technology specific data model objects on the managed entity which is represented by the <*mgmtObj*> resource in the Hosting CSE. This attribute shall be provided during the creation of the <*mgmtObj*>, so that the Hosting CSE can correlate the created <*mgmtObj*> with the technology specific data model object on the managed entity for further management operations. It shall not be modifiable after creation. The format of this attribute shall be a local technology specific data model object path in the form as specified by technology specific protocol. (e.g. "./anyPath/Fw1" in OMA DM [i.3], "Device.USBHosts.Host.3." in BBF TR-069 [i.2]). The combination of the *objectPaths* and the *objectIDs* attribute, allows to address the technology specific data model.	OA
mgmtLink	0..1 (L)	RW	This attribute contains reference to a list of other <*mgmtObj*> resources in case a hierarchy of <*mgmtObj*> is needed.	OA
[objectAttribute]	0..n	RW	Each *[objectAttribute]* is mapped from a leaf node of a hierarchical structured technology specific data model object (including oneM2M data model and the technology specific data model objects) based on the mapping rules below the table.	OA

Attributes of <*mgmtObj*>	Multiplicity	RW/ RO/ WO	Description	<*mgmtObjAnnc*> Attributes
description	0..1	RW	Text format description of <*mgmtObj*>.	OA

When mapping objects from technology specific protocol to a corresponding <*mgmtObj*> resource, the following rules shall apply:

- The root objects of technology specific data model objects maps to the <*mgmtObj*> resource.
- For the child of the root of technology specific data model objects:
  - **Rule1:** If the child technology specific data model object cannot have another child technology specific data model object, the technology specific data model object maps to the *[objectAttribute]* attribute of the <*mgmtObj*> resource with the same resource name.
  - **Rule2:** If the child technology specific data model object can have another child technology specific data model object, the technology specific data model object maps to a new <*mgmtObj*> resource. The ID of the new <*mgmtObj*> resource is stored as an *mgmtLink* attribute of the <*mgmtObj*> resource which is mapped from the parent technology specific data model object.

9.6.16 Resource Type *mgmtCmd*

The <*mgmtCmd*> resource represents a method to execute management procedures or to model commands and remote procedure calls (RPC) required by existing management protocols (e.g. BBF TR-069 [i.2]), and enables AEs to request management procedures to be executed on a remote entity. It also enables cancellation of cancellable and initiated but unfinished management procedures or commands.

Each <*mgmtCmd*> corresponds to a specific type of management command, as defined by its attribute *cmdType*. For multiple requests of the same management command, <*mgmtCmd*> shall use separate child-resources (i.e. <*execInstance*>) to contain each execution instance. The execution of the management procedure represented by <*mgmtCmd*> shall be triggered using the UPDATE method to its attribute *execEnable*.

The <*mgmtCmd*> resource shall contain the child resources specified in table 9.6.16-1.

Table 9.6.16-1: Child resources of <*mgmtCmd*> resource

Child Resources of <*mgmtCmd*>	Child Resource Type	Multiplicity	Description
[variable]	<*subscription*>	0..n	See clause 9.6.8
[variable]	<*execInstance*>	0..n	See clause 9.6.17
[variable]	<*transaction*>	0..n	See clause 9.6.48

The <*mgmtCmd*> resource shall contain the attributes specified in table 9.6.16-2.

Table 9.6.16-2: Attributes of <*mgmtCmd*> resource

Attributes of <*mgmtCmd*>	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3
labels	0..1 (L)	RW	See clause 9.6.1.3
creationTime	1	RO	See clause 9.6.1.3
lastModifiedTime	1	RO	See clause 9.6.1.3
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.
description	0..1	RW	The text-format description of this resource.
cmdType	1	WO	The type to identify the management operation (e.g. download).
execReqArgs	0..1	RW	Structured attribute (e.g. abstract type) to contain any command-specific arguments of the request.
execEnable	1	RW	Writing a value to this attribute via the UPDATE method causes execution of the <mgmtCmd>.
execTarget	1	RW	ID of the <*node*> resource of the target on which this <*mgmtCmd*> will be executed. It may be the URI of a <*group*> resource in which case the <*mgmtCmd*> will be executed on all members in the *memberIDs* attribute of the addressed <*group*> resource.
execMode	0..1	RW	The mode used to specify how the command will be executed (e.g. Immediate Once, Immediate and Repeatedly, Random Once, Random and Repeatedly). May be used together with *execFrequency*, *execDelay* and *execNumber* to provide the scheduling information.
execFrequency	0..1	RW	The minimum interval between two executions, to be used in conjunction with *execMode*. Modes involving random execution can be used to add random values between individual executions.
execDelay	0..1	RW	The minimum delay before the instance should be executed. Modes involving random execution can be used to increase this delay randomly.
execNumber	0..1	RW	The number of times the instance should be executed, to be used when *execMode* indicates a repetition pattern.

9.6.17 Resource Type *execInstance*

The <*execInstance*> resource represents a successful instance of <*mgmtCmd*> execution request, which had been triggered by a M2M network application using the UPDATE method to the attribute *execEnable* of <*mgmtCmd*> resource.

The <*execInstance*> resource shall contain the child resources specified in table 9.6.17-1.

Table 9.6.17-1: Child resources of <*execInstance*> resource

Child Resources of <*execInstance*>	Child Resource Type	Multiplicity	Description
[variable]	<*subscription*>	0..n	See clause 9.6.8
[variable]	<*transaction*>	0..n	See clause 9.6.48

The <*execInstance*> resource shall contain the attributes specified in table 9.6.17-2.

Table 9.6.17-2: Attributes of *<execInstance>* resource

Attributes of *<execInstance>*	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
expirationTime	1	RO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
labels	0..1 (L)	RW	See clause 9.6.1.3.
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.
execStatus	1	RO	The status of <*execInstance*>. It can be Initiated, Started, Finished, Cancelled, or Deleted.
execResult	1	RO	The execution result of <*execInstance*>.
execDisable	0..1	RW	The attribute is used to cancel <*execInstance*> using UPDATE method.
execTarget	1	RO	ID of <*node*> resource of the target on which the <*execInstance*> will be executed.
execMode	0..1	RO	Modes used to specify how the command will be executed (e.g. Immediate Once, Immediate and Repeatedly, Random Once, Random and Repeatedly). May be used together with *execFrequency*, *execDelay* and *execNumber* to provide the scheduling information.
execFrequency	0..1	RO	The minimum interval between two executions, to be used in conjunction with *execMode*. Modes involving random execution can be used to add random values between individual executions.
execDelay	0..1	RO	The minimum delay before the instance should be executed. Modes involving random execution can be used to increase this delay randomly.
execNumber	0..1	RO	The number of times the instance should be executed, to be used when *execMode* indicates a repetition pattern.
execReqArgs	0..1 (L)	RO	Structured attribute (e.g. abstract type) to contain any command-specific arguments (as a list) used to trigger this <*execInstance*>.

9.6.18 Resource Type *node*

The <*node*> resource represents specific information that provides properties of an M2M Node that can be utilized by other oneM2M operations. The <*node*> resource has specialization of the <*mgmtObj*> as its child resources. These resources represent the Node's context information (e.g. memory and battery), network topology, device information, device capability etc. The specialized <*mgmtObj*> resources are used to perform management of the Node.

This node specific information stored in these resources such as *[memory]* and *[battery]* can be obtained either by the existing device management technologies (OMA DM [i.3], BBF TR-069 [i.2]) or any other way (e.g. JNI [i.18]).

For the case when the <*node*> resource belongs to an ADN, please see figure 9.6.18-1 in conjunction with the description of *nodeLink* attribute in the <*AE*> resource (clause 9.6.5).

For the case when the <*node*> resource belongs to an NoDN and the applications that correspond to interworked devices are represented by <*flexContainer*>*s* please see figure 9.6.18-2.

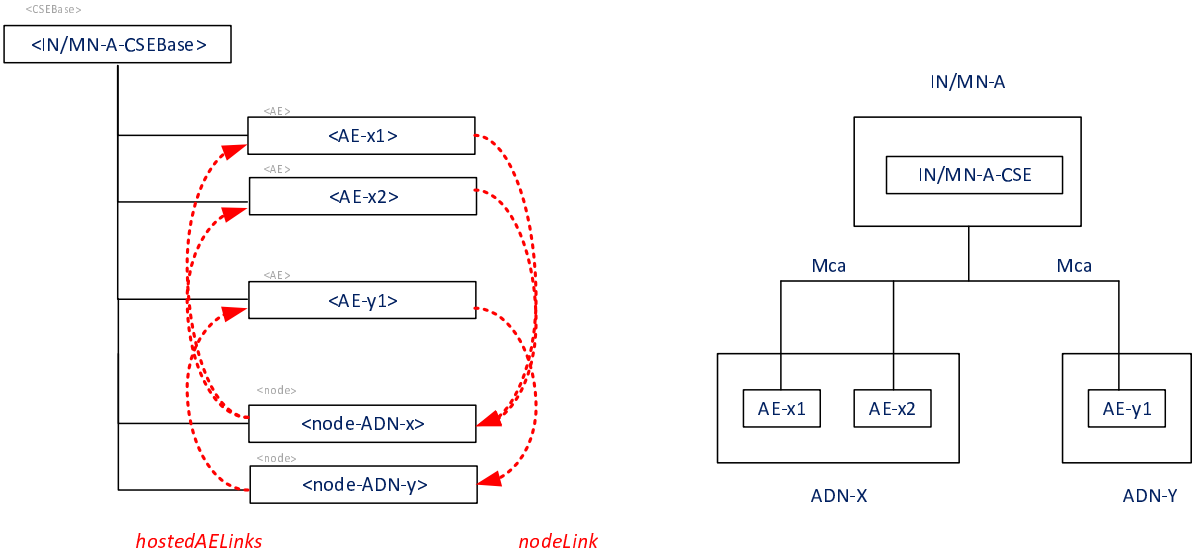


Figure 9.6.18-1: Relationship between IN/MN and ADN

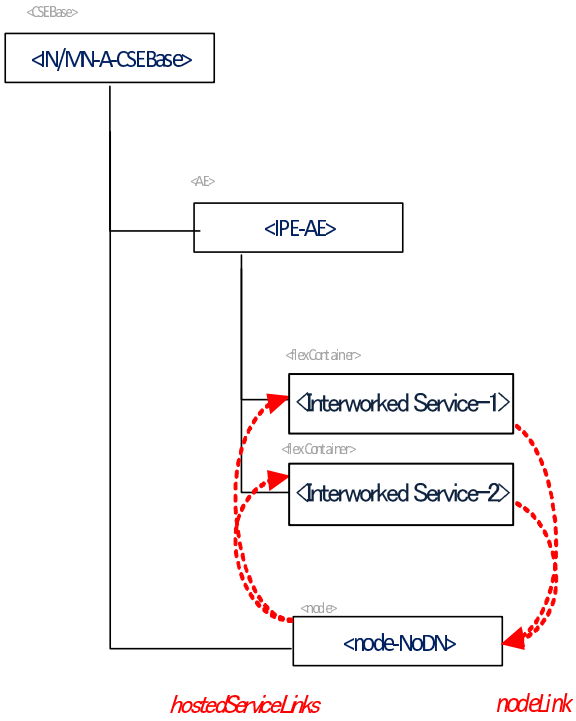


Figure 9.6.18-2: Relationship between IPE, interworked Services and NoDN

The <*node*> resource shall contain the child resources specified in table 9.6.18-1.

Table 9.6.18-1: Child resources of <*node*> resource

Child Resources of <*node*>	Child Resource Type	Multiplicity	Description	<*nodeAnnc*> Child Resource Type
[variable]	<*semanticDescriptor*>	0..n	See clause 9.6.30	<*semanticDescriptor*>, <*semanticDescriptorAnnc*>
[variable]	<*mgmtObj*> as defined in the specialization [*memory]*	0..1	This resource provides the memory (typically RAM) information of the node. (E.g. the amount of total volatile memory), See clause D.4.	<*mgmtObjAnnc*>
[variable]	<*mgmtObj*> as defined in the specialization [*battery]*	0..n	The resource provides the power information of the node. (E.g. remaining battery charge). See clause D.7.	<*mgmtObjAnnc*>
[variable]	<*mgmtObj*> as defined in the specialization [*areaNwkInfo]*	0..n	This resource describes the list of Nodes attached behind the MN/ASN node and its physical or underlying relation among the nodes in the M2M Area Network. This attribute is defined in case the Node is MN/ASN. See clause D.5.	<*mgmtObjAnnc*>
[variable]	<*mgmtObj*> as defined in the specialization [*areaNwkDeviceInfo]*	0..n	This resource describes the information about the Node in the M2M Area Network. See clause D.6.	<*mgmtObjAnnc*>
[variable]	<*mgmtObj*> as defined in the specialization [*firmware]*	*0..n*	This resource describes the information about the firmware of the Node include name, version etc. See clause D.2.	<*mgmtObjAnnc*>
[variable]	<*mgmtObj*> as defined in the specialization [*software]*	0..n	This resource describes the information about the software of the Node. See clause D.3.	<*mgmtObjAnnc*>
[variable]	<*mgmtObj*> as defined in the specialization [*deviceInfo]*	0..n	The resource contains information about the identity, manufacturer and model number of the device. See clause D.8.	<*mgmtObjAnnc*>
[variable]	<*mgmtObj*> as defined in the specialization [*deviceCapability]*	0..n	The resource contains information about the capability supported by the Node. See clause D.9.	<*mgmtObjAnnc*>
[variable]	<*mgmtObj*> as defined in the specialization [*reboot]*	0..1	The resource is the place to reboot or reset the Node. See clause D.10.	<*mgmtObjAnnc*>
[variable]	<*mgmtObj*> as defined in the specialization [*eventLog]*	0..1	The resource contains the information about the log of events of the Node. See clause D.11.	<*mgmtObjAnnc*>
[variable]	<*mgmtObj*> as defined in the specialization *[cmdhPolicy]*	0..n	The resource(s) contain(s) information about CMDH policies that are applicable to the CMDH processing on the CSE hosted on the node represented by this <*node*> resource and identified by the *hostedCSELink* attribute of this <*node*> resource. See clause D.12.	NA
[variable]	<*mgmtObj*> as defined in the specialization *[activeCmdhPolicy]*	0..1	This resource defines which of the present *[cmdhPolicy]* resource(s) shall be active for the CMDH processing on the CSE hosted on the node represented by this <*node*> resource and identified by the *hostedCSELink* attribute of this <*node*> resource. See clause D.12.	NA
[variable]	<*subscription*>	0..n	See clause 9.6.8.	<*subscription*>
[variable]	<*schedule*>	0..n	See clause 9.6.9.	<*scheduleAnnc*>

Child Resources of <*node*>	Child Resource Type	Multiplicity	Description	<*nodeAnnc*> Child Resource Type
[variable]	<*transaction*>	0..n	See clause 9.6.48	<*transaction*>

The <*node*> resource shall contain the attributes specified in table 9.6.18-2.

Table 9.6.18-2: Attributes of <*node*> resource

Attributes of <*node*>	Multiplicity	RW/ RO/ WO	Description	<*nodeAnnc*> attributes
resourceType	1	RO	See clause 9.6.1.3.	NA
resourceID	1	RO	See clause 9.6.1.3.	NA
resourceName	1	WO	See clause 9.6.1.3.	NA
parentID	1	RO	See clause 9.6.1.3.	NA
expirationTime	1	RW	See clause 9.6.1.3.	MA
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.	MA
creationTime	1	RO	See clause 9.6.1.3.	NA
lastModifiedTime	1	RO	See clause 9.6.1.3.	NA
labels	0..1 (L)	RW	See clause 9.6.1.3.	MA
announceTo	0..1 (L)	RW	See clause 9.6.1.3.	NA
announcedAttribute	0..1 (L)	RW	See clause 9.6.1.3.	NA
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.	OA
nodeID	1	RW	The M2M-Node-ID of the node which is represented by this <*node*> resource.	MA
hostedCSELink	0..1	RW	This attribute allows to find the <CSEBase> or <remoteCSE> resource representing the CSE that is residing on the node that is represented by this <*node*> resource. The attribute contains the resource ID of a resource where all of the following applies: • The resource is a <*CSEBase*> resource or a <*remoteCSE*> resource. • The resource represents the CSE which resides on the specific node that is represented by the current <*node*> resource. In case the node that is represented by this <node> resource does not contain a CSE, this attribute shall not be present.	OA
hostedAELinks	0..1(L)	RW	This attribute allows to find the AEs hosted by the node that is represented by this <*node*> resource. The attribute shall contain a list of resource identifiers of <*AE*> resources representing the ADN-AEs residing on the node that is represented by the current <*node*> resource. In case the node that is represented by this <node> resource does not contain an AE, this attribute shall not be present.	OA

Attributes of <*node*>	Multiplicity	RW/ RO/ WO	Description	<*nodeAnnc*> attributes
hostedServiceLinks	0..1(L)	RW	This attribute allows to find <*flexContainer*> *resources that have* been created by an IPE to represent services hosted on a NoDN, the NoDN being represented by this <*node*> resource. If the NoDN hosts a set of services represented by <*flexContainer*>*s,* then the attribute shall contain the list of resource identifiers of these <*flexContainer*> resources. In case the node that is represented by this <*node*> resource does not contain a service that is represented by a <*flexContainer*>*,* this attribute shall not be present.	OA
mgmtClientAddress	0..1	RW	Represents the physical address of management client of the node which is represented by this <node> resource. This attribute is absent if management server is able to acquire the physical address of the management client.	OA
roamingStatus	0..1	RO	Indicates if the M2M Node is currently roaming from the perspective of the underlying network. The allowed values are "Yes" or "No".	OA
networkID	0..1	RO	Configured with the identity of the underlying network which the M2M Node is currently attached to.	OA

9.6.19 Resource Type *m2mServiceSubscriptionProfile*

The <*m2mServiceSubscriptionProfile*> resource represents an M2M Service Subscription. It is used to represent all data pertaining to the M2M Service Subscription, i.e. the technical part of the contract between an M2M Application Service Provider and an M2M Service Provider and is only stored on IN-CSE. The data is also represented in <serviceSubscribedNode> and <serviceSubscribedAppRule> resources as well as <m2mServiceSubscriptionProfile> resource. The relationship among those three resource types are depicted as follows. Note that the diagram does not capture all attributes and child resources. Those resource types shall only be instantiated on IN-CSE.

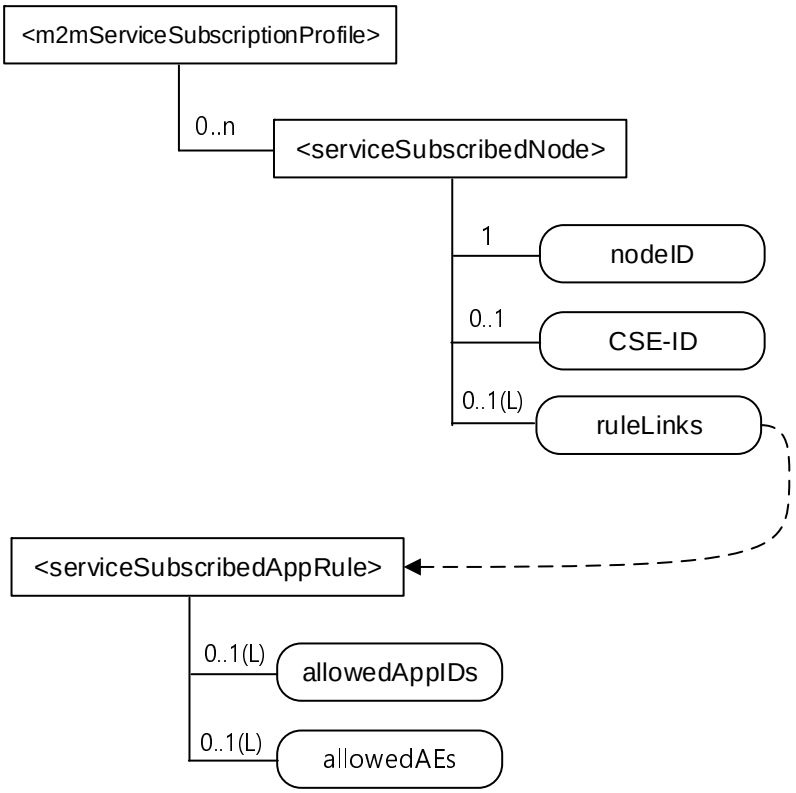


Figure 9.6.19-1: Relationship among M2M Service Subscription related resources

The <*m2mServiceSubscriptionProfile*> resource shall contain the child resources specified in table 9.6.19-1.

Table 9.6.19-1: Child resources of <*m2mServiceSubscriptionProfile*> resource

Child Resources of <*m2mServiceSubscriptionProfile*>	Child Resource Type	Multiplicity	Description
[variable]	<*subscription*>	0..n	See clause 9.6.8
[variable]	<*serviceSubscribedNode*>	0..n	See clause 9.6.20
[variable]	<*transaction*>	0..n	See clause 9.6.48

The <*m2mServiceSubscriptionProfile*> resource shall contain the attributes specified in table 9.6.19-2.

Table 9.6.19-2: Attributes of <*m2mServiceSubscriptionProfile*> resource

Attributes of <*m2mServiceSubscriptionProfile*>	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
labels	0..1 (L)	RW	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.

9.6.20 Resource Type *serviceSubscribedNode*

The <*serviceSubscribedNode*> resource represents M2M Node information that is needed as part of the M2M Service Subscription resource and is only stored on IN-CSE. It contains M2M-Node-ID and optionally CSE-ID running on that Node.

The <*serviceSubscribedNode*> resource shall contain the child resource specified in table 9.6.20-1.

Table 9.6.20-1: Child resources of <*serviceSubscribedNode*> resource

Child Resources of <*serviceSubscribedNode*>	Child Resource Type	Multiplicity	Description
[variable]	<*subscription*>	0..n	See clause 9.6.8
[variable]	<*transaction*>	0..n	See clause 9.6.48

The <*serviceSubscribedNode*> resource shall contain the attributes specified in table 9.6.20-2.

Table 9.6.20-2: Attributes of <*serviceSubscribedNode*> resource

Attributes of <*serviceSubscribedNode*>	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
labels	0..1 (L)	RW	See clause 9.6.1.3
lastModifiedTime	1	RO	See clause 9.6.1.3.
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.
nodeID	1	WO	M2M-Node-ID of the node that is represented by this instance.
CSE-ID	0..1	WO	CSE-ID pertaining to this node (for nodes that have a CSE).

Attributes of <*serviceSubscribedNode*>	Multiplicity	RW/ RO/ WO	Description
deviceIdentifier	0..1 (L)	WO	A list of device identifiers that uniquely identify a device. The format of a device identifier is one of the following: • **Case 1:** Identify a device using the format <OUI> "-" <ProductClass> "-" <SerialNumber> as defined in section 3.4.4 of BBF TR-069 [i.2]. The format of the URN is urn:dev:ops:<OUI> "-" <ProductClass> "-" <SerialNumber>. • **Case 2:** Identify a device using the format <OUI> "-"<SerialNumber> as defined in section 3.4.4 of BBF TR-069 [i.2]. The format of the URN is urn:dev:os:<OUI> "-"<SerialNumber>. • **Case 3:** Identify a device using an International Mobile Equipment Identifiers of ETSI TS 123 003 [i.23]. This URN specifies a valid, 15 digit IMEI. The format of the URN is urn:imei:################. • **Case 4:** Identify a device using an Electronic Serial Number. The ESN specifies a valid, 8 digit ESN. The format of the URN is urn:esn:########. • **Case 5:** Identify a device using a Mobile Equipment Identifier. This URN specifies a valid, 14 digit MEID. The format of the URN is urn:meid:##############. • **Case 6:** Identify a device using an Object IDentifier (OID). This URN specifies a valid OID - see annex H for one possible naming convention. The format of the URN is urn:oid:<OID>. • **Case 7:** Identify a device using a Universally Unique IDentifier (UUID). The UUID specifies a valid, hex digit character string as defined in IETF RFC 4122 [i.26]. The format of the URN is urn:uuid: ########-####-####-####-############.
ruleLinks	0..1 ((L)	RW	This attribute contains a list of links towards <*serviceSubscribedAppRule*> resources pertaining to this <*serviceSubscribedNode*>. See clause 9.6.29 for an explanation of the <*serviceSubscribedAppRule*> resource. This attribute shall exist only when the CSE-ID attribute is present. When the list is empty, it means no applications are allowed to register on the CSE which is indicated by the CSE-ID attribute.
niddRequired	0..1	RW	Controls whether the IN-CSE configures the underlying network to enable Non-IP Data Delivery for this node. Valid values are "TRUE" or "FALSE". If not configured, then IN-CSE default policy shall apply. See ETSI TS 118 126 [15].

9.6.21 Resource Type *pollingChannel*

The <*pollingChannel*> resource represents a channel that can be used for a request-unreachable entity (i.e. an AE or a CSE which is behind NAT so it cannot receive a request from other Nodes). The request-unreachable entity creates a <*pollingChannel*> resource on a request-reachable CSE, and then polls any type of request(s) for itself from the <*pollingChannel*> Hosting CSE.

EXAMPLE: An AE can retrieve notifications by long polling on the channel when it cannot receive notifications asynchronously from a subscription Hosting CSE.

The <*pollingChannel*> resource shall contain the child resource specified in table 9.6.21-1.

Table 9.6.21-1: Child resources of *<pollingChannel>* resource

Child Resources of *<pollingChannel>*	Child Resource Type	Multiplicity	Description
pcu	*<pollingChannelURI>*	1	See clause 9.6.22
[variable]	*<transaction>*	0..n	See clause 9.6.48

The <*pollingChannel*> resource shall contain the attributes specified in table 9.6.21-2.

Table 9.6.21-2: Attributes of *<pollingChannel>* resource

Attributes of *<pollingChannel>*	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
expirationTime	1	RO	See clause 9.6.1.3.
labels	0..1 (L)	RW	See clause 9.6.1.3.

9.6.22 Resource Type *pollingChannelURI*

The <*pollingChannelURI*> virtual resource is the child resource of the <*pollingChannel*> resource and is used to perform service layer long polling. The AE or CSE which created the <*pollingChannel*> resource on its Registrar CSE sends a Retrieve request targeting the <*pollingChannelURI*> resource as a service layer long polling request. The response to the long polling request shall be pending until there are any requests received on the channel or the request reaches the request expiration time.

9.6.23 Resource Type *statsConfig*

The <*statsConfig*> resource is used to store policies of statistics for AEs. The <*statsConfig*> resource may be established by the IN-CSEs or by IN-AEs. The <*statsConfig*> resource shall be located directly under <*CSEBase*>.

The <*statsConfig*> resource shall contain the child resources specified in table 9.6.23-1.

Table 9.6.23-1: Child resources of *<statsConfig>* resource

Child Resources of *<statsConfig>*	Child Resource Type	Multiplicity	Description
[variable]	*<eventConfig>*	0..n	See clause 9.6.24. This resource configures an event for statistics collection.
[variable]	*<subscription>*	0..n	See clause 9.6.8 where the type of this resource is described.
[variable]	*<transaction>*	0..n	See clause 9.6.48

The <*statsConfig*> resource shall contain the attributes specified in table 9.6.23-2.

Table 9.6.23-2: Attributes of <*statsConfig*> resource

Attributes of <*statsConfig*>	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3
creationTime	1	RO	See clause 9.6.1.3
expirationTime	1	RW	See clause 9.6.1.3
lastModifiedTime	1	RO	See clause 9.6.1.3
labels	0..1 (L)	RW	See clause 9.6.1.3
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.
creator	0..1	RO	See clause 9.6.1.3.

9.6.24 Resource Type *eventConfig*

<*eventConfig*> sub-resource shall be used to define events that trigger statistics collection. Below are some examples of events that can be generated:

- Collection based on a certain operation: collects any RETRIEVE operations on the data (i.e. resources) stored in the IN-CSE.
- Collection based on storage size: collects the size of storage when a "Content Sharing Resource" stored in the IN-CSE exceeds a quota.
- Combined configuration: collects all RETRIEVE operations on the data stored in the IN-CSE during a period of time.

The <*eventConfig*> resource shall contain the child resource specified in table 9.6.24-1.

Table 9.6.24-1: Child resources of <*eventConfig*> resource

Child Resources of <*eventConfig*>	Child Resource Type	Multiplicity	Description
[variable]	<*subscription*>	0..n	See clause 9.6.8 where this type of resource is described.
[variable]	<*transaction*>	0..n	See clause 9.6.48

The <*eventConfig*> resource shall contain the attributes specified in table 9.6.24-2.

Table 9.6.24-2: Attributes of <*eventConfig*> resource

Attributes of <*eventConfig*>	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
labels	0..1 (L)	RW	See clause 9.6.1.3.
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.
creator	0..1	RO	See clause 9.6.1.3.
eventID	1	RO	This attribute uniquely identifies the event to be collected for statistics for AEs.
eventType	1	RW	This attribute indicates the type of the event: timer based, data operation, or storage based.
eventStart	0..1	RW	This attribute indicates the start time of the event.
eventEnd	0..1	RW	This attribute indicates the end time of the event.
operationType	0..1 (L)	RW	This attribute defines the type of the operation to be collected by statistics, such as CREATE, RETRIEVE.
dataSize	0..1	RW	This attribute defines the data size that will trigger a storage based event. For <container> and <timeSeries> *currentByteSize* is compared. For <contentInstance>, <flexContainer>, <timeSeriesInstance> *contentSize* is compared. An event is triggered when the compared data size exceeds *dataSize* size.
eventResourceTypes	0..1 (L)	RW	This attribute indicates the list of resource types for which an event is to be captured and reported. This could be used to differentiate the same operation on different types of resources that triggers the charging activity. If this attribute is specified, then *eventResourceIDs* shall not be specified.
eventResourceIDs	0..1 (L)	RW	This attribute indicates the list of resourceIDs for which the event is to be captured and reported. Whenever an operation is performed on the resourceIDs in this list, an event will be recorded provided other event criteria are met such as eventResourceType, locationRestriction and the event information based on the type of event. If this attribute is specified, then *eventResourceTypes* shall not be specified.

9.6.25 Resource Type *statsCollect*

The <*statsCollect*> resource shall be used to collect information for AEs using the <*eventConfig*> resource as the triggers in the IN-CSE. Multiple triggers can be established at IN-CSE for the same AE. Each trigger may be activated or de-activated independently of others. The <*statsCollect*> resource shall be located directly under <*CSEBase*> of IN-CSE.

The <*statsCollect*> resource shall contain the child resource specified in table 9.6.25-1.

Table 9.6.25-1: Child resources of <*statsCollect*> resource

Child Resources of <*statsCollect*>	Child Resource Type	Multiplicity	Description
[variable]	<*subscription*>	0..n	See clause 9.6.8 where the type of this resource is described.
[variable]	<*transaction*>	0..n	See clause 9.6.48.

The <*statsCollect*> resource shall contain the attributes specified in table 9.6.25-2.

Table 9.6.25-2: Attributes of <*statsCollect*> resource

Attributes of <*statsCollect*>	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
labels	0..1 (L)	RW	See clause 9.6.1.3.
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.
creator	0..1	RO	See clause 9.6.1.3.
statsCollectID	1	RO	This is the unique ID to identify a specific statistics collection scenario. It is created by the IN-CSE when the <*statsCollect*> resource is first created.
collectingEntityID	1	WO	This is the unique ID of the entity that requests the collection of statistics. For example, it can be an *AE-ID* or *CSE-ID*.
collectedEntityID	0..1(L)	WO	This is the list of unique ID of the entities whose request triggered the configured event for statistics collection. For example, each ID can be an AE-ID or IN-CSE-ID. If no specific value is provided for this attribute, the IN-CSE interprets it as "any entity".
statsRuleStatus	1	RW	This attribute indicates whether the rule is "active" or "inactive".
statModel	1	RW	This attribute indicates the collection model, such as "Subscriber based", "event based", etc.
collectPeriod	0..1	RW	Expresses time periods defined by second, minute, hour day of month, month, and year. Supports repeating periods, and wildcards expressed as a list.
eventID	0..1	RW	This attribute refers to the <*eventConfig*> resource that defines the events that can be collected by the IN-CSE. It is mandatory if the *statModel* attribute is set to "event based".

9.6.26 Resource Announcement

9.6.26.1 Overview

A resource can be announced to one or more remote CSEs to inform the remote CSEs of the existence of the original resource. An announced resource can have a limited set of attributes and a limited set of child resources from the original resource. The announced resource includes a link to the original resource hosted by the original resource-Hosting CSE.

In case that the original resource is deleted, all announced resources for the original resource shall be deleted, except for <*AEAnnc*> resources that were created during the registration of an AE with AE-ID-Stem starting with "S", which shall not be deleted. If the announced resource is not deleted promptly (e.g. the announced resource is not reachable), the announced resource can be deleted later either by the original resource Hosting CSE or by the expiration of the announced resource itself. The original resource shall store the list of links for the announced resources for those purposes.

Synchronization between the attributes announced by the original resource and the announced resource shall be the responsibility of the original resource Hosting CSE. There shall not be any synchronization for children created at the original resource and the announced resource. The access control policy for the announced resource shall synchronize with the one from the original resource. In case that the attribute *accessControlPolicyIDs* is not present in the original resource it is the responsibility of the original resource Hosting CSE to choose the appropriate value depending on the policy for the original resource (e.g. take the parent *accessControlPolicyIDs* value).

The original resource shall have at least *announceTo* attribute present if the resource itself has been announced. If any of the Optional Announced (OA) attributes are also announced, then *announcedAttribute* attribute shall also be present. An AE or other CSE can request the original resource Hosting CSE for announcing the original resource to the list of CSE-IDs or the address(es) listed in the *announceTo* attribute in the announcing request. An Update to the *announceTo* attribute will trigger new resource announcement(s) or the de-announcement(s) of the announced resource. After a successful announcement procedure the attribute *announceTo* contains only the list of address(es) of the announced resources.

In order to announce an attribute marked as **OA** (see clause 9.5.0), the attribute shall be included in the *announcedAttribute* attribute list at the original resource. The attributes included in the *announcedAttribute* attribute are announced to the announced resource. On successful announcement of the resource, such attributes shall be created at the announced resource; otherwise they shall not be present in the announced resource. Update to the *announcedAttribute* attribute in the original resource will trigger new attribute announcement or the de-announcement of the announced attribute(s). The announced attributes shall have the same value as the original resource, and synchronization between the value of the announced attributes at the original resource and the announced resource is the responsibility of the original resource Hosting CSE.

An announced resource may have child resources. In general, a child resource of an announced resource shall be of one of the resource types that are specified as possible child resource types for the original resource or of one of their associate announced resource types. However, for specific announced resource types, specific exceptions apply regarding which child resource types can occur. The details on which child resources are specified for each announced resource type are summarized in table 9.6.26.1-1.

Child resources of the original resource can be announced independently as needed. In this case, the child resources at the announced resource shall be of the child resource's associated announced type. When a child resource at the announced resource is created locally at the remote CSE, the child resource shall be of ordinary - i.e. not-announced - child resource type.

When a Hosting CSE of an original resource is initiating an announcement, it shall first check if the parent resource is announced to the announcement target CSE by checking the *announceTo* attribute of the parent resource and if so, create the announced resource as a child of the announced parent resource. If the parent resource is not announced, the Hosting CSE shall check if <CSEBase> is announced to the announcement target CSE by checking the *announceTo* attribute of <CSEBase>. If it is not announced, the Hosting CSE shall create a <CSEBaseAnnc> to the announcement target CSE. The Hosting CSE shall then create the announced resource as a child resource of the <CSEBaseAnnc> resource.

When a Hosting CSE of an original resource is initiating an announcement, the *From* parameter of the announce request shall contain either a SP-relative-CSE-ID of the Hosting CSE of the original resource if the announcement target CSE resides in the same SP domain or an Absolute-CSE-ID of the Hosting CSE of the original resource if the announcement target CSE resides in a different SP domain.

If an attribute is marked as **RO** and also marked as **MA** or **OA**, then only the attribute of the original resource shall be interpreted as **RO**. The corresponding attribute of the announced resource shall be always writable to the original resource hosting CSE to allow it to properly announce and de-announce the attribute and keep the announced attribute synchronized with the original one. Only the original resource Hosting CSE shall be allowed to update and delete the announced attribute which is created by the original resource Hosting CSE.

Table 9.6.26.1-1: Announced Resource Types

Announced Resource Type	Short Description	Child Resource Types	Clause
accessControlPolicyAnnc	Announced variant of *accessControlPolicy*	*subscription*	9.6.2
AEAnnc	Announced variant of *AE*	*subscription,* *container,* *containerAnnc,* *flexContainer,* *flexContainerAnnc,* *group,* *groupAnnc,* *accessControlPolicy,* *accessControlPolicyAnnc* *semanticDescriptor,* *semanticDescriptorAnnc,* *timeSeries,* *timeSeriesAnnc*	9.6.5
containerAnnc	Announced variant of *container*	*container,* *containerAnnc,* *flexContainer,* *flexContainerAnnc,* *contentInstance,* *contentInstanceAnnc,* *subscription,* *semanticDescriptor,* *semanticDescriptorAnnc,* *timeSeries,* *timeSeriesAnnc*	9.6.6
contentInstanceAnnc	Announced variant of *contentInstance*	*semanticDescriptor,* *semanticDescriptorAnnc*	9.6.7
CSEBaseAnnc	Announced variant of CSEBase	*container,* *containerAnnc,* *dynamicAuthorizationConsultationAnnc,* *flexContainer,* *flexContainerAnnc,* *group,* *groupAnnc,* *accessControlPolicy,* *accessControlPolicyAnnc,* *subscription,* *scheduleAnnc,* *semanticDescriptorAnnc,* *semanticMashupJobProfileAnnc,* *timeSeries,* *timeSeriesAnnc,* *remoteCSEAnnc,* *nodeAnnc,* *mgmtObjAnnc,* *AEAnnc,* *locationPolicyAnnc*	9.6.3
dynamicAuthorizationConsultationAnnc	Announced variant of *dynamicAuthorizationConsultation*	None specified	9.6.40
flexContainerAnnc	Announced variant of flex*Container*	*container,* *containerAnnc,* *flexContainer,* *flexContainerAnnc,* *subscription,* *semanticDescriptor,* *semanticDescriptorAnnc,* *timeSeries,* *timeSeriesAnnc*	9.6.35

Announced Resource Type	Short Description	Child Resource Types	Clause
groupAnnc	Announced variant of *group*	*subscription,* *semanticDescriptor,* *semanticDescriptorAnnc*	9.6.13
locationPolicyAnnc	Announced variant of *locationPolicy*	None specified	9.6.10
mgmtObjAnnc	Announced variant of *mgmtObj*	*subscription*	9.6.15
nodeAnnc	Announced variant of *node*	*mgmtObjAnnc,* *subscription,* *semanticDescriptor,* *semanticDescriptorAnnc,* *scheduleAnnc*	9.6.18
remoteCSEAnnc	Announced variant of *remoteCSE*	*container,* *containerAnnc,* *dynamicAuthorizationConsultationAnnc,* *flexContainer,* *flexContainerAnnc,* *group,* *groupAnnc,* *accessControlPolicy,* *accessControlPolicyAnnc,* *subscription,* *scheduleAnnc,* *semanticDescriptorAnnc,* *semanticMashupJobProfileAnnc,* *timeSeries,* *timeSeriesAnnc,* *remoteCSEAnnc,* *nodeAnnc,* *mgmtObjAnnc,* *AEAnnc,* *locationPolicyAnnc*	9.6.4
scheduleAnnc	Announced variant of *schedule*	None specified	9.6.9
semanticDescriptorAnnc	Announced variant of *semanticDescriptor*	Subscription	9.6.30
semanticMashupInstanceAnnc	Announced variant of *semanticMashupInstance*	None specified	9.6.54
semanticMashupJobProfileAnnc	Announced variant of *semanticMashupJobProfile*	None specified	9.6.53
timeSeriesAnnc	Announced variant of timeSeries	timeSeriesInstance, timeSeriesInstanceAnnc, subscription, semanticDescriptor, semanticDescriptorAnnc	9.6.36
timeSeriesInstanceAnnc	Announced variant of timeSeriesInstance	None specified	9.6.37

9.6.26.2 Universal Attributes for Announced Resources

Table 9.6.26.2-1 lists the universal attributes for the announced resources. If an attribute is marked **"NA"** in the original resource type or it is marked **"OA"** and is not provided by the Originator, then the value for the corresponding attribute in the announced resource is provided by the <*remote CSE*> resource.

Table 9.6.26.2-1: Universal Attributes for Announced Resources

Attributes Name	Mandatory /Optional	Description
resourceType	Mandatory	Resource Type. As specified in clause 9.2, a suffix of "**Annc**" to the name of the original resource type shall be used to indicate the name for the associated announced resource type.
resourceID	Mandatory	Identifies the resource at the remote CSE
resourceName	Mandatory	See clause 9.6.1.3 for information on this attribute
parentID	Mandatory	Identifies the parent resource at the remote CSE.
creationTime	Mandatory	See clause 9.6.1.3 for information on this attribute.
lastModifiedTime	Mandatory	See clause 9.6.1.3 for information on this attribute.
expirationTime	Mandatory	See clause 9.6.1.3.2 for information on this attribute. This attribute cannot exceed the value received from the original resource but it can be overridden by the policies of the remote CSE hosting the announced resource.
link	Mandatory	Provides the URI to the original resource.

9.6.26.3 Common Attributes for Announced Resources

Table 9.6.26.3-1 lists the common attributes for the announced resources.

Table 9.6.26.3-1: Commonly Used Attributes for Announced Resources

Attribute Name	Mandatory /Optional	Description
accessControlPolicyIDs	Conditionally Mandatory	The list of identifiers (either an ID or a URI) of an <*accessControlPolicy*> resource announced by the original resource See clause 9.6.1.3.2 for further information on this attribute. If this attribute was not present in the original resource, the original resource shall include this attribute by providing the *accessControlPolicyIDs* from the original resource's parent resource or from the local policy according at the original resource.
stateTag	Conditionally Mandatory	An incremental counter of modification on the resource. See clause 9.6.1.3.2 for information on this attribute.
labels	Conditionally Mandatory	Tokens used as keys for discovering resources as announced by the original resource. See clause 9.6.1.3 for further information on this attribute. The attribute is conditionally mandatory, which means that the attribute shall exist in the announced resource if it is present in the original resource.
registrationStatus	Optional	Only optional for announced <AE> resource. Denotes status of the announced AE registration. If ACTIVE, the announced <*AE*> resource and all its child resources may be discoverable. If INACTIVE, the announced <*AE*> registration and all its child resources shall not be discoverable. The attribute is conditionally mandatory, which means that the attribute shall exist in the announced resource if it is present in the original resource.

9.6.27 Resource Type *latest*

The <*latest*> resource is a virtual resource because it does not have a representation. It is the child resource of a <*container*> and a <*timeSeries*> resource. When a request addresses the <*latest*> resource, the Hosting CSE shall apply the request to the latest <*contentInstance*> or <*timeSeriesInstance*> resource among all existing <*contentInstance*> or <*timeSeriesInstance*> resources of the <*container*> or <*timeSeries*> resource.

The <*latest*> resource inherits access control policies that apply to the parent resource.

9.6.28 Resource Type *oldest*

The <*oldest*> resource is a virtual resource because it does not have a representation. It is the child resource of a <*container*> and a <*timeSeries*> resource. When a request addresses the <*oldest*> resource, the Hosting CSE shall apply the request to the oldest <*contentInstance*> or <*timeSeriesInstance*> resource among all existing <*contentInstance*> or <*timeSeriesInstance*> resources of the <*container*> or <*timeSeries*> resource.

The <*oldest*> resource inherits access control policies that apply to the parent resource.

9.6.29 Resource Type *serviceSubscribedAppRule*

The <*serviceSubscribedAppRule*> resource represents a rule that defines allowed Role-ID, App-ID and AE-ID combinations that are acceptable for registering an AE on a Registrar CSE and is only stored on IN-CSE. The rule in a <*serviceSubscribedAppRule*> resource shall apply for CSEs for which the associated <*serviceSubscribeNode*> resource is linked with the <*serviceSubscribedAppRule*> via the *ruleLinks* attribute of the <*serviceSubscribedNode*> resource. The rule contained in a <*serviceSubscribedAppRule*> resource defines a mapping between:

a) one or more Credential-ID(s); and

b) combinations of one or more Role-ID(s), one or more App-ID(s) and one or more AE-ID(s) which are allowed to be used for registering AE(s) that issued a registration request via a Security Association established with the credentials associated with the Credential-ID(s) listed in a).

When AEs are allowed to register with no Security Association, then a Credential-ID is not applicable.

The parent resource of a <*serviceSubscribedAppRule*> resource is the <*CSEBase*> resource of the IN-CSE hosting the <*serviceSubscribedNode*> resource(s) that point to the <*serviceSubscribedAppRule*> resource.

The <*serviceSubscribedAppRule*> resource shall contain the child resource specified in table 9.6.29-1.

Table 9.6.29-1: Child resources of <*serviceSubscribedAppRule*> resource

Child Resources of <*serviceSubscribedAppRule*>	Child Resource Type	Multiplicity	Description
[variable]	<*subscription*>	0..n	See clause 9.6.8 where the type of this resource is described.
[variable]	<*transaction*>	0..n	See clause 9.6.48.

The <*serviceSubscribedAppRule*> resource shall contain the attributes specified in table 9.6.29-2.

Table 9.6.29-2: Attributes of <*serviceSubscribedAppRule*> resource

Attributes of <*serviceSubscribedAppRule*>	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
labels	0..1 (L)	RW	See clause 9.6.1.3
lastModifiedTime	1	RO	See clause 9.6.1.3.
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.
applicableCredIDs	0..1 (L)	RW	List of credential IDs for which this rule is applicable, i.e. for registration requests coming into a CSE via a Security Association Endpoint (SEA) [2], that was authenticated using credentials that match with any of these credential-IDs, the current rule applies. This can contain a '*' for any credential ID or not specified for a case when there is no security association. Also Wildcards within an element of this list are possible (e.g. 'C123*X' for any Credential ID that starts with 'C123' and ends with 'X') to define sets or ranges of Credential-IDs.
allowedApp-IDs	0..1 (L)	RW	List of App-IDs that shall be considered to be allowed for AE registration requests received via Security Association Endpoint (SEA) [2] associated with credentialID stored in the attribute *applicableCredID*. This can contain '*' for any App-ID. Also Wildcards within an element of this list are possible (e.g. 'C123*X' for any App-ID that starts with 'C123' and ends with 'X') to define sets or ranges of App-IDs.
allowedAEs	0..1 (L)	RW	List of allowed AE-ID-Stems to be used for the registering AEs. This can contain zero or more specific AE-ID-Stem values, 'S*' for any SP-Assigned AE-ID-Stem, 'C*' for any CSE-assigned AE-ID-Stem, or '*' for any AE-ID-Stem. Also Wildcards within an element of this list are possible (e.g. 'C123*X' for any AE-ID that starts with 'C123' and ends with 'X') to define sets or ranges of AE-ID-Stems.
allowedRole-IDs	0..1(L)	RW	List of Role-IDs that shall be considered to be allowed in Request operations.

9.6.30 Resource Type *semanticDescriptor*

The <*semanticDescriptor*> resource is used to store a semantic description pertaining to a resource and potentially sub-resources. Such a description may be provided according to ontologies. The semantic information is used by the semantic functionalities of the oneM2M system and is also available to applications or CSEs. oneM2M TR-0007 [i.28] provides an informative example of a descriptor attribute.

The <*semanticDescriptor*> resource shall contain the child resources specified in table 9.6.30-1.

Table 9.6.30-1: Child resources of <*semanticDescriptor*> resource

Child Resources of <*semanticDescriptor*>	Child Resource Type	Multiplicity	Description	<*semanticDescriptor Annc*> Child Resource Types
[variable]	<*subscription*>	0..n	See clause 9.6.8 where the type of this resource is described.	<*subscription*>
[variable]	<*transaction*>	0..n	See clause 9.6.48.	<*transaction*>

The <*semanticDescriptor*> resource shall contain the attributes specified in table 9.6.30-2.

Table 9.6.30-2: Attributes of <*semanticDescriptor*> resource

Attributes of <*semanticDescriptor*>	Multiplicity	RW/ RO/ WO	Description	<*semantic Descriptor Annc*> Attributes
resourceType	1	RO	See clause 9.6.1.3.	NA
resourceID	1	RO	See clause 9.6.1.3.	NA
resourceName	1	WO	See clause 9.6.1.3.	NA
parentID	1	RO	See clause 9.6.1.3.	NA
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.	MA
creationTime	1	RO	See clause 9.6.1.3.	NA
expirationTime	1	RW	See clause 9.6.1.3.	MA
lastModifiedTime	1	RO	See clause 9.6.1.3.	NA
labels	0..1 (L)	RW	See clause 9.6.1.3.	MA
announceTo	0..1 (L)	RW	See clause 9.6.1.3.	NA
announcedAttribute	0..1 (L)	RW	See clause 9.6.1.3.	NA
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.	OA
creator	0..1	RO	See clause 9.6.1.3.	NA
descriptorRepresentation	1	RW	Indicates the type used for the serialization of the descriptor attribute, e.g. RDF /XML, OWL/XML.	OA
semanticOpExec	0..1	RW	This attribute cannot be retrieved. Contains a SPARQL query request for execution of semantic operations on the *descriptor* attribute e.g. SPARQL update as described in [3].	NA
descriptor	1	RW	Stores a semantic description pertaining to a resource and potentially sub-resources. Such a description shall be according to subject-predicate-object triples as defined in the RDF graph-based data model [4]. Examples of such descriptors in RDF can be found in oneM2M TR-0007 [i.28].	OA
ontologyRef	0..1	WO	A reference (URI) of the ontology used to represent the information that is stored in the *descriptor attribute*. If this attribute is not present, the *ontologyRef* from the parent resource is used if present.	OA
relatedSemantics	0..1(L)	WO	List of resource identifiers containing related semantic information to be used in processing semantic queries. The resource identifiers may reference either a <*group*> resource or <*semanticDescriptor*> resources and <contentInstance> resources with semantic information in their content attributes as indicated by their contentInfo attribute. In the latter case, the resource identifier may reference a <latest> resource representing the most recent <contentInstance> in a container.	OA
semanticValidated	0..1	RO	A Boolean value representing the validation result of the triples in the *descriptor* attribute. The validation is against the referenced ontology as pointed by the *ontologyRef* attribute as well as other associated <*semanticDescriptor*> resources (and their referenced ontologies) linked by *relatedSemantics* attribute and triples in the *descriptor* attribute.	OA
validationEnable	0..1	RW	A Boolean value indicating whether the triples in the *descriptor* attribute needs to be validated by the hosting CSE. See note.	OA
NOTE: The hosting CSE may override this value according to local policy to enforce or disable semantic validation despite the suggested value from the issuer.				

9.6.31 Resource Type *notificationTargetMgmtPolicyRef*

The <*notificationTargetMgmtPolicyRef*> resource is a child resource of a <subscription> resource and lists a reference to the policy to be followed by the hosting CSE for every Target Notification of a subscription. The policy is applied by the hosting CSE when it receives a request to stop receiving a notification from a Target Notification. If no policy is defined for the Target Notification, then the hosting CSE shall apply the default policy. The default policy is either created by the subscription originator or the hosting CSE shall have a system created default one to apply. The system created default policy shall be configurable by the M2M Service Provider.

Table 9.6.31-1: Child resources of *<notificationTargetMgmtPolicyRef>* resource

Child Resources of *<notificationTargetMgmtPolicyRef>*	Child Resource Type	Multiplicity	Description
[variable]	*<subscription>*	0..n	See clause 9.6.8
[variable]	*<transaction>*	0..n	See clause 9.6.48

Table 9.6.31-2: Attributes of *<notificationTargetMgmtPolicyRef>* resource

Attributes of *<notificationTargetMgmtPolicyRef>*	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
labels	0..1 (L)	RW	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.
notificationTargetURI	1 (L)	RW	address(es) of the resource subscriber receiving a notification. The notificationTarget URI shall be listed in the *notificationURI* attribute of the parent <subscription> resource, otherwise the default Notification Target policy shall apply.
notificationlPolicyID	0..1	RW	A link to the <*notificationTargetPolicy*> resource applicable to the notificationTargetURI. If none is specified than the default policy shall apply to the targetNotificationURI. See clause 9.6.32 for an explanation of the <*notificationTargetPolicy*> resource.

9.6.32 Resource Type *notificationTargetPolicy*

The <*notificationTargetPolicy*> resource is a child resource of <CSEBase> resource and lists the policies to be applied by the hosting CSE. A policy has a rules(s), represented by the <policyDeletionRules> and an action. The action is applied when the rules in the policy are fulfilled.

Rules are grouped in 2 groups to support a combination of rules for flexibility e.g. ((rule 1 AND rule 2) OR rule 3). A maximum of two groups of <policyDeletionRules> can be defined. The relationship to be applied between the 2 groups (AND/OR) shall be defined in the ruleRelationShip attribute. If no rules are defined for a <*notificationTargetPolicy*> then the action is executed.

Each policy has the policyLabel which can take any form. There shall be at minimum a single notificationTargetPolicy which can be defined by the subscription originator with the label "default" to be applied when no specific policy is defined for a Target Notification. If a default policy is required and none is defined by the subscription originator, then the system defined default policy shall be applied.

Table 9.6.32-1: Child resources of *<notificationTargetPolicy>* resource

Child Resources of *<notificationTargetPolicy>*	Child Resource Type	Multiplicity	Description
[variable]	*<policyDeletionRules>*	0..2	Groups listing the rules that apply to this policy and that needs to be fulfilled for the listed action to take place. Only two groups of rules shall be supported. See clause 9.6.33
[variable]	*<subscription>*	0..n	See clause 9.6.8
[variable]	*<transaction>*	0..n	See clause 9.6.48

Table 9.6.32-2: Attributes of *<notificationTargetPolicy>* resource

Attributes of *<notificationTargetPolicy>*	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
labels	0..1 (L)	RW	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.
creator	0..1	RO	See clause 9.6.1.3.
action	1	RW	Defines the action to be performed if the groups of rules are fulfilled. The action includes one of the following; accept request, reject request, seek authorization from subscription originator before responding, or inform the subscription originator without taking any action.
policyLabel	1	RW	At minimum a default policy shall be specified. The policyLabel "Default" shall be used in this case.
rulesRelationship	0..1	RW	Shall be either AND or OR This attribute is mandatory if more than one policy DeletionRule is specified.

9.6.33 Resource Type *policyDeletionRules*

The <*policyDeletionRules*> resource lists the rules to be applied by the hosting CSE during policy execution. Each <policyDeletionRules> can define any number of rules with an AND or OR relationship to be applied between them. The attribute deletionRulesRelation define the relationship between rules. It can have an AND or OR value.

Table 9.6.33-1: Child resources of *<policyDeletionRules>* resource

Child Resources of *<policyDeletionRules>*	Child Resource Type	Multiplicity	Description
[variable]	*<subscription>*	0..n	See clause 9.6.8
[variable]	*<transaction>*	0..n	See clause 9.6.48

Table 9.6.33-2: Attributes of *<policyDeletionRules>* resource

Attributes of *<policyDeletionRules>*	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
labels	0..1 (L)	RW	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.
deletionRules	0..1(L)	RW	Lists the applicable rules. The rules include at minimum; time of the day, geographical location of the Target Notification. Where the rule applies.
deletionRulesRelation	0..1	RW	Defines the relation to be applied between the deletionRules. This shall be either AND or OR.

9.6.34 Resource Type *notificationTargetSelfReference*

The <*notificationTargetSelfReference*> resource is a virtual resource, which does not have a representation and it is the child resource of a <*subscription*> resource. Whenever a Delete Request is sent to the <*notificationTargetSelfReference*> resource from a Notification Target which wants to remove itself from the Notification Target list (i.e. notificationURI) later, the Notifier shall act according to the action attribute defined in the <*notificationTargetPolicy*> resource which is linked from the <*notificationTargetMgmtPolicyRef*> resource defined for the specific notificationURI. If no specific policy is defined for the notification URI then the default policy shall apply.

9.6.35 Resource Type *flexContainer*

The <*flexContainer*> resource type is a customizable container for data instances. It is a template for the definition of flexible specializations of data containers. Like a <*container*> resource, specializations of this <*flexContainer*> resource type are used to share information with other entities and potentially to track the data. While the <*container*> resources includes data to be made accessible to oneM2M entities inside <*contentInstance*> children, a specialization of the <*flexContainer*> resource includes associated content directly inside the <*flexContainer*> by means of one or more [*customAttribute*] attribute(s). The attribute name and attribute data type of [*customAttribute*] attributes are defined explicitly for each specialization of <*flexContainer*>, i.e. the specific set of attribute name and type are defined in a corresponding XSD-file.

Example usage of <*flexContainer*>: As a specialization of <*flexContainer*> that includes two [customAttribute] attributes, named "temperature"(xs:float type) and "humidity"(xs:positiveInteger type) can be specified in some TS. The actual data types of [customAttribute] will be described both in the specification document or XSD file which are referred by the value of *containerDefinition* attribute.

The <*flexContainer*> resource shall contain the child resource specified in table 9.6.35-1.

Table 9.6.35-1: Child resources of <*flexContainer*> resource

Child Resources of <*flexContainer*>	Child Resource Type	Multiplicity	Description	<*flexContainerAnnc*> Child Resource Type
[variable]	<*semanticDescriptor*>	0..n	See clause 9.6.30	<*semanticDescriptor*>, <*semanticDescriptorAnnc*>
[variable]	<*subscription*>	0..n	See clause 9.6.8	<*subscription*>
[variable]	<*container*>	0..n	See clause 9.6.6	<*container*> <*containerAnnc*>
[variable]	<*flexContainer*>	0..n	<flexContainer> resource can include any of its specializations as child resource	<*flexContainer*> <*flexContainerAnnc*>
[variable]	<*timeSeries*>	0..n	See clause 9.6.36	<*timeSeries*>, <*timeSeriesAnnc*>
[variable]	<*transaction*>	0..n	See clause 9.6.48	<*transaction*>

The <*flexContainer*> resource shall contain the attributes specified in table 9.6.35-2.

Table 9.6.35-2: Attributes of <*flexContainer*> resource

Attributes of <*flexContainer*>	Multiplicity	RW/ RO/ WO	Description	<*flexContainerAnnc*> Attributes
resourceType	1	RO	See clause 9.6.1.3.	NA
resourceID	1	RO	See clause 9.6.1.3.	NA
resourceName	1	WO	See clause 9.6.1.3.	NA
parentID	1	RO	See clause 9.6.1.3.	NA
expirationTime	0..1 (note)	RW	See clause 9.6.1.3.	MA
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.	MA
labels	0..1 (L)	RW	See clause 9.6.1.3.	MA
creationTime	0..1 (note)	RO	See clause 9.6.1.3.	NA
lastModifiedTime	0..1 (note)	RO	See clause 9.6.1.3.	NA
stateTag	1	RO	See clause 9.6.1.3. This *stateTag* attribute value shall be incremented when a custom attribute of the flexContainer is modified.	NA
announceTo	0..1 (L)	RW	See clause 9.6.1.3.	NA
announcedAttribute	0..1 (L)	RW	See clause 9.6.1.3.	NA
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.	OA
creator	0..1	RO	See clause 9.6.1.3.	NA
containerDefinition	1	WO	This contains an identifier reference (URI) to the <*flexContainer*> schema definition which shall be used by the CSE to validate the syntax of the <*flexContainer*> resource. This URI may refer to one of the oneM2M <*flexContainer*> definitions specified in the following documents: • Generic Interworking [6] • AllJoyn Interworking [7]; • Home Domain Information Model [8] A list of oneM2M <*flexContainer*> definitions is also provided in clause 9.6.1.2.2 of ETSI TS 118 104 [3]. Other URI for other <*flexContainer*> definitions may be specified.	MA
ontologyRef	0..1	RW	A reference (URI) of the ontology used to represent the information that is stored in the present <*flexContainer*> resource.	OA

Attributes of <*flexContainer*>	Multiplicity	RW/ RO/ WO	Description	<*flexContainerAnnc*> Attributes
contentSize	1	RO	Sum of the size in bytes of all of the custom attributes.	NA
nodeLink	0..1	RW	The resource identifier of a <node> resource that stores the node specific information of the NoDN on which the interworked service represented by this <flexContainer> resource resides.	OA
[customAttribute]	0..n	RW	Specialization-specific attribute(s). Name and data type defined in each specialization of <*flexContainer*> resource.	OA
NOTE: When an instance of <*flexContainer*> is a child of a <*flexContainer*> resource, these attributes can be optional. Their presence is determined by the respective definition referred to by the *containerDefinition* attribute.				

9.6.36 Resource Type *timeSeries*

The <*timeSeries*> resource represents a container for Time Series Data instances. It is used to share information with other entities and potentially to track, detect and report the missing data in Time Series. A <*timeSeries*> resource has no associated content. It has only attributes and child resources.

Table 9.6.36-1: Child resources of <*timeSeries*> resource

Child Resources of <*timeSeries*>	Child Resource Type	Multiplicity	Description	<*timeSeriesAnnc*> Child Resource Types
[variable]	<*semanticDescriptor*>	0..n	See clause 9.6.30	<*semanticDescriptor*>, <*semanticDescriptorAnnc*>
[variable]	<*timeSeriesInstance*>	0..n	See clause 9.6.37	<*timeSeriesInstance*>, <*timeSeriesInstanceAnnc*>
[variable]	<*subscription*>	0..n	See clause 9.6.8	<*subscription*>
la	<*latest*>	1	See clause 9.6.27	*None*
ol	<*oldest*>	1	See clause 9.6.28	*None*
[variable]	<*transaction*>	0..n	See clause 9.6.48	<*transaction*>

The <*timeSeries*> resource shall contain the attributes specified in table 9.6.36-2.

Table 9.6.36-2: Attributes of <*timeSeries*> resource

Attributes of <*timeSeries*>	Multiplicity	RW/ RO/ WO	Description	<*timeSeriesAnnc*> Attributes
resourceType	1	RO	See clause 9.6.1.3.	NA
resourceID	1	RO	See clause 9.6.1.3.	NA
resourceName	1	WO	See clause 9.6.1.3.	NA
parentID	1	RO	See clause 9.6.1.3.	NA
expirationTime	1	RW	See clause 9.6.1.3	MA
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.	MA
labels	0..1 (L)	RW	See clause 9.6.1.3.	MA
creationTime	1	RO	See clause 9.6.1.3.	NA
lastModifiedTime	1	RO	See clause 9.6.1.3.	NA
announceTo	0..1 (L)	RW	See clause 9.6.1.3.	NA
announcedAttribute	0..1 (L)	RW	See clause 9.6.1.3.	NA
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.	OA
creator	0..1	RO	See clause 9.6.1.3.	NA
maxNrOfInstances	0..1	RW	Maximum number of direct child <*timeSeriesInstance*> resources in the <*timeSeries*> resource.	OA

Attributes of <*timeSeries*>	Multiplicity	RW/ RO/ WO	Description	<*timeSeriesAnnc*> Attributes
maxByteSize	0..1	RW	Maximum size in bytes of data that is allocated for the <*timeSeries*> resource for all direct child<*timeSeriesInstance*> resources.	OA
maxInstanceAge	0..1	RW	Maximum age of a direct child <*timeSeriesInstance*> resource in the <*timeSeries*> *resource*. The value is expressed in seconds.	OA
currentNrOfInstances	1	RO	Current number of direct child <*timeSeriesInstance*> resource in the <*timeSeries*> resource. It is limited by the *maxNrOfInstances*. The *currentNrOfInstances* attribute of the <timeSeries> resource shall be updated on successful creation or deletion of direct child <*timeSeriesInstance*> resource of <timeSeries> resource.	NA
currentByteSize	1	RO	Current size in bytes of data stored in all direct child <*timeSeriesInstance*> resources of a <*timeSeries*> resource. It is limited by the *maxByteSize*. The *currentByteSize* attribute of the <timeSeries> resource shall be updated on successful creation or deletion of direct child <*timeSeriesInstance*> resource of <timeSeries> resource.	NA
periodicInterval	0..1	WO	If the Time Series Data is periodic, this attribute shall contain the expected amount of time between two instances of Time Series Data.	OA
periodicIntervalDelta	0..1	WO	If the Time Series Data is periodic, this attribute contains a +/- delta value relative to *periodicInterval* for the purpose of detecting missing data. The value of this attribute shall be less than or equal to (*periodicInterval/2*). If the attribute is omitted the hosting CSE can use a local policy to determine a default value.	OA
missingDataDetect	0..1	WO	Indicates whether the Receiver shall detect the missing Time Series Data if it is periodic.	NA
ontologyRef	0..1	RW	A reference (URI) of the ontology used to represent the information that is stored in the child <*timeSeriesInstance*> resources of the present <*timeSeriesData*> resource (see note).	OA
missingDataMaxNr	0..1	RW	Maximum number of entries in the *missingDataList* if the *periodicInterval* is set and the *missingDataDetect* is TRUE.	OA
missingDataList	0..1(L)	RO	The list of the *dataGenerationTime* value representing the missing Time Series Data in descending order by time if the *periodicInterva*l is set and the *missingDataDetect* is TRUE.	NA
missingDataCurrentNr	0..1	RO	Current number of the missing Time Series Data in the *missingDataList*.	NA
missingDataDetectTimer	0..1	RW	The *missingDataDetectTimer* is a duration after which a <*timeSeriesInstance*> shall be considered missing by the hosting CSE. If *periodicIntervalDelta* is present, the value of this attribute shall be greater than *periodicIntervalDelta.*	OA

Attributes of <*timeSeries*>	Multiplicity	RW/ RO/ WO	Description	<*timeSeriesAnnc*> Attributes
contentInfo	0..1	WO	This attribute contains information to understand the contents of the *content* attribute of <timeSeriesInstance>. It shall be composed of two mandatory components consisting of an Internet Media Type (as defined in the IETF RFC 6838 [i.36]) and an encoding type. In addition, an optional content security component may also be included. The format of this attribute is defined in ETSI TS 118 104 [3]. This attribute should be used to represent the content information of the *content* attribute of child <*timeSeriesInstance*> resources so that AEs can understand the content.	OA
NOTE: The access to this URI is out of scope of oneM2M.				

9.6.37 Resource Type *timeSeriesInstance*

The <*timeSeriesInstance*> resource represents a data instance in the <*timeSeries*> resource. The <*timeSeriesInstance*> resource shall not be modified once created. An AE shall be able to delete a <*timeSeriesInstance*> resource explicitly or it may be deleted by the platform based on policies. If the platform has policies for <*timeSeriesInstance*> retention, these shall be represented by the attributes *maxByteSize*, *maxNrOfInstances* and/or *maxInstanceAge* attributes in the <*timeSeries*> resource. If multiple policies are in effect, the strictest policy shall apply. The <*timeSeriesInstance*> resource inherits the same access control policies of the parent <*timeSeries*> resource, and does not have its own *accessControlPolicyIDs* attribute.

Table 9.6.37-1: Child resources of <*timeSeriesInstance*> resource

Child Resources of <*timeSeriesInstance*>	Child Resource Type	Multiplicity	Description	<*timeSeriesInstanceAnnc*> Child Resource Types
[variable]	<*transaction*>	0..n	See clause 9.6.48	<*transaction*>

The <*timeSeriesInstance*> resource shall contain the attributes specified in table 9.6.37-2.

Table 9.6.37-2: Attributes of *<timeSeriesInstance>* resource

Attributes of *<timeSeriesInstance>*	Multiplicity	RW/ RO/ WO	Description	*<timeSeriesInstance Annc>* Attributes
resourceType	1	RO	See clause 9.6.1.3.	NA
resourceID	1	RO	See clause 9.6.1.3.	NA
resourceName	1	WO	See clause 9.6.1.3.	NA
parentID	1	RO	See clause 9.6.1.3.	NA
labels	0..1 (L)	WO	See clause 9.6.1.3.	MA
creationTime	1	RO	See clause 9.6.1.3.	NA
expirationTime	1	WO	See clause 9.6.1.3.	NA
announceTo	0..1 (L)	WO	See clause 9.6.1.3.	NA
announcedAttribute	0..1 (L)	WO	See clause 9.6.1.3.	NA
lastModifiedTime	1	RO	See clause 9.6.1.3.	NA
dataGenerationTime	1	WO	This attribute contains the time when the data was generated by the Originator. The value of this attribute shall be unique among the child <*timeSeriesInstance*> resources belonging to the same parent <*timeSeries*> resource.	OA
content	1	WO	This attribute contains the data generated by the AE/CSE.	OA
contentSize	1	RO	Size in bytes of the *content* attribute.	NA
sequenceNr	0..1	WO	This attribute contains the data sequence number generated by the AE/CSE	OA

9.6.38 Resource Type *role*

The <*role*> resource represents a role that is assigned to an AE or CSE.

The <*role*> resource shall contain the child resources specified in table 9.6.38-1.

Table 9.6.38-1: Child resources of *<role>* resource

Child Resources of *<role>*	Child Resource Type	Multiplicity	Description
[variable]	*<subscription>*	0..n	See clause 9.6.8
[variable]	*<transaction>*	0..n	See clause 9.6.48

The <*role*> resource shall contain the attributes specified in table 9.6.38-2.

Table 9.6.38-2: Attributes of <*role*> resource

Attributes of <*role*>	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
labels	0..1 (L)	RW	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.
roleID	1	WO	The identifier of the role.
issuer	1	WO	The identifier of the entity that is responsible for assigning the role to the AE or CSE.
holder	1	WO	The identifier of the AE or CSE that the role is assigned
notBefore	1	WO	Start time of the role can be used for access control.
notAfter	1	WO	End time of the role can be used for access control.
roleName	0..1	WO	Human readable name of the <*role*>.
tokenLink	0..1	RW	This attribute contains a reference to a token in which this role assignment is described.

9.6.39 Resource Type *token*

The <*token*> resource is used for storing a token that is issued to an AE or CSE. Details of the token may also be stored here in plaintext.

The <*token*> resource shall contain the child resources specified in table 9.6.39-1.

Table 9.6.39-1: Child resources of <*token*> resource

Child Resources of <*token*>	Child Resource Type	Multiplicity	Description
[variable]	<*subscription*>	0..n	See clause 9.6.8
[variable]	<*transaction*>	0..n	See clause 9.6.48

The <*token*> resource shall contain the attributes specified in table 9.6.39-2.

Table 9.6.39-2: Attributes of <*token*> resource

Attributes of <*token*>	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
labels	0..1 (L)	RW	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.
tokenID	1	WO	The identifier of the token.
tokenObject	1	WO	Used to store the token. See clause 7.3.2.4 ETSI TS 118 103 [2] for further details of a token.
version	0..1	WO	Version of the token.
issuer	0..1	WO	The identifier of the entity that is responsible for issuing the token to the AE or CSE.
audience	0..1 (L)	WO	List of identifiers of the CSEs expected to accept the token.
holder	0..1	WO	The identifier of the AE or CSE to which the token is issued.
notBefore	0..1	WO	Start time of the token can be used for access control.
notAfter	0..1	WO	End time of the token can be used for access control.
tokenName	0..1	WO	Human readable name of the <*token*>.
permissions	0..1 (L)	WO	List of token permissions associated with the token. The structure of token permission is specified in table 9.6.39-3.
extension	0..1	WO	Extension information held by the token, e.g. application-specific information.

The structure of token permission is specified in table 9.6.39-3.

Table 9.6.39-3: Structure of token permission

Element	Multiplicity	Description	Note
resourceIDs	0..1	The resources to which this permission applies. If the privileges element is present, then this element shall be present.	
privileges	0..1	A set of access control rules applicable to the identified resources (for the identified holder)	At least one of these shall be present.
roleIDs	0..1	A set of role IDs applicable to the identified resources (for the identified holder).	

9.6.40 Resource Type *dynamicAuthorizationConsultation*

The <*dynamicAuthorizationConsultation*> resource shall be used by a CSE to perform consultation-based dynamic access control to resources as specified in the present document and in ETSI TS 118 103 [2].

The <*dynamicAuthorizationConsultation*> resource is comprised of configuration information that a resource Hosting CSE may use to determine whether or not to initiate a consultation-based dynamic authorization request.

For a resource that is not of <*dynamicAuthorizationConsultation*> resource type, the common attribute *dynamicAuthorizationConsultationIDs* for such resources (defined in table 9.6.1.3.2-1) may contain a list of identifiers which link that resource to <*dynamicAuthorizationConsultation*> resources.

The <*dynamicAuthorizationConsultation*> resource shall contain the child resources specified in table 9.6.40-1.

Table 9.6.40-1: Child resources of <*dynamicAuthorizationConsultation*> resource

Child Resources of <*dynamicAuthorization Consultation*>	Child Resource Type	Multiplicity	Description
[variable]	<*subscription*>	0..n	See clause 9.6.8
[variable]	<*transaction*>	0..n	See clause 9.6.48

The <*dynamicAuthorizationConsultation*> resource shall contain the attributes specified in table 9.6.40-2.

Table 9.6.40-2: Attributes of <*dynamicAuthorizationConsultation*> resource

Attributes of <*dynamicAuthorizationConsultation*>	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
labels	0..1 (L)	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.
dynamicAuthorizationEnabled	1	RW	Controls whether consultation-based dynamic authorization is enabled or disabled. If disabled, Hosting CSE shall NOT initiate consultation-based dynamic authorization. Valid values are "TRUE" or "FALSE".
dynamicAuthorizationPoA	1 (L)	RW	A list of contact URIs of supporting consultation-based dynamic authorization.
dynamicAuthorizationLifetime	0..1	RW	The preferred lifetime of dynamic access control privileges that CSE shall specify as a parameter when issuing a consultation-based dynamic authorization request.

9.6.41 Resource Type *authorizationDecision*

An <*authorizationDecision*> resource represents an access control decision point that is responsible for making access control decisions. <*authorizationDecision*> resources are the child resources of a <*CSEBase*> resource. When an UPDATE request addresses an <*authorizationDecision*> resource, the Hosting CSE may act as a Policy Decision Point (PDP) that is defined in ETSI TS 118 103 [2]. The PDP shall make an access control decision according to the access control policies and provide the access control decision in the response of the request.

The resource specific attributes of <*authorizationDecision*> resource type are classed into two categories according to their usage. The *decision* and *status* attributes are used for describing an access control decision responses and the others are used for describing access control decision requests.

An access control decision request shall be provided to a PDP through an UPDATE operation on an <*authorizationDecision*> resource that represents the PDP, and the updated resource attributes shall be the attributes used for describing access control decision request parameters. The mandatory and optional parameters used for describing an access control decision request are specified in ETSI TS 118 103 [2]. When an UPDATE request that represents a valid access control decision request addresses an <*authorizationDecision*> resource, the PDP procedure bound to the <*authorizationDecision*> resource shall be triggered. The PDP procedure shall make an access control decision and then update the *decision* and/or *status* attributes. The *decision* and/or *status* attributes that represents an access control decision response shall be returned to the requester in the UPDATE response. An UPDATE request that does not represent a valid access control decision request shall not trigger the bound PDP procedure. Before triggering a PDP procedure, accessing an <*authorizationDecision*> resource is governed by the access control policies assigned to this resource.

For the lifecycle management of <*authorizationDecision*> resources, see ETSI TS 118 103 [2].

The <*authorizationDecision*> resource shall contain the child resources specified in table 9.6.41-1.

Table 9.6.41-1: Child resources of <*authorizationDecision*> resource

Child Resources of <*role*>	Child Resource Type	Multiplicity	Description
[variable]	<*subscription*>	0..n	See clause 9.6.8
[variable]	<*transaction*>	0..n	See clause 9.6.48

The <*authorizationDecision*> resource shall contain the attributes specified in table 9.6.41-2.

Table 9.6.41-2: Attributes of <*authorizationDecision*> resource

Attributes of <*role*>	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
labels	0..1 (L)	RW	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
decision	0..1	RO	Authorization decision for an access control decision request. See clause 7 in ETSI TS 118 103 [2].
status	0..1	RO	Status of an authorization evaluation process. See clause 7 in ETSI TS 118 103 [2].
to	0..1	RW	Same as the *To* parameter in the request sent from the Originator to the Hosting CSE. See clause 7 in ETSI TS 118 103 [2].
from	0..1	RW	Same as the *From* parameter in the request sent from the Originator to the Hosting CSE. See clause 7 in ETSI TS 118 103 [2].
operation	0..1	RW	Same as the *Operation* parameter in the request sent from the Originator to the Hosting CSE. See clause 7 in ETSI TS 118 103 [2].
requestedResourceType	0..1	RW	Resource type that the Originator wants to create. See clause 7 in ETSI TS 118 103 [2].
filterUsage	0..1	RW	Same as the *filterUsage* parameter in the request sent from the Originator to the Hosting CSE. See clause 7 in ETSI TS 118 103 [2].
roleIDs	0..1 (L)	RW	Same as the *Role IDs* parameter in the request sent from the Originator to the Hosting CSE. See clause 7 in ETSI TS 118 103 [2].
tokenIDs	0..1 (L)	RW	Same as the *Token IDs* parameter in the request sent from the Originator to the Hosting CSE. See clause 7 in ETSI TS 118 103 [2].
tokens	0..1 (L)	RW	Same as the *Tokens* parameter in the request sent from the Originator to the Hosting CSE. See clause 7 in ETSI TS 118 103 [2].
requestTime	0..1	RW	Time stamp when the request message was received at the hosting CSE. Obtained by the hosting CSE's system time clock. See clause 7 in ETSI TS 118 103 [2].
originatorLocation	0..1	RW	Location information about the Originator of the request. Obtained over the Mcn reference point. See clause 7 in ETSI TS 118 103 [2].
originatorIP	0..1	RW	IP source address associated with the IP packets that carry the request message. Obtained over the Mcn reference point. See clause 7 in ETSI TS 118 103 [2].

9.6.42 Resource Type *authorizationPolicy*

An <*authorizationPolicy*> resource represents an access control policy retrieval point that is responsible for retrieving access control policies. <*authorizationPolicy*> resources are the child resources of a <*CSEBase*> resource. When an UPDATE request addresses an <*authorizationPolicy*> resource, the Hosting CSE acts as a Policy Retrieval Point (PRP) as defined in ETSI TS 118 103 [2]. The PRP shall retrieve the applicable access control policies according to the access control policy request and provide the retrieved access control policies in the UPDATE response.

The resource specific attributes of <*authorizationPolicy*> resource type are classed into two categories according to their usage. The *policies*, *combiningAlgorithm* and *status* attributes are used for describing access control policy responses. The others are used for describing access control policy requests.

An access control policy request shall be provided to a PRP through an UPDATE operation on an <*authorizationPolicy*> resource that represents the PRP, and the updated resource attributes shall be the attributes used for describing access control policy request parameters. The mandatory and optional parameters used for describing an access control policy request are specified in ETSI TS 118 103 [2]. When an UPDATE request that represents a valid access control policy request addresses an <*authorizationPolicy*> resource, the PRP procedure bound to the <*authorizationPolicy*> resource shall be triggered. The PRP procedure shall retrieve applicable access control policies and then update the *policies*, *combiningAlgorithm* and/or *status* attributes. The *policies*, *combiningAlgorithm* and/or *status* attributes that represents an access control policies response shall be returned to the requester in the UPDATE response. An UPDATE request that does not represent a valid access control policy request shall not trigger the bound PRP procedure. Before triggering a PRP procedure, accessing an <*authorizationPolicy*> resource is governed by the access control policies assigned to this resource.

For the lifecycle management of <*authorizationPolicy*> resources, see ETSI TS 118 103 [2].

The <*authorizationPolicy*> resource shall contain the child resources specified in table 9.6.42-1.

Table 9.6.42-1: Child resources of <*authorizationPolicy*> resource

Child Resources of <*role*>	Child Resource Type	Multiplicity	Description
[variable]	<*subscription*>	0..n	See clause 9.6.8
[variable]	<*transaction*>	0..n	See clause 9.6.48

The <*authorizationPolicy*> resource shall contain the attributes specified in table 9.6.42-2.

Table 9.6.42-2: Attributes of <*authorizationPolicy*> resource

Attributes of <*role*>	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
labels	0..1 (L)	RW	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
policies	0..1 (L)	RO	List of access control policies for an access control policy request. Each access control policy contains a set of access control rules as specified in clause 9.6.2. See clause 7 in ETSI TS 118 103 [2].
combiningAlgorithm	0..1	RO	Algorithm used for combining multiple access control policies. See clause 7 in ETSI TS 118 103 [2].
status	0..1	RO	Status of retrieving access control policies. See clause 7 in ETSI TS 118 103 [2].
to	0..1	RW	Same as the *To* parameter in the access control decision request. See clause 7 in ETSI TS 118 103 [2].

9.6.43 Resource Type *authorizationInformation*

The <*authorizationInformation*> resource represents an access control information retrieval point that is responsible for retrieving access control information. <*authorizationInformation*> resources are the child resources of a <*CSEBase*> resource. When an UPDATE request addresses an <*authorizationInformation*> resource, the Hosting CSE acts as a Policy Information Point (PIP) as defined in ETSI TS 118 103 [2]. The PIP shall retrieve the required access control information according to the access control information request and provide the access control information in the UPDATE response.

The resource specific attributes and child resources of <*authorizationInformation*> resource type are classed into two categories according to their usage. The <*role*> and <*token*> resources and *status* attribute are used for describing access control information responses. The others are used for describing access control information requests.

An access control information request shall be provided to a PIP through an UPDATE operation on an <*authorizationInformation*> resource that represents the PIP, and the updated resource attributes shall be the attributes used for describing access control information request parameters. The mandatory and optional parameters used for describing an access control information request are specified in ETSI TS 118 103 [2]. When an UPDATE request that represents a valid access control information request addresses an <*authorizationInformation*> resource, the PIP procedure bound to the <*authorizationInformation*> resource shall be triggered. The PIP procedure shall retrieve required access control information and then create corresponding <*role*> and/or <*token*> child resources and/or update *status* attributes. The <*role*> and/or <*token*> child resources and/or *status* attributes that represents an access control information response shall be returned to the requester in the UPDATE response. An UPDATE request that does not represent a valid access control information request shall not trigger the bound PIP procedure. Before triggering a PIP procedure, accessing an <*authorizationInformation*> resource is governed by the access control policies assigned to this resource.

For the lifecycle management of <*authorizationInformation*> resources, see ETSI TS 118 103 [2].

The <*authorizationInformation*> resource shall contain the child resources specified in table 9.6.43-1.

Table 9.6.43-1: Child resources of <*authorizationInformation*> resource

Child Resources of <*role*>	Child Resource Type	Multiplicity	Description
[variable]	<*role*>	0..n	See clause 9.6.38
[variable]	<*token*>	0..n	See clause 9.6.39
[variable]	<*subscription*>	0..n	See clause 9.6.8
[variable]	<*transaction*>	0..n	See clause 9.6.48

The <*authorizationInformation*> resource shall contain the attributes specified in table 9.6.43-2.

Table 9.6.43-2: Attributes of <*authorizationInformation*> resource

Attributes of <*role*>	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
labels	0..1 (L)	RW	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
status	0..1	RO	Status of retrieving access control information. See clause 7 in ETSI TS 118 103 [2].
from	0..1	RW	Same as the *From* parameter in the request. See clause 7 in ETSI TS 118 103 [2].
roleIDs	0..1 (L)	RW	Same as the *Role IDs* parameter in the request. See clause 7 in ETSI TS 118 103 [2].
tokenIDs	0..1 (L)	RW	Same as the *Token IDs* parameter in the request. See clause 7 in ETSI TS 118 103 [2].

9.6.44 Resource Type *localMulticastGroup*

The <*localMulticastGroup*> resource is used by a member hosting CSE to indicate that this CSE is a member of a multicast group. <*localMulticastGroup*> is created as the child resource of the <*CSEBase*> resource. And there may be multiple <*localMulticastGroup*> resources under the same <CSEBase>.

The <*localMulticastGroup*> resource shall contain the child resources specified in table 9.6.44-1.

Table 9.6.44-1: Child resources of <*localMulticastGroup*> resource

Child Resources of <*localMulticastGroup*>	Child Resource Type	Multiplicity	Description
[variable]	<*transaction*>	0..n	See clause 9.6.48

The <*localMulticastGroup*> resource shall contain the attributes specified in table 9.6.44-2.

Table 9.6.44-2: Attributes of <*localMulticastGroup*> resource

Attributes of <*localMulticastGroup*>	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
labels	0..1 (L)	RW	See clause 9.6.1.3.
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.
announceTo	0..1(L)	RW	See clause 9.6.1.3.
announcedAttribute	0..1(L)	RW	See clause 9.6.1.3.

Attributes of <*localMulticastGroup*>	Multiplicity	RW/ RO/ WO	Description
externalGroupID	0..1	RW	It is used by an M2M Service Provider (M2M SP) when services targeted to a group of M2M Devices are requested from the Underlying Network. It is assumed to be a globally unique ID exposed by the underlying network to identify a group of M2M Devices (e.g. ASN, MN) that is mapped to an internally used identifier for group related services.
multicastAddress	1	RW	The multicast address assigned by the Group Hosting CSE for the Member Hosting CSE to join the multicast group. The procedure of multicast address assignment is specified in IETF RFC 5771 [11] and IETF RFC 2375 [12].
multicastGroupFanoutTarget	1	RW	Represents a unique fan out target that a Member Hosting CSE shall accept and process incoming multicast requests for this multicast group. It is assigned by the Group Hosting CSE to identify the collection of all the member resources of this multicast group across different member Hosting CSEs. It shall be used in the ***To*** parameter of the multicast request sent to the member Hosting CSEs. If a Member Hosting CSE receives a request while listening on the *multicastAddress* defined for this multicast group, and the incoming request has a specified target that matches this attribute, then the request shall be processed by the Member Hosting CSE. Otherwise, the request shall be ignored.
memberList	1(L)	RW	List of local member resourceIDs in the multicast group which are hosted on the same member hosting CSE. Each member resource ID corresponds to a member resource. A <*localMulticastGroup*> resource with an empty member list shall not be allowed.
responseTarget	1	RW	Indicates the target that the multicast member hosting CSE sends the notification to when finishing the operation in the multicast message from the group hosting CSE. The default value should be the CSE-ID of the group hosting CSE.
responseTimeWindow	0..1	RW	Upon receiving a multicast request, this attribute defines the upper bound on the amount of delay the Member Hosting CSE shall wait before sending a response message. The Member Hosting CSE shall wait a randomized time that is less than the value of this attribute. This randomized delay helps prevent network congestion caused by multiple Member Hosting CSEs responding at the same time as one another.
TMGI	0..1	RW	The Temporary Mobile Group Identity is allocated to identify the MBMS bearer service as specified in ETSI TS 123 246 [i.32]. It is used to uniquely identify the 3GPP multicast or broadcast message with *externalGroupID* together.

9.6.45 Resource Type *AEContactList*

An <*AEContactList*> resource shall contain <*AEContactListPerCSE*> child resources, one for each CSE that has sent a NOTIFY request to the CSE about the creation, update, or deletion of a resource that references an Application Entity resource identifier. The <*AEContactList*> resource shall only be created as a child of <*CSEBase*> in the IN-CSE.

The <*AEContactList*> resource shall contain the child resources specified in table 9.6.45-1.

Table 9.6.45-1: Child resources of <*AEContactList*> resource

Child Resources of <*AEContactList*>	Child Resource Type	Multiplicity	Description
[variable]	<*subscription*>	0..n	See clause 9.6.8
[variable]	<*AEContactListPerCSE*>	0..n	See clause 9.6.46
[variable]	<*transaction*>	0..n	See clause 9.6.48

The <*AEContactList*> resource shall contain the attributes specified in table 9.6.45-2.

Table 9.6.45-2: Attributes of <*AEContactList*> resource

Attributes of <*AEContactList*>	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	RO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RO	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RO	See clause 9.6.1.3.
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
labels	0..1 (L)	RO	See clause 9.6.1.3.
numberImpactedCSEs	1	RO	The number of Hosting CSEs that have reported that they have a reference to an Application Entity resource identifier.

9.6.46 Resource Type *AEContactListPerCSE*

An <*AEContactListPerCSE*> resource shall represent information about a CSE that has resources that reference an Application Entity resource identifier (SP-relative-Resource-IDs of an AE). For example, these Application Entity resource identifiers may occur in announcement links, notification targets, group member IDs, or in the *OriginatorID* list of the *accessControlOriginators* parameter tied to an <accessControlPolicy> resource. The <*AEContactListPerCSE*> resource shall only be created in the IN-CSE.

The <*AEContactListPerCSE*> resource shall contain the attributes specified in table 9.6.46-1.

Table 9.6.46-1: Attributes of *<AEContactListPerCSE>* resource

Attributes of *<AContactListPerCSE>*	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	RO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RO	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RO	See clause 9.6.1.3.
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
labels	0..1 (L)	RO	See clause 9.6.1.3.
CSE-ID	1	RO	The identifier of the Hosting CSE which has a reference to an Application Entity resource identify (SP-relative-Resource-ID that points to an AE). Hosting CSEs notify the IN-CSE when they have a reference to an <AE> resource through e.g. announcements, notification targets, group member IDs, <accessControlPolicy> resource *OriginatorID* lists.
AE-IDList	0..1(L)	RO	List of Application Entity resource identifiers hosted on CSE with identifier CSE-ID.

9.6.47 Resource Type *transactionMgmt*

The <*transactionMgmt*> resource is used to initiate and manage the atomic and consistent processing of a transaction consisting of multiple oneM2M request primitives.

The <*transactionMgmt*> resource supports the child resources specified in table 9.6.47-1.

Table 9.6.47-1: Child resources of <transactionMgmt> resource

Child Resources of *<transactionMgmt>*	Child Resource Type	Multiplicity	Description
[variable]	*<subscription>*	0..n	See clause 9.6.8

The <*transactionMgmt*> resource supports the attributes specified in table 9.6.47-2.

Table 9.6.47-2: Attributes of <transactionMgmt> resource

Attributes of <*transactionMgmt*>	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
expirationTime	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.
labels	0..1 (L)	RW	See clause 9.6.1.3.
creator	1	RO	This attribute is configured with the identifier of the entity that originated the request to create this <*transactionMgmt*> resource.
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.
transactionLockTime	0..1	WO	This attribute contains timing information that the <*transactionMgmt*> Hosting CSE uses to configure the corresponding *transactionLockTime* attribute of each <*transaction*> resource it creates for this transaction. If this attribute is not set, the <*transactionMgmt*> Hosting CSE may configure the *transactionLockTime* attribute of each <*transaction*> resource with a default value based on local policy.
transactionExecuteTime	0..1	WO	This attribute contains timing information that the <*transactionMgmt*> Hosting CSE uses to configure the corresponding *transactionExecuteTime* attribute of each <*transaction*> resource it creates for this transaction. If this attribute is not set, the <*transactionMgmt*> Hosting CSE may configure the *transactionExecuteTime* attribute of each <*transaction*> resource with a default value based on local policy.
transactionCommitTime	0..1	WO	This attribute contains timing information that the <*transactionMgmt*> Hosting CSE use to configure the corresponding *transactionCommitTime* attribute of each <*transaction*> resource it creates for this transaction. If this attribute is not set, the <*transactionMgmt*> Hosting CSE may configure the *transactionCommitTime* attribute of each <*transaction*> resource with a default value based on local policy.
transactionExpirationTime	0..1	WO	This attribute contains timing information of when the transaction is set to expire. The <*transactionMgmt*> Hosting CSE shall use this value to configure the *expirationTime* attribute of the <*transaction*> resources it creates. If this attribute is not set, the <*transactionMgmt*> Hosting CSE may configure the *expirationTime* attribute of each <*transaction*> resource with a default value based on local policy. If the transaction fails to complete before this time elapses, the <*transactionMgmt*> Hosting CSE shall abort the transaction.
transactionMode	0..1	WO	This attribute is used by the Hosting CSE to determine whether it is responsible for controlling the execution of the transaction (via the *transactionControl* attribute) or whether the creator is responsible for controlling it. The allowed values are: • CSE_CONTROLLED (Default) • CREATOR_CONTROLLED

Attributes of <*transactionMgmt*>	Multiplicity	RW/ RO/ WO	Description
transactionLockType	0..1	WO	This attribute indicates the type of lock that is required on the targeted resource in order to perform the transaction. The <*transactionMgmt*> Hosting CSE shall use the value in this attribute to configure the corresponding *transactionLockType* attribute of each <*transaction*> resource it creates. The following are the supported types of locks: • BLOCK_ALL - Block oneM2M request primitives not associated with this transaction from performing any CRUD operations on the targeted resource while it is locked for this transaction. This shall be the default value if this attribute is not configured. • ALLOW_RETRIEVES - Block oneM2M request primitives not associated with this transaction from performing any operation except RETRIEVE on the targeted resource while it is locked for this transaction.
transactionControl	0..1	RW	This attribute is used to control the state of the transaction. The allowed values are: • INITIAL (default) • LOCK • EXECUTE • COMMIT • ABORT If the *transactionMode* is set to "CSE_CONTROLLED", then only the <*transactionMgmt*> Hosting CSE shall be allowed to update this attribute. If the *transactionMode* is set to "CREATOR_CONTROLLED", then only the *creator* shall be allowed to update this attribute. This attribute should either not be present in a <transactionMgmt> create request or have a value of "INITIAL".

Attributes of <*transactionMgmt*>	Multiplicity	RW/ RO/ WO	Description
transactionState	1	RO	This attribute contains the current state of the transaction. Only the <*transactionMgmt*> Hosting CSE shall be allowed to update this attribute. It calculates the value of this attribute based on collective *transactionState* of the individual <*transaction*> resources associated with this <*transactionMgmt*> resource. The allowed values are: • INITIAL • LOCKED • EXECUTED • COMMITTED • ERROR • ABORTED To update this attribute to a new state, all of the *transactionState* attributes of the individual <*transaction*> resources should be consistent and reflect the value of the new state. The exception is updating this attribute to "ERROR". If any of the *transactionState* attributes of the individual <*transaction*> resources have a value of "ERROR", then the <*transactionMgmt*> Hosting CSE shall update *transactionState* of this <*transactionMgmt*> resource to "ERROR". Before doing so however, the <*transactionMgmt*> Hosting CSE shall check if the *transactionMaxRetries* attribute is configured with a non-zero value and if so whether the retry limit has been exhausted. If not exhausted, the <*transactionMgmt*> Hosting CSE shall attempt to retry the transaction. If the *transactionMaxRetries* attribute is configured with a zero value or the retry limit is exhausted, then this attribute shall be set to "ERROR". A creator may subscribe to this attribute to receive notifications of changes to *transactionState*.
transactionMaxRetries	0..1	RW	If set, this attribute shall be used by the <*transactionMgmt*> Hosting CSE to determine the max number of times it may attempt to retry a transaction when detecting an "ERROR" *transactionState* from one or more <*transaction*> Hosting CSEs. If not set or if the max number of retries is exhausted, and the <*transactionMgmt*> Hosting CSE detects an "ERROR" *transactionState* from one or more <*transaction*> Hosting CSEs, then the *transactionState* of the <transactionMgmt> resource shall be updated to "ERROR".
transactionMgmtHandling	0..1	RW	This attribute is used by the <*transactionMgmt*> Hosting CSE to determine whether to persist or delete the <*transactionMgmt*> resource after its completion (i.e. *transactionState* is set to "COMMITTED" or "ABORTED"). The allowed values are: • DELETE (default) • PERSIST If set to "PERSIST" the <*transactionMgmt*> resource shall be deleted when the *expirationTime* elapses.
requestPrimitives	1(L)	WO	This attribute contains an aggregated list of oneM2M request primitives associated with this transaction. When processing this transaction, the <*transactionMgmt*> Hosting CSE shall create a corresponding <*transaction*> resource for each oneM2M request primitive in this list. Each <*transaction*> resource shall be created as a child resource under the resource targeted by its respective request primitive.

Attributes of <*transactionMgmt*>	Multiplicity	RW/ RO/ WO	Description
responsePrimitives	1(L)	RO	This attribute contains an aggregated list of oneM2M response primitives associated with this transaction. This attribute shall be updated by the <*transactionMgmt*> Hosting CSE and includes the individual *responsePrimitive* attributes received from the <*transaction*> Hosting CSE(s). The creator may subscribe to this attribute to receive notifications each time this attribute is updated with new response primitive(s).

9.6.48 Resource Type *transaction*

The <*transaction*> resource is used to initiate and manage the atomic and consistent processing of a single oneM2M request primitive of a oneM2M transaction.

With the exception of the <*request*>, <*delivery*>, <*transaction*> and <*transactionMgmt*> resources, a <*transaction*> resource may be created as a child resource of any resource targeted by a oneM2M transaction. A <*transaction*> create request may be originated by a CSE that hosts an associated <*transactionMgmt*> resource. Alternatively, a <*transaction*> resource can be used independent of <*transactionMgmt*> resource for the case where an application wishes to create individual <transaction> resources itself.

In order to process a transaction across multiple targeted resources, multiple <*transaction*> resources are created on the corresponding Hosting CSE(s) of the targeted resources. Depending on the intention of the application that is coordinating the transaction, each <*transaction*> resource represents a oneM2M *requestPrimitive* that needs to be performed on a targeted resource. The *requestPrimitive* of different <*transaction*> resources may be different or the same based on the desired outcome of the transaction. The time related attributes of different <*transaction*> resources (i.e. *transactionLockTime*, *transactionExecuteTime* and *transactionCommitTime)* may be different or the same as well to coordinate the desired order or schedule of the transaction.

The <*transaction*> resource supports the child resources specified in table 9.6.48-1.

Table 9.6.48-1: Child resources of <transaction> resource

Child Resources of <*transaction*>	Child Resource Type	Multiplicity	Description
[variable]	<*subscription*>	0..n	See clause 9.6.8

The <*transaction*> resource supports the attributes specified in table 9.6.48-2.

Table 9.6.48-2: Attributes of <transaction> resource

Attributes of <*transaction*>	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
expirationTime	1	WO	See clause 9.6.1.3. The value of the *expirationTime* serves as the expiration time of the transaction.
parentID	1	RO	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RO	See clause 9.6.1.3.
labels	0..1 (L)	RO	See clause 9.6.1.3.
creator	1	RO	This attribute is configured with the CSE-ID of the <*transactionMgmt*> Hosting CSE or AE-ID of the AE that created this <*transaction*> resource.
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.
transactionID	1	WO	This attribute is configured with the identifier of the transaction.

Attributes of <*transaction*>	Multiplicity	RW/ RO/ WO	Description
			This attribute is configured by the <*transactionMgmt*> Hosting CSE with the resource identifier of the <transactionMgmt> resource. If an AE creates the <transaction> resource, then this attribute is configured with an AE specified identifier.
transactionControl	1	RW	This attribute shall be used to configure or change the state of the transaction. The allowed values are: • LOCK • EXECUTE • COMMIT • ABORT Only the *creator* of this <*transaction*> resource is allowed to update this attribute.
transactionState	1	RO	This attribute contains the current state of the transaction. Only the Hosting CSE of this <transaction> resource shall be allowed to update this attribute. The allowed values are: • LOCKED • EXECUTED • COMMITTED • ERROR • ABORTED
transactionLockTime	0..1	WO	This attribute contains timing information that the <*transaction*> Hosting CSE itself shall use to schedule when it sets *transactionControl* to LOCK rather than relying on an UPDATE request to set the value of *transactionControl* to LOCK.
transactionExecuteTime	0..1	WO	This attribute contains timing information that the <*transaction*> Hosting CSE itself shall use to schedule when it sets *transactionControl* to EXECUTE rather than relying on an UPDATE request to set the value of *transactionControl* to EXECUTE. If the targeted resource is not locked at this scheduled time, the <transaction> Hosting CSE shall first lock the resource before executing the request primitive.
transactionCommitTime	0..1	WO	This attribute contains timing information that the <*transaction*> Hosting CSE itself shall use to schedule when it sets *transactionControl* to COMMIT rather than relying on an UPDATE request to set the value of *transactionControl* to COMMIT. If the targeted resource is not locked or the request primitive has not yet been executed at this scheduled time, the <transaction> Hosting CSE shall first lock the resource and execute the request primitive at this time before committal.
transactionLockType	0..1	WO	This attribute indicates the type of lock that is required on the targeted resource in order to perform the transaction. The following are the supported types of locks: • BLOCK_ALL - Block oneM2M request primitives not associated with this transaction from performing any CRUD operations on the targeted resource while it is locked for this transaction. This shall be the default value if this attribute is not configured. • ALLOW_RETRIEVES - Block oneM2M request primitives not associated with this transaction from performing any operation except RETRIEVE on the targeted resource while it is locked for this transaction.

Attributes of <*transaction*>	Multiplicity	RW/ RO/ WO	Description
requestPrimitive	1	WO	This attribute contains the request primitive to be executed on the parent of this <transaction> resource.
responsePrimitive	1	RO	This attribute contains the oneM2M response primitive associated with this transaction. This attribute is updated by the <*transaction*> Hosting CSE after it executes the *requestPrimitive* on the parent resource.

9.6.49 Resource Type *triggerRequest*

The <*triggerRequest*> resource is used to initiate a device trigger request. This resource type shall only be instantiated on an IN-CSE.

The successful creation of a <*triggerRequest*> resource results in the IN-CSE initiating a trigger request to a targeted device (e.g. 3GPP UE). A pending trigger request can be replaced with a new trigger request by updating the <*triggerRequesst*> resource. A pending trigger request can be cancelled by deleting the <*triggerRequesst*> resource.

The <*triggerRequest*> resource shall contain the child resource specified in table 9.6.49-1.

Table 9.6.49-1: Child resources of <*triggerRequest*> resource

Child Resources of <*triggerRequest*>	Child Resource Type	Multiplicity	Description
[variable]	<*subscription*>	0..n	See clause 9.6.8 of the present document

The <*triggerRequest*> resource shall contain the attributes specified in table 9.6.49-2.

Table 9.6.49-2: Attributes of <*triggerRequest*> resource

Attributes of <*triggerRequest*>	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
labels	0..1 (L)	RW	See clause 9.6.1.3.
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.
M2M-Ext-ID	1	WO	M2M External Identifier of the device being triggered. See clause 7.1.8. This attribute shall be configured by the Originator when the resource is created.
Trigger-Recipient-ID	1	RW	Trigger-Recipient-ID of the ASN/MN-CSE or ADN-AE that is hosted on the device being triggered. See clause 7.1.10. This attribute shall be configured by the Originator when the resource is created and may also be updated when performing a trigger replace procedure. See clause 8.3.3.2.2. This attribute is application port ID for device trigger to uniquely identify the triggered application.
triggerPurpose	1	RW	The purpose of the trigger. See clause 8.3.3.2.1.

Attributes of <*triggerRequest*>	Multiplicity	RW/ RO/ WO	Description
			This attribute may be configured by the Originator when the resource is created and may also be updated when performing a trigger replace procedure. The allowed values are: • establishConnection • enrolmentRequest • registrationRequest • executeCRUD If not specified by the Originator, the default is "establishConnection".
triggerStatus	1	RO	The status of the trigger request. The Hosting CSE shall control the value of this attribute. The following values are valid values. • PROCESSING • ERROR-NSE-NOT-FOUND • TRIGGER-TRIGGERED • TRIGGER-DELIVERED • TRIGGER-FAILED • TRIGGER-REPLACED • TRIGGER-EXPIRED • TRIGGER-UNCONFIRMED • TRIGGER-TERMINATED • TRIGGER-SUCCESS
triggerValidityTime	1	RW	The time duration for which the trigger request is valid. After this time expires, the trigger shall be recalled (i.e. cancelled) by the Hosting CSE. This attribute may be configured by the Originator when the resource is created and may also be updated when performing a trigger replace procedure. See clause 8.3.3.2.2. If this attribute is not set, the CSE may configure the *triggerValidityTime* attribute based on local policy.
triggerInfoAE-ID	0..1	RW	When the *triggerPurpose* is "executeCRUD", this attribute is mandatory otherwise it is not applicable. This attribute is configured with the AE-ID of the ASN/MN-AE that should perform the CRUD operation. When this attribute is configured, the trigger originator shall also configure the *triggerInfoAddress*, *triggerInfoOperation* and *targetedResourceType* attributes.
triggerInfoAddress	0..1	RW	When the triggerPurpose is "enrolmentRequest", this field shall be configured with the absolute URI of the <*MEFBase*> resource of the MEF that the ASN/MN-CSE or ADN-AE shall enrol to. When the triggerPurpose is "establishConnection" and the *pointOfAccess* attribute of the <*AE*> or <*remoteCSE*> representing the ASN/MN-CSE or ADN-AE needs updating, this field shall be configured with an unstructured CSE-Relative-Resource-ID of the <*remoteCSE*> or <*AE*> resource. If this attribute is not provided, the trigger recipient shall establish a network connection with its registrar CSE but not update its pointOfAccess.

Attributes of <*triggerRequest*>	Multiplicity	RW/ RO/ WO	Description
			When the triggerPurpose is "registrationRequest", and this field is provided by the trigger originator, then this field shall be configured with the unstructured CSE-Relative-Resource-ID of the Registrar CSE's <cseBase> resource that the trigger recipient shall register to. When the triggerPurpose is "registrationRequest", and this field is not provided by the trigger originator, the trigger recipient shall register to the Registrar CSE using a pre-provisioned address of the Registrar CSE. The pre-provisioning method is outside the scope of the present document. When the *triggerPurpose* is "executeCRUD", this attribute shall be configured with an unstructured CSE-Relative-Resource-ID of the resource that the ASN/MN-AE shall perform the CRUD operation on. When this attribute is configured for "executeCRUD", the trigger originator shall also configure the *triggerInfoAE-ID*, *triggerInfoOperation* and *targetedResourceType* attributes.
triggerInfoOperation	0..1	RW	When the *triggerPurpose* is "executeCRUD", this attribute is mandatory otherwise it is not applicable. This attribute is configured with the CRUD operation that the ASN/MN-AE should perform on the targeted resource specified by *triggerInfoAddress*. When this attribute is configured, the trigger originator shall also configure the *triggerInfoAE-ID, triggerInfoAddress* and *targetedResourceType* attributes.
targetedResourceType	0..1	RW	When the *triggerPurpose* is "executeCRUD", this attribute is mandatory otherwise it is not applicable. This attribute is configured with the resource type of the targeted resource specified by *triggerInfoAddress*. When this attribute is configured, the trigger originator shall also configure the *triggerInfoAE-ID, triggerInfoAddress* and *triggerInfoOperation* attributes.
triggerReference	0..1	RO	This attribute is a reference number which is allocated by the IN-CSE of a transaction and is used in all subsequent messages related to that transaction to support device triggering.

9.6.50 Resource type *ontologyRepository*

An <*ontologyRepository*> resource is a child resource of the <*CSEBase*> resource.

The <*ontologyRepository*> resource may have one or multiple <*ontology*> child resources to represent and manage both internal and external ontologies in the oneM2M system. By performing the CRUD operations on the <*ontology*> resources, explicit ontologies can be imported (created), discovered, retrieved, updated and deleted inside the oneM2M system, and can be used for semantic validation when they are referenced by <*semanticDescriptor*> resources. The details of <*ontology*> resource are specified in clause 9.6.51.

The <*ontologyRepository*> resource may also contain a (virtual) child resource <*semanticValidation*> as the interface to accept semantic validation request from an AE or a CSE. Upon receiving an Update request with <*semanticDescriptor*> resource representation addressing the <*semanticValidation*> resource, the hosting CSE shall perform the semantic validation against the <*semanticDescriptor*> resource received in the request. The details of <*semanticValidation*> resource are specified in clause 9.6.52.

The <*ontologyRepository*> resource shall contain the child resources as specified in table 9.6.50-1.

Table 9.6.50-1: Child resources of *<ontologyRepository>* resource

Child Resources of *<ontologyRepository>*	Child Resource Type	Multiplicity	Description	*<ontologyRepositoryAnnc>* Child Resource Types
[variable]	*<ontology>*	0..n	See clause 9.6.51	*<ontologyAnnc>*
smv	*<semanticValidation>*	1	See clause 9.6.52	*None*
[variable]	*<subscription>*	0..n	See clause 9.6.8	*<subscription>*

The <*ontologyRepository*> resource above contains the attributes specified in table 9.6.50-2.

Table 9.6.50-2: Attributes of *<ontologyRepository>* resource

Attribute Name	Multiplicity	RW/RO/WO	Description	*<ontologyRepository Annc>* Attributes
resourceName	1	WO	See clause 9.6.1.3.	NA
parentID	1	RO	See clause 9.6.1.3.	NA
expirationTime	1	RW	See clause 9.6.1.3.	NA
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.	NA
labels	0..1 (L)	RW	See clause 9.6.1.3.	MA
creationTime	1	RO	See clause 9.6.1.3.	MA
lastModifiedTime	1	RO	See clause 9.6.1.3.	MA
announceTo	0..1 (L)	RW	See clause 9.6.1.3.	NA
announcedAttribute	0..1 (L)	RW	See clause 9.6.1.3.	NA
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.	OA
creator	0..1	RO	See clause 9.6.1.3.	NA

9.6.51 Resource Type *ontology*

The <*ontology*> resource is a child resource of the <*ontologyRepository*> resource. The <*ontology*> resource is used to store the representation of an ontology. This representation may contain ontology descriptions in a variety of formats, given the requirements for re-use of existing ontologies, for support of ontologies available only externally and for support of ontology imported into the system. The ontology description is made available to the semantic-related functions of the oneM2M system provided by applications or CSEs.

Given the possible need to have access to multiple versions of an ontology, and to different formats, a *ontologyFormat* attribute provides information necessary for the system to interpret the information available in the *ontologyContent* attribute.

The <*ontology*> resource above contains the child resources specified in table 9.6.51-1.

Table 9.6.51-1: Child resources of *<ontology>* resource

Child Resources of <semanticDescriptor>	Child Resource Type	Multiplicity	Description	*<ontologyAnnc>* Child Resource Types
[variable]	*<subscription>*	0..n	See clause 9.6.8 where the type of this resource is described.	*<subscription>*

The <*ontology*> resource above contains the attributes specified in table9.6.51-2.

Table 9.6.51-2: Attributes of *<ontology>* resource

Attribute Name	Multiplicity	RW/RO/WO	Description	*<ontologyAnnc>* Attributes
resourceName	1	WO	See clause 9.6.1.3.	NA
parentID	1	RO	See clause 9.6.1.3.	NA
expirationTime	1	RW	See clause 9.6.1.3.	NA
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.	NA
labels	0..1 (L)	RW	See clause 9.6.1.3.	MA
creationTime	1	RO	See clause 9.6.1.3.	MA
lastModifiedTime	1	RO	See clause 9.6.1.3.	MA
announceTo	0..1 (L)	RW	See clause 9.6.1.3.	NA
announcedAttribute	0..1 (L)	RW	See clause 9.6.1.3.	NA
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.	OA
creator	0..1	RO	See clause 9.6.1.3.	NA
description	0..1	RW	Text description of the ontology	OA
ontologyFormat	1	RW	Attribute providing information about the format of the *ontologyContent* attribute. It may indicate the content as: IRI - for an ontology to be accessed via the IRI [i.37] provided in the *ontologyContent* attribute OR File format - for an ontology for which the document is stored in the *ontologyContent* attribute. In this case *ontologyFormat* provides ontology format, e.g. OWL/XML, RDF/Turtle	OA
ontologyContent	1	RW	Depending on the *ontologyFormat* attribute, it may be interpreted either as: The IRI of the corresponding ontology document OR The content of the corresponding ontology document	OA
semanticOpExec	0..1	RW	This attribute cannot be retrieved. Contains a SPARQL query request for execution of semantic operations on the *ontologyContent* attribute e.g. SPARQL update as described in ETSI TS 118 104 [3].	NA

9.6.52 Resource Type *semanticValidation*

The <*semanticValidation*> resource is a virtual resource because it does not have a representation. It is the child resource of a <*ontologyRepository*> resource. It is the interface to accept a semantic validation (Update) request which includes a <*semanticDescriptor*> resource to be validated.

The Hosting CSE shall perform the semantic validation functionality by checking the validity (i.e. conformance) of the triples in the descriptor attribute of the received <*semanticDescriptor*> resource, as well as any linked <*semanticDescriptor*> resources linked by the *relatedSemantics* attribute or the *onem2m:resourceDescriptorLink* annotation property (see clause 10.2.14) in the descriptor attribute, against the referenced ontologies (including the reference ontologies of the linked <*semanticDescriptor*> resources, if any) as pointed by the *ontologyRef* attribute. If the *ontologyRef* attribute is absent, the Hosting CSE shall return an error. The Hosting CSE may need to retrieve (and cache) the linked <*semanticDescriptor*> resources and their referenced ontologies from a remote CSE to perform the semantic validation process. The aspects to be checked in the semantic validation process is specified in oneM2M TS-0034 [14].

The <*semanticValidation*> resource does not have a resource representation by itself and consequently it does not have an *accessControlPolicyIDs* attribute. The <*accessControlPolicy*> resource used for access control policy validation is indicated by the *accessControlPolicyIDs* attribute in the parent <*ontologyRepositoy*> resource.

9.6.53 Resource Type *semanticMashupJobProfile*

The <*semanticMashupJobProfile*> resource represents a Semantic Mashup Job Profile (SMJP). A SMJP describes the profile and necessary information required for a specific mashup service such as input parameters, member resources, mashup function, and output parameters. Based on the profile described in the SMJP, Originators (e.g. AEs) can create corresponding semantic mashup instances where semantic mashup results will be generated and stored.

The <*semanticMashupJobProfile*> resource shall contain the child resources specified in table 9.6.53-1.

Table 9.6.53-1: Child resources of <*semanticMashupJobProfile*> resource

Child Resources of <*semanticMashupJobProfile*>	Child Resource Type	Multiplicity	Description	<*semanticMashupJobProfileAnnc*> Child Resource Types
<*variable*>	<*semanticMashupInstance*>	0..n	Represents semantic mashup instances which have been created based on this <*semanticMashupJobProfile*> resource. This child resource is optional as related<*semanticMashupJobProfile*> and <*semanticMashupInstance*> may be stored separately within the resource tree or on different CSEs. See clause 9.6.54.	<*semanticMashupInstance*>, <*semanticMashupInstanceAnnc*>
<*variable*>	<*semanticDescriptor*>	0..1	Describes general semantic information about this <*semanticMashupJobProfile*> resource. See clause 9.6.30.	<*semanticDescriptor*>, <*semanticDescriptorAnnc*>
<*variable*>	<*subscription*>	0..n	Represents subscriptions on this resource. See clause 9.6.8.	<*subscription*>

The <*semanticMashupJobProfile*> resource shall contain the attributes specified in table 9.6.53-2.

Table 9.6.53-2: Attributes of <*semanticMashupJobProfile*> resource

Attributes of <*semanticMashupJobProfile*>	Multiplicity	RW/ RO/ WO	Description	<*semanticMashupJobProfileAnnc*> Attributes
resourceType	1	RO	See clause 9.6.1.3.	NA
resourceID	1	RO	See clause 9.6.1.3.	NA
resourceName	1	WO	See clause 9.6.1.3.	NA
parentID	1	RO	See clause 9.6.1.3.	NA
expirationTime	1	RW	See clause 9.6.1.3.	MA
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.	MA
labels	0..1 (L)	RW	See clause 9.6.1.3.	MA
creationTime	1	RO	See clause 9.6.1.3.	NA
lastModifiedTime	1	RO	See clause 9.6.1.3.	NA
announceTo	0..1 (L)	RW	See clause 9.6.1.3.	NA
announcedAttribute	0..1 (L)	RW	See clause 9.6.1.3.	NA
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.	OA
creator	0..1	RO	See clause 9.6.1.3.	NA

Attributes of <*semanticMashupJobProfile*>	Multiplicity	RW/ RO/ WO	Description	<*semanticMashupJobProfileAnnc*> Attributes
memberFilter	1	RW	Semantically describes the types of member resources which are involved in this semantic mashup job profile <*semanticMashupJobProfile*>. When a <*semanticMashupInstance*> is created based on this <*semanticMashupJobProfile*>, the member resources of the <*semanticMashupInstance*> shall be discovered and selected based on this *memberFilter* attribute. The value of this attribute is a SPARQL query.	OA
smiID	0..1(L)	RO	List of resource identifiers of related semantic mashup instance resources which have been created based on this <*semanticMashupJobProfile*>.	OA
inputDescriptor	0..1	RW	Semantically (i.e. using semantic triples) describes the types of input parameters, which are required in order to use this <*semanticMashupJobProfile*> to create <*semanticMashupInstance*>. Some semantic mashup job profiles may not need input parameters and as such this attribute is optional.	OA
outputDescriptor	1	RW	Semantically (e.g. in semantic triples) describes the types of output parameters generated as semantic mashup results if using this <*semanticMashupJobProfile*>.	OA
functionDescriptor	1	RW	Semantically (e.g. in semantic triples) describes the mashup function of this <*semanticMashupJobProfile*>. The mashup function specifies how semantic mashup results should be generated based on input parameters (defined by the *inputDescriptor* attribute) and original member resources (defined by the *memberFilter* attribute).	OA

9.6.54 Resource Type *semanticMashupInstance*

<*semanticMashupInstance*> models and represents a Semantic Mashup Instance (SMI) resource. A CSE/AE as a Mashup Requestor can request to create <*semanticMashupInstance*> resources at another oneM2M CSE which implements the semantic mashup function. Each created <*semanticMashupInstance*> resource corresponds to a semantic mashup job profile (i.e. a <*semanticMashupJobProfile*> resource); in other words, how the <*semanticMashupInstance*> resource should execute the mashup operation to calculate the mashup result is specified in the corresponding <*semanticMashupJobProfile*> resource. Note that the <*semanticMashupInstance*> and its corresponding <*semanticMashupJobProfile*> resources may be placed at the same CSE or at different CSEs, and the *smjpID* attribute of the <*semanticMashupInstance*> allows locating the corresponding <*semanticMashupJobProfile*> resource. If the <*semanticMashupInstance*> resource has a <*semanticMashupResult*> as its child resource, the Mashup Requestor may use it to retrieve the mashup result.

The <*semanticMashupInstance*> resource shall contain the child resources specified in table 9.6.54-1.

Table 9.6.54-1: Child resources of <*semanticMashupInstance*> resource

Child Resources of <*semanticMashupInstance*>	Child Resource Type	Multiplicity	Description	<*semanticMashupInstanceAnnc*> Child Resource Types
<*variable*>	<*semanticMashupResult*>	0..n	Contains mashup result. A <*semanticMashupInstance*> resource may have multiple <*semanticMashupResult*> child resources, with each mashup result instance resulting from different input parameters and/or member resource values. The hosting CSE generates <*semanticMashupResult*> each time when it executes the mashup operation and calculate a new semantic mashup result.	<*semanticMashupResult*>, <*semanticMashupResultAnnc*>
<*variable*>	<*semanticDescriptor*>	0..1	Describes general semantic information about this <*semanticMashupInstance*> resource.	<*semanticDescriptor*>, <*semanticDescriptorAnnc*>
<*variable*>	<*subscription*>	0..n	Stands for any subscription on this <*semanticMashupInstance*>.	<*subscription*>
msp	<*mashup*>	1	This is a standard oneM2M virtual resource. When a Mashup Requestor sends a RETRIEVE operation on this virtual resource, it triggers a re-calculation and re-generation of the mashup result.	*None*

The <*semanticMashupInstance*> resource shall contain the attributes specified in table 9.6.54-2.

Table 9.6.54-2: Attribute of <*semanticMashupInstance*> resource

Attributes of <*semanticMashupInstance*>	Multiplicity	RW/ RO/ WO	Description	<*semanticMashupInstanceAnnc*> Attributes
resourceType	1	RO	See clause 9.6.1.3.	NA
resourceID	1	RO	See clause 9.6.1.3.	NA
resourceName	1	WO	See clause 9.6.1.3.	NA
parentID	1	RO	See clause 9.6.1.3.	NA
expirationTime	1	RW	See clause 9.6.1.3.	MA
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.	MA
labels	0..1 (L)	RW	See clause 9.6.1.3.	MA
creationTime	1	RO	See clause 9.6.1.3.	NA
lastModifiedTime	1	RO	See clause 9.6.1.3.	NA
announceTo	0..1 (L)	RW	See clause 9.6.1.3.	NA
announcedAttribute	0..1 (L)	RW	See clause 9.6.1.3.	NA
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.	OA
creator	0..1	RO	See clause 9.6.1.3.	NA
smjpID	1	RW	Denotes the identifier (e.g. URI) of the semantic mashup job profile resource <*semanticMashupJobProfile*> which this <*semanticMashupInstance*> is based on.	OA
smjpInputParameter	0..1	RW	Contains the value of all input parameters which are required to calculate the mashup result. Note that the types of these input parameters are specified by the *inputDescriptor* attribute of the corresponding <*semanticMashupJobProfile*> which is denoted by the *smjpID* attribute of this <*semanticMashupInstance*> resource. This attribute is not needed if the corresponding <*semanticMashupJobProfile*> does not have *inputDescriptor* attribute.	OA

Attributes of <*semanticMashupInstance*>	Multiplicity	RW/ RO/ WO	Description	<*semanticMashupInstanceAnnc*> *Attributes*
memberStoreType	1	RW	Indicates the way which member resources should be stored under this <*semanticMashupInstance*>. For example, • If *memberStoreType*="URI Only", the *mashupMember* attribute contains the URI of each member resource; • If *memberStoreType*="URI and Value", the *mashupMember* attribute contains both the URI and the value of each member resource.	OA
mashupMember	0..1(L)	RW	Stores the URI and/or value of each mashup member resource, which is dependent on the value of *memberStoreType* attribute.	OA
resultGenType	1(L)	RW	Describes how the mashup result should be generated using this <*semanticMashupInstance*>. Example values for this attribute could be one of the following or a combination of them. • If *resultGenType*="When SMI Is Created", the semantic mashup result is generated when this <*semanticMashupInstance*> is created by running semantic functions specified by the corresponding <*semanticMashupJobProfile*>. • If *resultGenType*="When Mashup Requestor Requests", the mashup result is to be calculated and generated when requested or triggered by a Mashup Requestor which sends a RETRIEVE operation on the virtual child resource *mashup*. • If *resultGenType*="Periodically", the CSE which hosts <*semanticMashupInstance*> calculates and generates the semantic mashup result periodically based on the *periodForResultGen* attribute. • If *resultGenType*="When A Mashup Member Is Updated", the CSE which hosts <*semanticMashupInstance*> calculates and generates the semantic mashup result whenever there is any update on the *mashupMember* attribute of <*semanticMashupInstance*>.	OA
periodForResultGen	0..1	RW	Is the time period for re-calculating and generating the semantic mashup result. When it is the time to re-calculate the semantic mashup result, the CSE hosting this <*semanticMashupInstance*> needs to retrieve the latest content value of each member resource if it is not obtained yet. This attribute is needed when *resultGenType*="Periodically".	OA

9.6.55 Resource Type *mashup*

The <*mashup*> resource is a virtual resource because it does not have a representation. It is the child resource of a <*semanticMashupInstance*> resource. When a RETRIEVE operation is sent to the <*mashup*> resource, it triggers a calculation and generation of the mashup result based on its parent resource <*semanticMashupInstance*>.

9.6.56 Resource Type *semanticMashupResult*

<*semanticMashupResult*> resource stores the mashup result. It is the child resource of a <*semanticMashupInstance*> resource. A <*semanticMashupResult*> resource shall be automatically generated by a Hosting CSE when it executes a semantic mashup operation on a <*semanticMashupInstance*> resource.

The <*semanticMashupResult*> resource shall contain the child resources specified in table 9.6.56-1 and the attributes specified in table 9.6.56-2.

Table 9.6.56-1: Child resources of <*semanticMashupResult*> resource

Child Resources of <*semanticMashupResult*>	Child Resource Type	Multiplicity	Description	<*semanticMashupResultAnnc*> Child Resource Types
<*variable*>	<*semanticDescriptor*>	0..1	Describes general semantic information for this <*semanticMashupResult*> resource.	<*semanticDescriptor*>, <*semanticDescriptorAnnc*>
<*variable*>	<*subscription*>	0..n	Stands for any subscription on this <*semanticMashupResult*> resource.	<*subscription*>

Table 9.6.56-2: Attribute of <*semanticMashupResult*> resource

Attributes of <*semanticMashupResult*>	Multiplicity	RW/ RO/ WO	Description	<*semanticMashupInstanceAnnc*> Attributes
resourceType	1	RO	See clause 9.6.1.3.	NA
resourceID	1	RO	See clause 9.6.1.3.	NA
resourceName	1	WO	See clause 9.6.1.3.	NA
parentID	1	RO	See clause 9.6.1.3.	NA
expirationTime	1	RW	See clause 9.6.1.3.	MA
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.	MA
labels	0..1 (L)	RW	See clause 9.6.1.3.	MA
creationTime	1	RO	See clause 9.6.1.3.	NA
lastModifiedTime	1	RO	See clause 9.6.1.3.	NA
announceTo	0..1 (L)	RW	See clause 9.6.1.3.	NA
announcedAttribute	0..1 (L)	RW	See clause 9.6.1.3.	NA
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.	OA
creator	0..1	RO	See clause 9.6.1.3.	NA
smjpInputParameter	0..1	RO	Contains the value of all input parameters which are required to calculate the mashup result. Note that the types of these input parameters are specified by the *inputDescriptor* attribute of the corresponding <*semanticMashupJobProfile*> which is denoted by the *smjpID* attribute of the parent resource <*semanticMashupInstance*>. This attribute is not needed if the corresponding <*semanticMashupJobProfile*> does not have *inputDescriptor* attribute. The value of this attribute shall be automatically copied from the *smjpInputParameter* attribute of the parent resource <*semanticMashupInstance*>. This attribute shall not be updated by other entities except the Hosting CSE.	OA

Attributes of <*semanticMashupResult*>	Multiplicity	RW/ RO/ WO	Description	<*semanticMashupInstanceAnnc*> *Attributes*
mashupResultFormat	1	RO	Stands for the format of mashupResult representation (e.g. Integer, Float, Text, XML, JSON, etc.). The value of this attribute shall be obtained by a Hosting CSE directly from *outputDescriptor* attribute of corresponding <*semanticMashupJobProfile*> resource. This attribute shall not be updated by other entities except the Hosting CSE.	OA
mashupResult	1	RO	Contains the representation of mashup result. The value of this attribute shall be only generated by the Hosting CSE when it executes a semantic mashup operation. This attribute shall not be updated by other entities except the Hosting CSE.	OA

9.6.57 Resource Type *multimediaSession*

A <*multimediaSession*> resource shall represent information about a multimedia session involving two AEs. This resource is created by the session originator as the child of the <*AE*> resource which represents a session target. The creation, update or deletion of the <*multimediaSession*> resource triggers the AEs to manage (e.g. establish, tear-down) the multimedia session. The multimedia session described in the <*multimediaSession*> resource is managed by the two AEs using non-oneM2M protocols.

NOTE: Additional features (e.g. manage QoS settings for a session in an underlying network) involving Mcn reference point will be considered in future releases.

The <*multimediaSession*> resource shall contain the child resources specified in table 9.6.57-1.

Table 9.6.57-1: Child resources of <*multimediaSession*> resource

Child Resources of <*multimediaSession*>	Child Resource Type	Multiplicity	Description	<*multimediaSessionAnnc*> Child Resource Types
[variable]	<*subscription*>	0..n	See clause 9.6.8	<*subscription*>
[variable]	<*transaction*>	0..n	See clause 9.6.48	<*transaction*>

The <*multimediaSession*> resource shall contain the attributes specified in table 9.6.57-2.

Table 9.6.57-2: Attributes of <*multimediaSession*> resource

Attributes of <*multimediaSession*>	Multiplicity	RW/ RO/ WO	Description	<*multimediaSessionAnnc*> Attributes
resourceType	1	RO	See clause 9.6.1.3.	NA
resourceID	1	RO	See clause 9.6.1.3.	MA
resourceName	1	WO	See clause 9.6.1.3.	MA
parentID	1	RO	See clause 9.6.1.3.	NA
expirationTime	1	RW	See clause 9.6.1.3.	MA
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.	MA
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.	OA
creationTime	1	RW	See clause 9.6.1.3.	NA
lastModifiedTime	1	RO	See clause 9.6.1.3.	NA
labels	0..1 (L)	RO	See clause 9.6.1.3 where this common attribute is described.	MA
announceTo	0..1 (L)	RW	See clause 9.6.1.3.	NA
announcedAttribute	0..1 (L)	RW	See clause 9.6.1.3.	NA
sessionOriginatorID	1	WO	The AE-ID of the multimedia session originator.	OA
acceptedSessionDescriptions	1(L)	RW	This is the final accepted and agreed upon session description(s) based on the received response from the target of the multimedia session. When this attribute is set by the session target, the session originator establishes a session with a non-oneM2M protocol. The session description is compliant to the Session Description Protocol [17].	OA
offeredSessionDescriptions	1(L)	RW	A list of session descriptions offered by the Originator of the session to the target. The session descriptors are compliant to the Session Description Protocol [17].	NA
sessionState	1	RW	The current state of the multimedia session. The supported values are ONLINE and OFFLINE. This attribute is set either by the session originator or the target. When this attribute is OFFLINE, the Hosting CSE shall allow updates to *offeredSessionDescriptions* and/or *acceptedSessionDescription*. Otherwise, the Hosting CSE shall reject the updates to these attributes. When the session is in the OFFLINE state, the corresponding AE session endpoints shall not initiate the flow of media between one another. When in the ONLINE state, the AEs are free to initiate the flow of media.	OA

9.6.58 Resource Type *crossResourceSubscription*

The <*crossResourceSubscription*> resource represents a cross-resource subscription over a set of target resources which could be existing <*subscription*> and/or other subscribable oneM2M resources. The Hosting CSE shall generate a cross-resource notifications only when expected changes occur on a designated number of target resources concurrently within a time window. The <*crossResourceSubscription*> resource shall specify the involved target resources in order to generate cross-resource notification.

The <*crossResourceSubscription*> resource shall contain the child resources specified in table 9.6.58-1.

Table 9.6.58-1: Child resources of *<crossResourceSubscription>* resource

Child Resources of *<crossResourceSubscription>*	Child Resource Type	Multiplicity	Description
notificationSchedule	*<schedule>*	0..1	See clause 9.6.9.
[variable]	*<notificationTargetMgmtPolicyRef>*	0..n	See clause 9.6.31.
nstr	*<notificationTargetSelfReference>*	1	See clause 9.6.34.
[variable]	*<transaction>*	0..n	See clause 9.6.48.

The <*crossResourceSubscription*> resource shall contain the attributes specified in table 9.6.58-2.

Table 9.6.58-2: Attributes of *<crossResourceSubscription>* resource

Attributes of *<crossResourceSubscription>*	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
labels	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
announceTo	0..1 (L)	RW	See clause 9.6.1.3.
announcedAttribute	0..1 (L)	RW	See clause 9.6.1.3.
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.
creator	1	RO	See clause 9.6.1.3.
expirationCounter	0..1	RW	See clause 9.6.8.
notificationURI	1 (L)	RW	See clause 9.6.8.
notificationEventCat	0..1	RW	See clause 9.6.8.
subscriberURI	0..1	WO	See clause 9.6.8.
regularResourcesAsTarget	0..1	RW	This attribute indicates a list of regular resources (i.e. normal resources rather than <*subscription*> resources), which shall be used as the target resource for this cross-resource subscription. Here, the regular resource is referred to as any subscribable oneM2M resources.
subscriptionResourcesAsTarget	0..1	RW	This attribute indicates a list of existing <*subscription*> resources, which shall be used as the target resource for this cross-resource subscription.
timeWindowType	1	RW	This attribute indicates the type of time window mechanisms (e.g. *timeWindowType*=1 stands for periodic time window without any overlapping and *timeWindowType*=2 represents sliding time window where current time window will be slid to become next time window when a cross-resource notification is generated for instance) which will be used to determine the generation of a cross-resource notification.
timeWindowSize	1	RW	This attribute indicates the size or time duration (e.g. in seconds) of the time window, based on which cross-resource notifications shall be generated. Note that the maximum window size (e.g. 60 seconds) may be enforced by the Hosting CSE for a subscriber; if the *timeWindowSize* indicated or requested by a subscriber is larger than the maximum window size, the Hosting CSE may reject the subscriber's request for cross-resource subscription.

Attributes of <*crossResourceSubscription*>	Multiplicity	RW/ RO/ WO	Description
eventNotificationCriteriaSet	0..1(L)	RW	This attribute lists *eventNotificationCriteria* for each regular target resource as indicated in *regularResourcesAsTarget* attribute and involved in a cross-resource subscription. If there is only one *eventNotificationCriteria* contained in this attribute, it shall be applied to all target resources as indicated by *regularResourcesAsTarget* attribute. If only *subscriptionResourcesAsTarget* attribute appears (i.e. no *regularResourcesAsTarget* attribute), *eventNotificationCriteriaSet* shall not be needed. See clause 9.6.8 for the description of *eventNotificationCriteria*.

9.6.59 Void

9.6.60 Resource Type *backgroundDataTransfer*

The <*backgroundDataTransfer*> resource is used to request that the IN-CSE negotiates a background data transfer for a set of field nodes, with the Underlying Network. The resource attributes provide the characteristics of the background data transfer, optional guidance for transfer policy selection and the field nodes involved with the data transfer.

The <*backgroundDataTransfer*> resource contains the child resources specified in table 9.6.60-1.

Table 9.6.60-1: Child resources of <*backgroundDataTransfer*> resource

Child Resources of <*backgroundDataTransfer*>	Child Resource Type	Multiplicity	Description	<*backgroundDataTransfer*> Child Resource Types
[variable]	<*subscription*>	0..n	See clause 9.6.8.	<*subscription*>

The <*backgroundDataTransfer*> resource contains the attributes specified in table 9.6.60-2.

Table 9.6.60-2: Attributes of <*backgroundDataTransfer*> resource

Attributes of <*backgroundDataTransfer*>	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
dynamicAuthorizationConsultationIDs	0..1 (L)	RW	See clause 9.6.1.3.
labels	0..1 (L)	RW	See clause 9.6.1.3.
creator	0..1	RO	See clause 9.6.1.3.
volumePerNode	1	WO	Expected data volume for the background data transfer.
numberOfNodes	1	WO	Desired number of nodes for the background data transfer.
desiredTimeWindow	0..1	WO	Desired time window for the background data transfer.
transferSelectionGuidance	0..1(L)	WO	List that includes guidance to IN-CSE in selecting from multiple transfer policies provided by underlying network. Possible values include:"lowest cost", "highest throughput given maximum cost of X", etc. If not included, the IN-CSE may independently choose from among multiple transfer policies.
geographicInformation	0..1	WO	Provides geographic information for the policy request.
groupLink	0..1	RW	This attribute shall be used if the background data transfer is requested for sending a request to a group of field domain nodes. It is assumed that a <*group*> resource, with a *memberIDs* list including all field domain nodes that need to be reached, has already been created. This attribute contains the *resource identifier* of the <*group*> resource of field domain nodes for which the background data transfer applies. The *backgroundDataTransfer* resource may have either a *groupLink* attribute or a list of *memberIDs.* If the *memberIDs* attribute contains a valid list of member resource IDs, the *groupLink* attribute shall be ignored.
memberIDs	0..1 (L)	RW	List of member resource IDs for which the transfer policy applies. The valid resource types are <remoteCSE> and <AE>. The *backgroundDataTransfer* resource may have either a *groupLink* attribute or a list of *memberIDs.* If the *memberIDs* attribute contains a valid list of member resource IDs, the *groupLink* attribute shall be ignored.

10 Information Flows

10.1 Basic Procedures

10.1.1 Overview

As a pre-condition to the execution of the following procedures, M2M operational security procedures as specified in clauses 11.3.1 through 11.3.4 shall have been performed. In case of failure, the error shall be reported as specified in ETSI TS 118 104 [3].

The procedures in the following clauses assume blocking requests as described in clause 8.2.2.

10.1.2 CREATE (C)

Originator requests to create a resource by using the CREATE method. See clause 8.1.2 for the parameters to be included in the Request message.

Hosting CSE If the request is allowed by the given privileges, the Receiver shall create the resource.

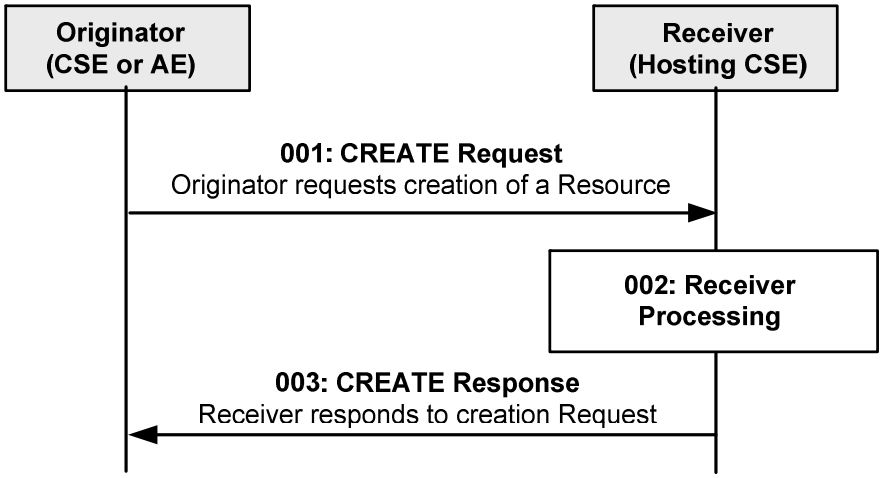


Figure 10.1.2-1: Procedure for CREATing a Resource

Step 001: The Originator shall send mandatory parameters and may send optional parameters in Request message for CREATE operation as specified in clause 8.1.2.

Step 002: The Receiver shall:

1) Check if the Originator has the appropriate privileges for performing the request. Privileges of the targeted resource are linked by the *accessControlPolicyIDs* attribute. Different handlings for a target resource which does not have the *accessControlPolicyIDs* attribute by the resource type definition(e.g. *schedule* resource type) or a target resource which has the definition but the attribute has no value and so on are defined in table 9.6.1.3.2-1 (common attributes description).

2) Verify that the name for the created resource as suggested by the *resourceName* attribute in ***Content*** parameter, if provided by the Originator in the CREATE Request message, does not already exist among child resources of the target resource. If no child within the targeted resource exists with the same *resourceName* as suggested by the Originator, use that name for the resource to be created. If a child uses the resourceName already, the Receiver shall reject the request and return an error to the Originator. If the name was not suggested by the Originator, assign a name generated by the Receiver to the resource to be created.

NOTE: The name of a resource in general is not the same as its Resource ID. While a name of a resource only needs to be unique among the children of the same parent resource, the Resource ID needs to be unique in context of the Hosting CSE. When the name of the resource to be created is assigned by the Receiver, it may choose to use a name that is identical to the Unstructured-CSE-relative-Resource ID.

3) Assign a Resource-ID (see *resourceID* attribute in common attribute table 9.6.1.3.2-1) to the resource to be created.

4) Assign values for mandatory RO mode attributes of the resource and override values provided for other mandatory attributes, where needed, and where allowed by the resource type definition and if not provided by the Originator itself.

5) The Receiver shall assign a value to the following common attributes specified in clause 9.6.1.3:

   a) *parentID*;

   b) *creationTime*;

c) *expirationTime*: if not provided by the Originator, the Receiver shall assign the maximum value possible (within the restriction of the Receiver policies). If the value provided by the Originator cannot be supported, due to either policy or subscription restrictions, the Receiver will assign a new value;

d) *lastModifiedTime*: which is equals to the *creationTime*;

e) Any other RO (Read Only) attributes within the restriction of the Receiver policies.

6) The Receiver shall check whether a *creator* attribute is included in the ***Content*** parameter of the request. If included, the *creator* attribute shall not have a value in the ***Content*** parameter of the request. If the *creator* attribute is included in the request and the *creator* attribute is supported for the type of resource being created, then the Receiver shall to include the *creator* attribute in the resource to be created. The Receiver shall assign a value equal to the value carried in the ***From*** request parameter. In the event that the originator provides a value for the *creator* attribute within the request, this request shall be deemed invalid.

   On the other hand if the *creator* attribute is not included in the ***Content*** parameter of the request, then the Receiver shall not include the *creator* attribute in the resource to be created.

7) On successful validation of the Create Request, the Receiver shall create the requested resource.

8) The Receiver shall check if the created child resource leads to changes in its parent resource's attribute(s), if so the parent resource's attribute(s) shall be updated.

9) The Receiver shall check if the created child resource references an Application Entity Resource ID, if so the Hosting CSE shall send a NOTIFY request to the IN-CSE, requesting to add the entry to the <AEContactList> resource.

Step 003: The Receiver shall respond with mandatory parameters and may send optional parameters in Response message for CREATE operation as specified in clause 8.1.3.

General Exceptions:

1) The Originator does not have the privileges to create a resource on the Receiver. The Receiver responds with an error.

2) The resource with the specified name (if provided) already exists at the Receiver. The Receiver responds with an error.

3) The provided information in ***Content*** is not accepted by the Receiver (e.g. missing mandatory parameter). The Receiver responds with an error.

10.1.3 RETRIEVE (R)

The RETRIEVE operation shall be used for retrieving the information stored for any of the attributes for a resource at the Receiver CSE. The Originator CSE or AE may request to retrieve a specific attribute by including the name of such attribute in the ***Content*** parameter in the request message.

Originator requests retrieval of all attributes or a specific attributes of the target resource by using RETRIEVE Request. See clause 8.1.2 for the information to be included in the Request message. If only some specific attributes need to be retrieved, the name of such attributes shall be included in the ***Content*** parameter of the Request message.

Receiver performs local processing to verify the existence of requested resource and checks privileges for retrieving the information related to the resource. After successful verification, the Receiver shall return the requested information, otherwise an error indication shall be returned.

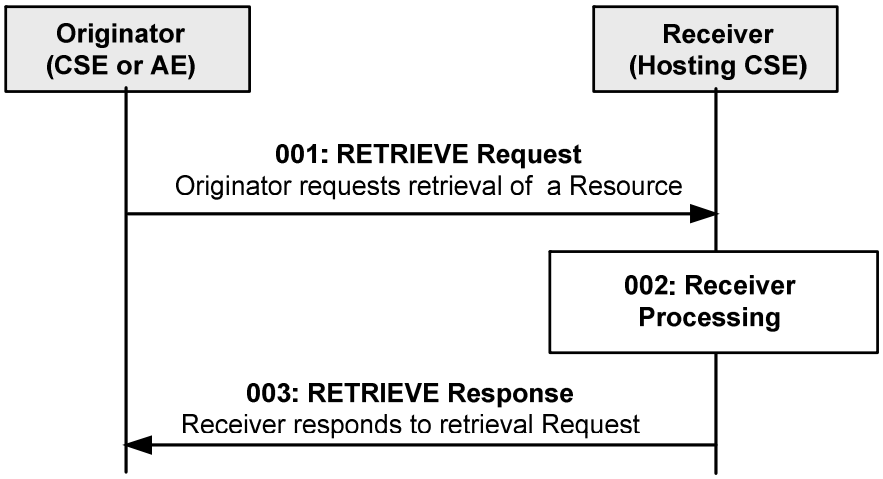


Figure 10.1.3-1: Procedure for RETRIEVing a Resource

Step 001: The Originator shall send mandatory parameters and may send optional parameters in Request message for RETRIEVE operation as specified in clause 8.1.2.

Step 002: The Receiver shall verify the existence (including ***Filter Criteria*** checking, if it is given) of the target resource or the attribute and check if the Originator has appropriate privileges to retrieve information stored in the resource/attribute. This privilege checking follows the rules defined in table 9.6.1.3.2-1 (common attributes description).

Step 003: The Receiver shall respond with mandatory parameters and may send optional parameters in Response message for RETRIEVE operation as specified in clause 8.1.3.

General Exceptions:

1) The targeted resource/attribute in ***To*** parameter does not exist. The Receiver responds with an error.

2) The Originator does not have privileges to retrieve information stored in the resource on the Receiver. The Receiver responds with an error.

10.1.4 UPDATE (U)

The UPDATE operation shall be used for updating the information stored for any of the attributes at a target resource. Especially important is the *expirationTime*, since a failure in refreshing this attribute may result in the deletion of the resource. The Originator CSE or AE can request to update, create or delete specific attribute(s) at the target resource by including the name of such attribute(s) and its values in the ***Content*** parameter of the request message.

Originator requests update any of the attributes at the target resource by using UPDATE Request message. The Originator shall send new (proposed) values for the attribute(s) that need to be updated. The UPDATE operation allows to modify or create previously non-existing attributes of the resource type (defined in clause 9.6) that are indicated as "RW" (Read Write) for the specific resource type definition.

The **Originator** requests to delete attributes at the target resource by using UPDATE Request message. The Originator shall send the name of the attributes to be deleted (defined in clause 9.6) for the specific resource type with their value set to NULL, in the Request message.

See clause 8.1.2 for the information to be included in the Request message.

The **Receiver** verifies the existence of the addressed resource, the validity of the attributes provided and the privileges to modify them, the Receiver shall update the attributes provided and shall return a Response message to the Originator with the operation results as specified in clause 8.1.3.

If the attributes provided do not exist, after verifying the existence of the addressed resource, the Receiver validates the attributes provided and the privileges to create them. On successful validation, the Receiver shall create the attributes provided with their associated values and shall return a Response message to the Originator with the operation results as specified in clause 8.1.3.

If the attributes provided have their value set to NULL, after verifying the existence of the addressed resource, the Receiver validates the attributes provided and the Update privileges of the Originator. On successful validation, the Receiver shall delete such attributes and shall return a Response message to the Originator with the operation results as specified in clause 8.1.3.

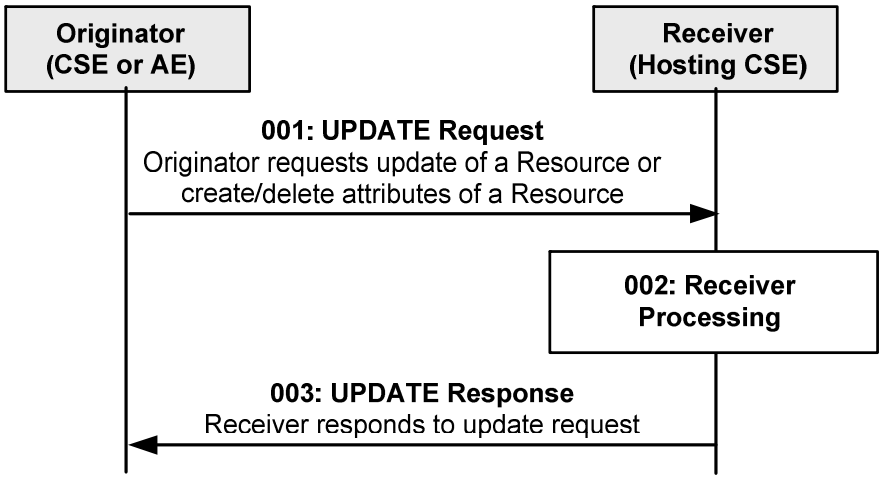


Figure 10.1.4-1: Procedure for UPDATing a Resource

Step 001: The Originator shall send mandatory parameters and may send optional parameters in Request message for UPDATE operation as specified in clause 8.1.2.

Step 002: The Receiver shall verify the existence (including ***Filter Criteria*** checking, if it is given) of the requested resource and if the Originator has the appropriate privilege to update the resource. This privilege checking follows the rules defined in table 9.6.1.3.2-1 (common attributes description). On successful validation, the Receiver shall update the resource as requested. If the attributes provided do not exist, the Receiver shall validate if the Originator has appropriate privileges to create the attributes at the target resource. On successful validation, the Receiver shall create the attributes with their associated values at the resource as requested. If the attributes provided have their value set to NULL, the Receiver shall validate if the Originator has Update privilege to delete the attributes at the target resource. On successful validation, the Receiver shall delete such attributes. The Receiver shall check if the updated target resource is a child of a parent resource having a stateTag attribute and increment the stateTag if present. The Receiver shall check if the update causes a change to a reference to an Application Entity Resource ID. If so the Hosting CSE shall send a NOTIFY request to the IN-CSE, requesting to update the entry to the <AEContactList> resource.

Step 003: The Receiver shall respond with mandatory parameters and may send optional parameters in Response message for UPDATE operation as specified in clause 8.1.3.

General Exceptions:

1) The targeted resource in ***To*** parameter does not exist. The Receiver responds with an error.

2) The Originator does not have the privilege to Update the resource including create non-existing attributes or delete existing attributes on the Receiver. The Receiver responds with an error.

3) The provided information in the ***Content*** is not accepted by the Receiver. The Receiver responds with an error.

10.1.5 DELETE (D)

The DELETE operation shall be used by an Originator CSE or AE to delete a resource at a Receiver CSE. For such operation, the DELETE procedure shall consist of the deletion of all related information of the target resource.

Originator requests deletion of a resource by using a DELETE Request message. See clause 8.1.2 for the information to be included in the Request message.

Receiver The Receiver verifies the existence of the requested resource, and the privileges for deleting the resource.

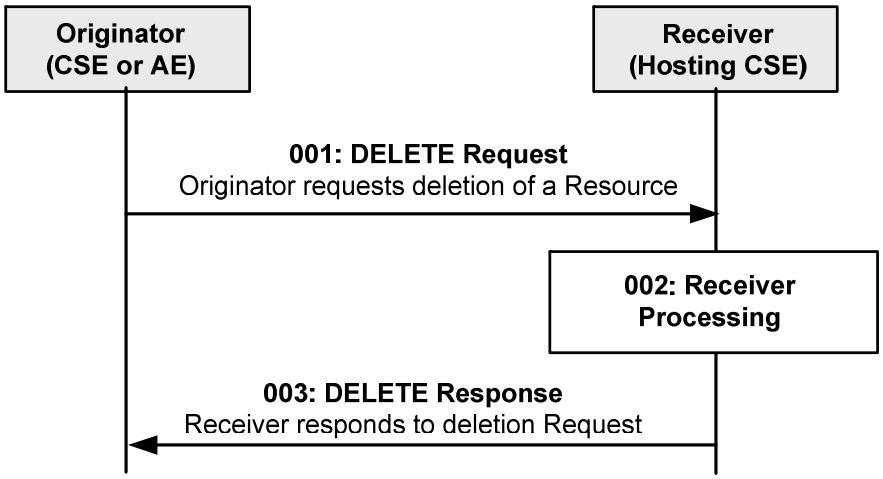


Figure 10.1.5-1: Procedure for DELETING a Resource

Step 001: The Originator shall send mandatory parameters and may send optional parameters in Request message for DELETE operation as specified in clause 8.1.2.

Step 002: The Receiver shall verify the existence (including ***Filter Criteria*** checking, if it is given) of the requested resource and if the Originator has the appropriate privilege to delete the resource. This privilege checking follows the rules defined in table 9.6.1.3.2-1 (common attributes description). On successful validation, the Receiver shall check for child resources and delete all child resources and the associated references in parent resources and it shall remove the resource itself. The Receiver shall check if the deleted child resource leads to changes in its parent resource's attribute(s), if so the parent resource's attribute(s) shall be updated. If the deleted resource had a reference to an Application Entity Resource ID, the Hosting CSE shall send a NOTIFY request to the IN-CSE, requesting to delete the entry from the <AEContactList> resource.

Step 003: The Receiver shall respond with mandatory parameters and may send optional parameters in Response message for DELETE operation as specified in clause 8.1.3.

General Exceptions:

1) The targeted resource in ***To*** information does not exist. The Receiver responds with an error.

2) The Originator does not have the privileges to delete the resource on the Receiver. The Receiver responds with an error.

10.1.6 NOTIFY (N)

The NOTIFY operation shall be used for notifying information. All the specific notification procedures defined in this present document are listed in clause 10.1.6 (Notification procedures).

The **Originator** requests to notify an entity by using NOTIFY method. See clause 8.1.2 for the information to be included in a Request message.

The **Receiver** responds to the Originator with the operation results as specified in clause 8.1.3.

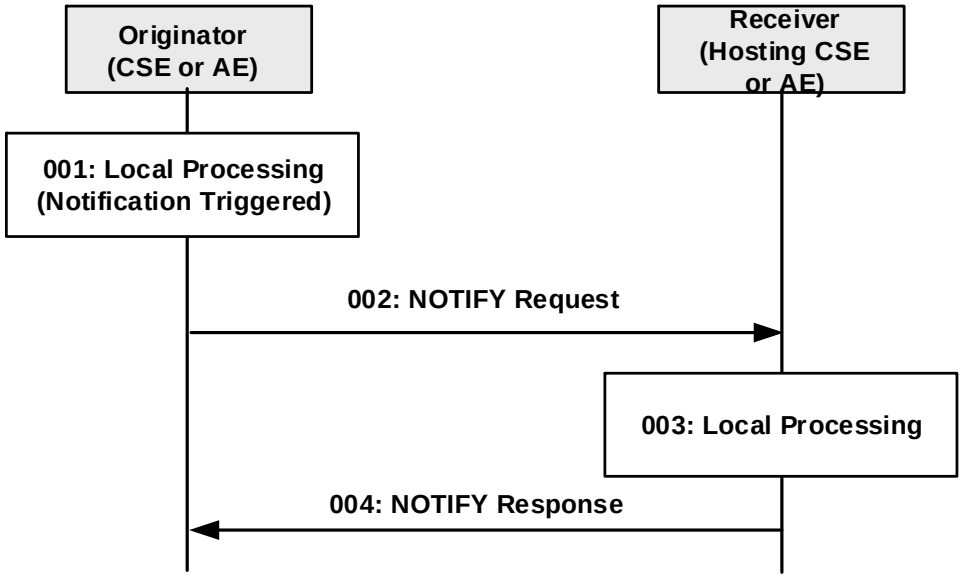


Figure 10.1.6-1: Procedure for NOTIFYing Information

Step 001: A notification to be sent to the Receiver is triggered in the Originator.

Step 002: The Originator shall send mandatory parameters and may send optional parameters in Request message for NOTIFY operation as specified in clause 8.1.2.

Step 003: Local Processing.

Step 004: The Receiver shall respond with mandatory parameters and may send optional parameters in Response message for NOTIFY operation as specified in clause 8.1.3.

General Exceptions:

- See ETSI TS 118 103 [2].

In the present document, notification procedures are defined in the following procedures:

- <subscription> resource handling (clause 10.2.10)
  - to notify Receiver(s) of modifications of a resource for an associated <subscription> resource
  - to notify aggregated notifications from <subscription> member resources of <group> resource
  - to request Receiver(s) to perform resource subscription verification
  - to notify deletion of the <subscription> resource
  - to seek authorization from the subscription creator during a notification target deletion
- Asynchronous non-blocking request handling (clause 8.2.2.3)
  - to send the result of the request
- <pollingChannelURI> resource handling (clause 10.2.5.19)
  - to send the response corresponding to a request delivered via service layer long polling
- IPE on-demand discovery handling (clause 10.2.6)
  - to notify Receiver(s) (i.e. IPE) for on-demand discovery request
- End-to-end security handling (clause 11.4)
  - to send the request/response that cannot be readable by Transit CSEs

- Dynamic authorization consultation handling (clause 11.5)
  - to seek authorization to access a resource from Dynamic Authorization Server
- Change in AE Registration Point (clause 10.2.16)
  - to notify Receivers that an AE has changed registration point
- Change in a resource with reference to an Application Entity Resource ID (clauses 10.1.2 (CREATE), 10.1.4 (UPDATE), 10.1.5 (DELETE) and 9.6.1.3.1 (expiration timer expiry)
  - to notify IN-CSE that the Originator has a new/updated reference to an Application Entity Resource identifier
- <*crossResourceSubscription*> resource handling (clause 10.2.10)
  - to notify Receiver(s) of cross-resource notification generated by a <*crossResourceSubscription*> Hosting CSE
  - to notify deletion of the <*crossResourceSubscription*> resource
  - to request Receiver(s) to perform cross-resource subscription verification
  - to seek authorization from the cross-resource subscription creator during a notification target deletion

10.2 Functional procedures

10.2.1 Overview

The basic procedure for the corresponding operations as specified in clause 10.1 shall be performed with the modifications specific to the resource type procedures as described in clause 10.2.

For resources without defined resource type-specific operations, the basic operations in clause 10.1 shall apply.

10.2.2 Registration

10.2.2.1 AE registration

AE Registration towards a Registrar CSE is a process to enable an AE to use services offered by the oneM2M System. It is achieved through the create <*AE*> procedure. The registration details can be viewed and updated using retrieve <*AE*> and update <*AE*> procedures, respectively. The delete <*AE*> procedure is used to de-register an AE from its Registrar CSE.

AE Registration helps the M2M Service Provider to offer its services only to authorized AEs and to protect its platform from malicious AE.

10.2.2.2 Create <*AE*>

This procedure shall be used for creating an <*AE*> resource. This operation is part of the registration procedure for AEs on the Registrar CSE (which is also the Hosting CSE), as described in clause 10.2.2.2.

Table 10.2.2.2-1: <*AE*> CREATE

<*AE*> CREATE	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***From*:** Registree AE only ***Content:*** The resource content shall provide the information as defined in clause 9.6.5
Processing at Originator before sending Request	According to clause 10.2.2.2
Processing at Receiver	According to clause 10.2.2.2
Information in Response message	All parameters defined in table 8.1.3-1
Processing at Originator after receiving Response	According to clause 10.2.2.2
Exceptions	According to clause 10.2.2.2

The procedure for AE registration follows the message flow description depicted in figure 10.2.2.2-1. It defines in which cases additional procedures need to be initiated by the Registrar CSE for creating or updating of <*AEAnnc*> resources hosted on the M2M SP's IN-CSE in case an AE-ID-Stem starting with an 'S' character shall be used, see table 7.2-1 for the definition of AE-ID-Stem. The above additional procedures i.e. steps related to announcement shall not be required when Registrar CSE is the IN-CSE.

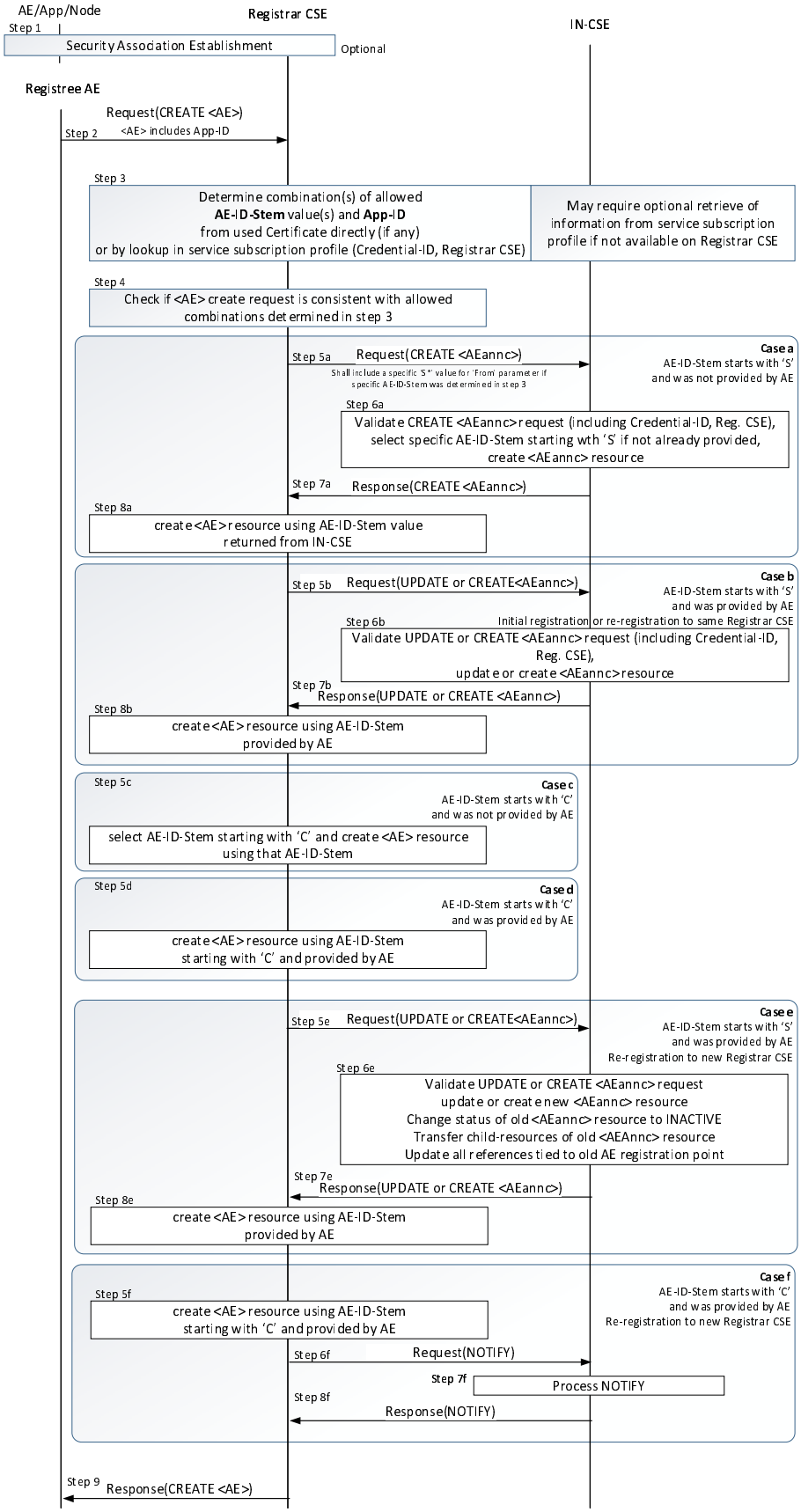


Figure 10.2.2.2-1: Procedure for Creating an <AE> Resource

Originator: The Originator shall be the Registree AE.

Receiver: The Receiver shall allow the creation of the <*AE*> resource according to the m2m service subscription validation. Note that access control for resource access (e.g. using <accessControlPolicy> resources) is omitted. To validate the m2m service subscription profile, the Receiver shall check the corresponding <serviceSubscribedNode> resource, by matching the CSE-ID in the m2m service subscription profile against the Receiver owned CSE-ID. Subsequently the Receiver shall check whether the Registree AE is included in the linked (i.e. ruleLinks attribute) <serviceSubscribedAppRules> resource(s).

- **Step 001:** Optional: In case the Registree AE intends to use a Security Association to perform the registration, a Security Association Establishment procedure (see clause 11.2.2) shall get carried out first. In some cases (e.g. registration of AE internal to an MN or ASN), this may not be required depending on deployment choices of the M2M SP. Therefore, this step is optional. This optional Security Association can be established between the following entities:

  - The Registree AE and the Registrar CSE - in which case the specific AE that is subsequently sending the request to get registered shall be authenticated.

  - The Node on which the Registree AE is hosted and the Registrar CSE - in which case only the Node from which the registration request is received at the Registrar CSE shall be authenticated. In this case one or more AEs hosted on the authenticated node may communicate over either a single Security Association or over individual Security Associations.

NOTE: The Node authentication should be used only when the M2M Service Provider trusts the AE (on the Node) to provide the correct AE-ID and App-ID. The present document does not provide mechanisms by which the M2M Service Provider can obtain assurance about the trustworthiness of the AE when using Node authentication. For example, such a mechanism (by which the M2M Service Provider can obtain assurance about the trustworthiness of the AE) could be provided by executing the M2M Application on a secure environment.

The identifier of the security credentials used for establishing the Security Association in this step shall be termed 'Credential-ID' for the remainder of this procedure description. If no Security Association has been performed the Credential-ID is not applicable.

- **Step 002:** The Originator shall send the information defined in clause 10.1.2 for the registration CREATE procedure with the following specific information in the CREATE Request message:

  ***From*:** AE-ID-Stem or Not Present:

  i) In case the Registree AE has already registered successfully before, then deregistered and intends to register again to the same Registrar CSE with the same AE-ID-Stem value as before, the Registree AE shall include that AE-ID-Stem value into the ***From*** parameter.

  ii) In case the Registree AE intends to initiate a fresh registration with a pre-provisioned AE-ID-Stem value, the Registree AE shall include that pre-provisioned AE-ID-Stem value into the ***From*** parameter.

  iii) In case the Registree AE has not registered successfully before and intends to get an M2M-SP-assigned AE-ID-Stem starting with an 'S' character assigned to itself but it does not have any specific value to suggest, it shall set the ***From*** parameter to the character 'S'.

  iv) In case the Registree AE has not registered successfully before and intends to get a Registrar CSE-assigned AE-ID-Stem starting with an 'C' character assigned to itself but it does not have any specific value to suggest, it shall set the ***From*** parameter to the character 'C'.

  v) In case the Registree AE intends to initiate a fresh registration and has no preference for the AE-ID-Stem value, the ***From*** parameter shall not be sent.

  vi) In case the Registree AE has already registered successfully to a Registrar CSE, and now intends to register to a different Registrar CSE (i.e. Registree AE has changed its registration point), the Registree AE shall include its AE-ID-Stem value (from the prior registration) into the ***From*** parameter.

The CSE shall allow unknown AEs to attempt the 'CREATE' before they are granted this permission. See ETSI TS 118 103 [2] for further details about authentication for the AE:

- **Step 003:** The Receiver shall determine whether the request to register the Registree AE meets any of the following conditions:

  - In case the Security Association Establishment in step 001 was performed using security credentials in form of a Certificate that included an App-ID and an AE-ID-Stem attribute, check if they match with the App-ID attribute in the ***Content*** parameter of the request and the AE-ID-Stem in the ***From*** parameter of the request.

  - Check if the applicable service subscription profile lists a combination of (allowed AE-ID-Stem value and allowed App-ID value) for the Credential-ID and the Registrar CSE-ID (see clause 11.2.2) that match with the App-ID attribute in the ***Content*** parameter of the request and the AE-ID-Stem in the ***From*** parameter of the request. If the information needed to perform that checking is not available to the Registrar CSE locally, the Registrar CSE shall retrieve that information from the applicable service subscription profile(s) from the IN-CSE. If the *From* parameter was not set in the request and the allowed AE-ID-Stem includes a wild card ("*") in the applicable service subscription profile(s), the Registrar CSE shall assign the starting character ('S', 'C') in accordance with provisioned Service Provider policy. The applicable rules for this checking are contained in the <*serviceSubscribedAppRule*> resource(s) which are linked to by the *ruleLinks* attribute of the <*m2mServiceSubscribedNode*> resource(s) associated with the Registrar CSE. The <*m2mServiceSubscribedNode*> resource(s) associated with the Registrar CSE can be retrieved from the IN-CSE by applying the ***Filter Criteria*** parameter set to "CSE-ID={Registrar-CSE-ID}"where {Registrar-CSE-ID} needs to be substituted by the actual CSE-ID of the Registrar-CSE.

If none of the conditions are met, the registration is not allowed and the Receiver shall respond with an error:

- **Step 004:** If the ***From*** parameter of the request provides a complete AE-ID-Stem value, i.e. case i), ii), or vi) of step 002 applied, the Registrar CSE shall check whether an <*AE*> resource with an Unstructured-CSE-relative-Resource-ID identical to the AE-ID-Stem value provided in the ***From*** parameter of the request does already exist on the Registrar CSE. If so, there is still an active registration using the same AE-ID-Stem on the Registrar CSE and the Registrar CSE shall respond with an error. If not, the Registrar CSE shall perform action (3) in step 002 of clause 10.1.2.

If the ***From*** parameter of the request provides a complete AE-ID-Stem and starts with 'S', i.e. case i), ii), or vi) of step 002 applied and 'S' is the first character of the provided AE-ID-Stem, and if the Registrar CSE determines that this is an initial registration or a re-registration to the same Registrar CSE, the procedure continues with case b) of the present step 004.

If the ***From*** parameter of the request provides a complete AE-ID-Stem and starts with 'S', i.e. case i), ii) or vi) of step 002 applied and 'S' is the first character of the provided AE-ID-Stem, and if the Registrar CSE determines that this is a re-registration due to a change in registration point, and if the Registree AE requests not to be tracked as it changes its registration point (*trackRegistrationPoints*= FALSE), the procedure continues as an initial registration, with case b) of the present step 004.

If the ***From*** parameter of the request provides a complete AE-ID-Stem and starts with 'S', i.e. case i), ii) or vi) of step 002 applied and 'S' is the first character of the provided AE-ID-Stem, and if the Registrar CSE determines that this is a re-registration due to a change in registration point, and if the Registree AE requests to be tracked as it changes its registration point (*trackRegistrationPoints* = TRUE), the procedure continues as a re-registration to a new Registrar CSE, with case e) of the present step 004.

If ***From*** parameter of the request provides a complete AE-ID-Stem and starts with 'C', i.e. case i), ii), or vi) of step 002 applied and 'C' is the first character of the provided AE-ID-Stem, and if the Registrar CSE determines that this is an initial registration or a re-registration to the same Registrar CSE, the procedure continues with case d) of the present step 004.

If ***From*** parameter of the request provides a complete AE-ID-Stem and starts with 'C', i.e. case i), ii) or vi) of step 002 applied and 'C' is the first character of the provided AE-ID-Stem, and if the Registrar CSE determines that this is a re-registration due to a change in registration point, and if the Registree AE requests not to be tracked as it changes its registration point (*trackRegistrationPoints* = FALSE), the procedure continues as an initial registration, with case d) of the present step 004.

If ***From*** parameter of the request provides a complete AE-ID-Stem and starts with 'C', i.e. case i), ii) or vi) of step 002 applied and 'C' is the first character of the provided AE-ID-Stem, and if the Registrar CSE determines that this is a re-registration due to a change in registration point, and if the Registree AE requests to be tracked as it changes its registration point (*trackRegistrationPoints* = TRUE), the procedure continues as a re-registration to a new Registrar CSE, with case f) of the present step 004.

If the ***From*** parameter of the request is equal to the value 'S', i.e. case iii) of step 002 applied, the procedure continues with case a) of the present step 004.

If the ***From*** parameter of the request is equal to the value 'C', i.e. case iv) of step 002 applied, the procedure continues with case c) of the present step 004.

If the ***From*** parameter of the request is not sent, the Registrar CSE shall perform action (3) in step 002 of clause 10.1.2 to assign the resourceID with starting character ('S', 'C') in accordance with provisioned Service Provider policy and shall set the corresponding value in AE-ID-Stem. If the assigned value in AE-ID-Stem attribute starts with 'S', the procedure continues with case b) else the procedure continues with case d).

Case a) AE-ID-Stem starts with 'S' and AE does not include an AE-ID-Stem (initial registration):

Condition: In **step 003** it was determined that the AE-ID-Stem value to be used for the Registree AE starts with an 'S' character but no specific AE-ID-Stem was provided with the CREATE request of the Registree AE. This case applies when the Registree AE is supposed to use an M2M-SP-assigned AE-ID and wants to perform the initial registration:

- **Step 005a:** The Receiver shall send a CREATE request for an <*AEAnnc*> resource to the IN-CSE in order to create an <*AEAnnc*> resource on the IN-CSE that is associated with the Registree AE. The following information shall be sent with that CREATE request:

  - In case no specific AE-ID-Stem value to be used for the Registree AE was determined during **step 003**, the value 'S' shall be used in what follows for the AE-ID-Stem. Otherwise use the value determined in **step 003**.

  - The ***From*** parameter of the CREATE request for the <*AEAnnc*> resource shall be set to the SP-relative-CSE-ID or Absolute-CSE-ID followed by '/S'.

  - The *link* attribute of the <AEAnnc> resource to be created shall be set to the SP-Relative-Resource-ID format of a - not yet existent - <*AE*> resource hosted on the Registrar CSE constructed with a Unstructured-CSE-relative-Resource-ID that is equal to the AE-ID-Stem value used for the Registree AE.

  - The App-ID attribute of the <*AEAnnc*> resource to be created shall be present and set to the App-ID attribute value of the Registree AE.

  - The concatenation of the string 'Credential-ID:' and the actual Credential-ID of the Security Association used by the Registree AE - if any - shall be placed into the labels attribute of the <*AE Annc*> resource. If no Security Association was used by the Registree AE, then Credential-ID is not applicable.

- **Step 006a:** Upon reception of the CREATE <*AEAnnc*> request, the IN-CSE shall validate the request and verify whether the provided values of the App-ID attribute and the AE-ID-Stem in the ***From*** parameter is allowed for the combination of Credential-ID included in the labels attribute and the CSE-ID of the Registrar CSE included in the link attribute, according to the applicable service subscription profile. If that verification is successful and no specific AE-ID-Stem is provided, i.e. if the ***From*** parameter contains only the character 'S', the IN-CSE shall select an AE-ID-Stem in line with the applicable service subscription profile.

- **Step 007a:** When the validation and verification in **step 006a** completed successfully, the IN-CSE shall create <*AEAnnc*> resource with an Unstructured-CSE-relative-Resource-ID equal to the value of the AE-ID-Stem, replace the AE-ID-Stem for the trailing 'S' character in the Unstructured-CSE-relative-Resource-ID present in the link attribute if the AE-ID-Stem was selected by the IN-CSE, and send a successful response to the Registrar CSE.

- **Step 008a:** Upon reception of a successful response from the IN-CSE, the Registrar CSE shall use the Unstructured-CSE-relative-Resource-ID that was used for the <*AEAnnc*> resource on the IN-CSE also as the assigned Unstructured-CSE-relative-Resource-ID for the <*AE*> resource to be created on the Registrar CSE and continue with action (4) of **step 002** of the non-registration related CREATE procedure in clause 10.1.2.

Case b) AE-ID-Stem starts with 'S' and AE includes an AE-ID-Stem (initial registration or re-registration to the same Registrar CSE):

Condition: In **step 003** it was determined that the AE-ID-Stem value to be used for the Registree AE starts with an 'S' character and a specific AE-ID-Stem was provided with the CREATE request of the Registree AE. This case applies when the Registree AE is supposed to use an M2M-SP-assigned AE-ID and wants to perform initial registration or re-registration using its already assigned AE-ID-Stem:

- **Step 005b:** The receiver shall determine if an <*AEAnnc*> resource already exists on the IN-CSE that is associated with the Registree AE. The Receiver shall send an UPDATE request for an <*AEAnnc*> resource to the IN-CSE in order to update the already existing <*AEAnnc*> resource on the IN-CSE that is associated with the Registree AE in case of re-registration or the Receiver shall send a CREATE request for an <*AEAnnc*> resource to the IN-CSE in order to create an <*AEAnnc*> resource on the IN-CSE that is associated with the Registree AE in case of initial registration. The following information shall be sent with that UPDATE or CREATE request:

  - The ***To*** parameter shall contain the SP-relative-Resource-ID format of the Resource ID for the <*AEAnnc*> resource which shall be constructed from the CSE-ID of the IN-CSE and the AE-ID-Stem that the Registree AE provided.

  - ***From*** parameter of the CREATE or UPDATE request for the <*AEAnnc*> resource shall be set to the SP-relative-CSE-ID or Absolute-CSE-ID followed by '/' and the AE-ID-Stem value.

  - The link attribute of the <*AEAnnc*> resource shall be set (in case of initial registration) or updated (in case of re-registration) to the SP-Relative-Resource-ID format of a - not yet existent - <*AE*> resource hosted on the Registrar CSE constructed with an Unstructured-CSE-relative-Resource-ID that is equal to the AE-ID-Stem value used for the Registree AE.

  - The labels attribute of the <*AEAnnc*> resource shall be set (in case of initial registration) or updated (in case of re-registration) to the concatenation of the string 'Credential-ID:' and the Credential-ID of the Security Association used by the Registree AE, replacing the existing entry starting with 'Credential-ID:' if present. If no Security Association was used by the Registree AE, then Credential-ID is not applicable.

- **Step 006b:** Upon reception of the CREATE or UPDATE <*AEAnnc*> request, the IN-CSE shall validate the request and verify whether the values suggested to be set or to be updated for the Credential-ID included in the labels attribute - if any - and the CSE-ID of the Registrar CSE included in the ***From*** parameter still match with any of the allowed combinations of *App-ID* attribute and the AE-ID-Stem in the *link* attribute according to the applicable service subscription profile.

- **Step 007b:** When the validation and verification in **step 006b** completed successfully, the IN-CSE shall create <*AEAnnc*> resource with an Unstructured-CSE-relative-Resource-ID equal to the value of the provided AE-ID-Stem or update the <*AEAnnc*> resource in line with the parameters provided in step 005b.

- **Step 008b:** Upon reception of a successful response from the IN-CSE, the Registrar CSE shall use the Unstructured-CSE-relative-Resource-ID equal to the AE-ID-Stem provided by the Registree AE for the <*AE*> resource to be created on the Registrar CSE and continue with action (4) of **step 002** of the non-registration related CREATE procedure in clause 10.1.2.

Case c) AE-ID-Stem starts with 'C' and AE does not include an AE-ID-Stem (initial registration):

Condition: In **step 003** it was determined that the AE-ID-Stem value to be used for the Registree AE starts with an 'C' character but no specific AE-ID-Stem was provided with the CREATE request of the Registree AE. This case applies when the Registree AE is not supposed to use an M2M-SP-assigned AE-ID and wants to perform the initial registration:

- **Step 005c:** The Registrar CSE shall select an AE-ID-Stem starting with a 'C' character and use it for the Unstructured-CSE-relative-Resource-ID for the **<AE>** resource to be created on the Registrar CSE and continue with action (4) of **step 002** of the non-registration related CREATE procedure in clause 10.1.2.

Case d) AE-ID-Stem starts with 'C' and AE includes an AE-ID-Stem (initial registration or re-registration registration to the same Registrar CSE):

Condition: In **step 003** it was determined that the AE-ID-Stem value to be used for the Registree AE starts with an 'C' character and a specific AE-ID-Stem was provided with the CREATE request of the Registree AE. This case applies when the Registree AE is not supposed to use an M2M-SP-assigned AE-ID and wants to perform initial registration or re-registration using its already assigned AE-ID-Stem:

- **Step 005d:** The Registrar CSE shall use the Unstructured-CSE-relative-Resource-ID equal to the AE-ID-Stem in the ***From*** parameter for the <AE> resource to be created on the Registrar CSE and continue with action (4) of **step 002** of the non-registration related CREATE procedure in clause 10.1.2.

Case e) AE-ID-Stem starts with 'S' and AE includes an AE-ID-Stem (re-registration to a new Registrar CSE)

Condition: In **step 003** it was determined that the AE-ID-Stem value to be used for the Registree AE starts with an 'S' character and a specific AE-ID-Stem was provided with the CREATE request of the Registree AE. This case applies when the Registree AE is supposed to use an M2M-SP-assigned AE-ID and wants to perform a re-registration to a new Registrar CSE, using its already assigned AE-ID-Stem from a registration to a prior Registrar CSE:

- **Step 005e:** The receiver shall determine if an *<AEAnnc>* resource already exists on the IN-CSE that is associated with the Registree AE. If so, the Receiver shall send an UPDATE request for an *<AEAnnc>* resource to the IN-CSE in order to update the already existing *<AEAnnc>* resource on the IN-CSE that is associated with the Registree AE in case of re-registration. Otherwise, if there is no already existing *<AEAnnc>* resource associated with the Registree AE, the Receiver shall send a CREATE request for an *<AEAnnc>* resource to the IN-CSE in order to create an *<AEAnnc>* resource on the IN-CSE that is associated with the Registree AE. The following information shall be sent with that UPDATE or CREATE request:

  - The ***To*** parameter shall contain the SP-relative-Resource-ID format of the Resource ID for the *<AEAnnc>* resource which shall be constructed from the CSE-ID of the IN-CSE and the AE-ID-Stem that the Registree AE provided.

  - ***From*** parameter of the CREATE or UPDATE request for the *<AEAnnc>* resource shall be set to the SP-relative-CSE-ID or Absolute-CSE-ID followed by '/' and the AE-ID-Stem value.

  - The link attribute of the *<AEAnnc>* resource shall be set or updated, to the SP-Relative-Resource-ID format of a - not yet existent - *<AE>* resource hosted on the Registrar CSE constructed with an Unstructured-CSE-relative-Resource-ID that is equal to the AE-ID-Stem value used for the Registree AE.

  - The labels attribute of the *<AEAnnc>* resource shall be set or updated to the concatenation of the string 'Credential-ID:' and the Credential-ID of the Security Association used by the Registree AE, replacing the existing entry starting with 'Credential-ID:' if present. If no Security Association was used by the Registree AE, then Credential-ID is not applicable.

- **Step 006e:** Upon reception of the CREATE or UPDATE *<AEAnnc>* request, the IN-CSE shall validate the request and verify whether the values suggested to be set or to be updated for the Credential-ID included in the labels attribute - if any - and the CSE-ID of the Registrar CSE included in the ***From*** parameter still match with any of the allowed combinations of *App-ID* attribute and the AE-ID-Stem in the *link* attribute according to the applicable service subscription profile.

- **Step 007e:** When the validation and verification in **step 006e** completed successfully, the IN-CSE shall create *<AEAnnc>* resource with an Unstructured-CSE-relative-Resource-ID equal to the value of the provided AE-ID-Stem or update the *<AEAnnc>* resource in line with the parameters provided in step 005e. The IN-CSE shall change the *registrationS*tatus attribute of the old *<AEAnnc>* resource (tied to the old registration point) to INACTIVE, and shall transfer all child-resources under the old *<AEAnnc>* resource to the newly created or updated *<AEAnnc>* resource. The IN-CSE shall update all references to the SP-Relative-Resource-ID references (e.g. in Announce links, Notification targets, group Member ID, <accessControlPolicy> resource *OriginatorID* lists) tied to the prior AE registration point, so that these refer to the new AE registration point. The IN-CSE shall manage the change in AE registration point, as described in clause 10.2.16.1.

- **Step 008e:** Upon reception of a successful response from the IN-CSE, the Registrar CSE shall use the Unstructured-CSE-relative-Resource-ID equal to the AE-ID-Stem provided by the Registree AE for the *<AE>* resource to be created on the Registrar CSE and continue with action (4) of **step 002** of the non-registration related CREATE procedure in clause 10.1.2.

Case f) AE-ID-Stem starts with 'C' and AE includes an AE-ID-Stem (re-registration to a new Registrar CSE):

Condition: In **step 003** it was determined that the AE-ID-Stem value to be used for the Registree AE starts with an 'C' character and a specific AE-ID-Stem was provided with the CREATE request of the Registree AE. This case applies when the Registree AE is not supposed to use an M2M-SP-assigned AE-ID and wants to perform a re-registration to a new Registrar CSE and it wants to have its registration points tracked:

- **Step 005f:** The Registrar CSE shall use the Unstructured-CSE-relative-Resource-ID equal to the AE-ID-Stem in the ***From*** parameter for the <AE> resource to be created on the Registrar CSE,
- **Step 006f:** The Registrar CSE shall send a NOTIFY request to the IN-CSE. The ***Content*** parameter shall contain the SP-relative-Resource-ID at the prior registration point and the SP-relative-Resource-ID at the new registration point.
- **Step 007f:** Upon reception of the NOTIFY request, the IN-CSE shall manage the change in AE registration point, as described in clause 10.2.16.1.
- **Step 008f:** Upon reception of a successful response from the IN-CSE, the receiver shall then continue with action (4) of **step 002** of the non-registration related CREATE procedure in clause 10.1.2.

10.2.2.3 Retrieve <*AE*>

This procedure shall be used for retrieving the representation of the <*AE*> resource.

Table 10.2.2.3-1: <*AE*> RETRIEVE

<*AE*> RETRIEVE	
Information in Request message	All parameters defined in table 8.1.2-3
Processing at Originator before sending Request	According to clause 10.1.3
Processing at Receiver	According to clause 10.1.3
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content*****:** attributes of the <*AE*> resource as defined in clause 9.6.5
Processing at Originator after receiving Response	According to clause 10.1.3
Exceptions	According to clause 10.1.3

10.2.2.4 Update <*AE*>

This procedure shall be used for updating the attributes and the actual data of an <*AE*> resource.

Table 10.2.2.4-1: <*AE*> UPDATE

<*AE*> UPDATE	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content*****:** attributes of the <*AE*> resource as defined in clause 9.6.5 which need be updated
Processing at Originator before sending Request	According to clause 10.1.4
Processing at Receiver	According to clause 10.1.4 If the *pointOfAccess* attribute is updated and there are any messages in the buffer for store-and-forward procedure, Receiver shall send all buffered messages
Information in Response message	According to clause 10.1.4
Processing at Originator after receiving Response	According to clause 10.1.4
Exceptions	According to clause 10.1.4

10.2.2.5 Delete <*AE*>

This procedure shall be used for deleting the <*AE*> resource with all related information.

Table 10.2.2.5-1: <*AE*> DELETE

<*AE*> DELETE	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.5
Processing at Receiver	According to clause 10.1.5
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	According to clause 10.1.5
Exceptions	According to clause 10.1.5

Application Entity Deregistration is performed by requesting a Delete operation for the <*AE*> resource representing the Application Entity.

In case an <*AE*> resource hosted on a MN-CSE or ASN-CSE with AE-ID-Stem starting with "S" is requested to be deleted, the <*AEAnnc*> resource that was created on the IN-CSE during the initial registration of the associated Application Entity shall be updated with the value "INACTIVE" for the link attribute, indicating that the associated Application Entity is currently not registered. After this update of the <*AEAnnc*> resource is completed, the procedure for AE Deregistration shall follow the procedure described in this clause.

In case an <AE> resource with AE-ID-Stem not starting with "S" is requested to be deleted, the procedure for AE Deregistration follows the procedure described in clause 10.1.5.

10.2.2.6 CSE registration

CSE Registration towards a Registrar CSE is a process to enable a CSE to use services offered by the oneM2M System. During the process of CSE registration, a Registree CSE creates a <*remoteCSE*> resource as a child of the <*CSEBase*> resource of its Registrar CSE.

The Create procedure shall be not apply to <*CSEBase*>. <*CSEBase*> can be created via management operation not defined in this version of the specification.

The Update procedure shall not apply to <*CSEBase*>. <*CSEBase*> can be updated via management operation not defined in this version of the specification.

The Delete procedure shall not apply to <*CSEBase*>. <*CSEBase*> can be deleted via management operation not defined in this version of the specification.

10.2.2.7 Create *<remoteCSE>*

This procedure shall be used for creating a *<remoteCSE>* resource. It is part of the registration procedure for remote CSEs on the Registrar CSE (which is also the Hosting CSE), as described in this clause.

Table 10.2.2.7-1: *<remoteCSE>* CREATE

***<remoteCSE>* CREATE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***From:*** Originator CSE-ID **Content**: The resource content shall provide the information as defined in clause 9.6.4
Processing at Originator before sending Request	According to clause 10.2.2.7
Processing at Receiver	According to clause 10.2.2.7 with the following specific processing: If the Receiver CSE has registered to another CSE, the Receiver CSE shall send an update request to its Registrar CSE to add the CSE-IDs of the Originator CSE and the Originator CSE's descendants into the *descendantCSEs* attribute of the Receiver CSE's <remoteCSE> hosted by the Registrar CSE If the IN-CSE is the receiver and if the M2M SP policies do allow access to the CSEs across multiple domains, then the IN shall create the appropriate entry in the M2M SP's DNS for successfully registered CSE
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content***: Address of the created *<remoteCSE>* resource, according to clause 10.2.2.7
Processing at Originator after receiving Response	The Originator upon receipt of successful CREATE response message, shall create <remoteCSE> resource locally and thereafter, it may issue a Retrieve request to its Registrar CSE's <CSEBase> resource to update the optional attributes of locally created <remoteCSE> resource
Exceptions	According to clause 10.2.2.7

The procedure for CSE Registration follows the procedure described in clause 10.1.2, but with some deviations. Below is the detailed description on how to perform the CSE Registration and which part of the procedure deviates from the one described in clause 10.1.2.

The Registration procedure requires the creation of two resources (a <remoteCSE> on the Receiver CSE and a <remoteCSE> on the Originator CSE) rather than one resource. The Registration procedure is always initiated by a CSE in the field domain except in the inter-domain case described in clause 6.5.

Originator: The Originator shall be the registering CSE.

Receiver: The Receiver shall create the <remoteCSE> resource.

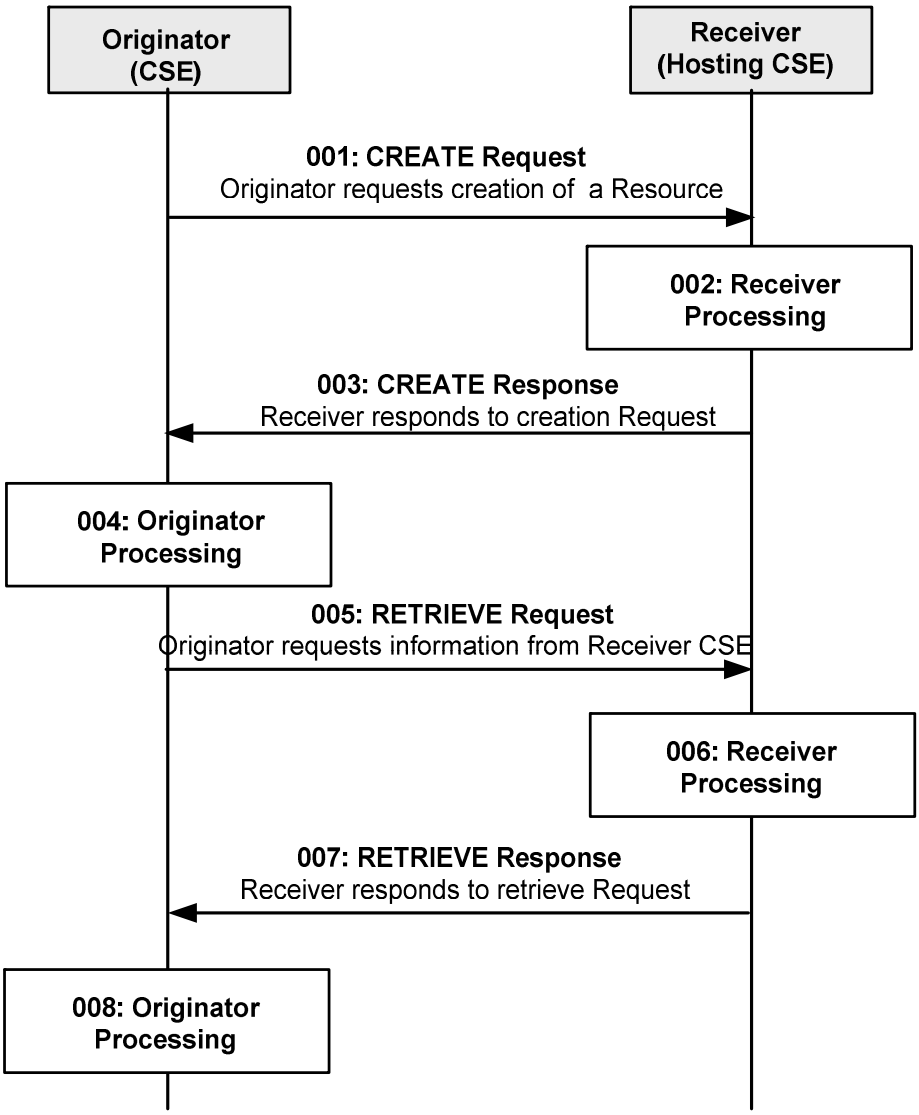


Figure 10.2.2.7-1: Procedure for CREATEing a <remoteCSE> Resource

All the parameters of the request and steps that are not indicated do not deviate from clause 10.1.2.

Step 001: The Originator shall send mandatory parameters and may send optional parameters in Request message for CREATE operation as specified in clause 8.1.2.

Step 002: The Receiver shall:

1) The registrar CSE shall allow unknown remote CSE to attempt to 'CREATE' when it was authenticated by credential provided by the entity. See ETSI TS 118 103 [2] further detail about authentication for the CSE.

2) Perform sub-steps: 2)-8), from step 002 from clause 10.1.2 are applicable. The access control which is sub-step 1) is omitted.

NOTE: Optionally, if the M2M Service Provider supports inter-domain communication, the Receiver could perform this step if the attribute *CSEBase* (part of the ***Content*** parameter of the request) contains the public domain of the CSE. The Receiver could construct the domain as described in clauses 6.4 and 6.5. The Receiver could add an AAA or AAAA record in DNS with the public domain name of the Originator CSE and the IP address of the IN-CSE associated with the Originator.

Step 003: See clause 10.1.2.

Step 004: The Originator, upon receipt of the successful CREATE response message, shall create a <remoteCSE> resource locally under its <CSEBase> resource. This resource is representing the Receiver CSE. The Originator shall provide the appropriate values to all mandatory parameters as described in clause 9.6.4.

Step 005: The Originator may issue a RETRIEVE Request towards the Receiver (same ***To*** as for the CREATE request message) to obtain the optional attributes of the <remoteCSE> resource created at the Originator in step 004 (e.g. *labels*, *accessControlPolicyIDs* attributes). The RETRIEVE procedure is described in clause 10.1.3.

See clause 8.1.2 for the information to be included in the Request message.

Step 006: The Receiver verifies that the Originator has the appropriate privileges to access the information.

Step 007: The Receiver sends a RETRIEVE response message, according to the procedure described in clause 10.1.3.

See clause 8.1.3 for the information to be included in the Response message.

Step 008: The Originator shall update the created <remoteCSE> resource for the Receiver with the information obtained in step 007.

General Exceptions:

All exceptions from clause 10.1.2 are applicable; in addition the following exception may occur:

1) The Originator does not have the privileges to retrieve the attributes of the Receiver CSE. The Receiver responds with an error.

10.2.2.8 Retrieve *<remoteCSE>*

This procedure shall be used for retrieving the representation of the *<remoteCSE>* resource with its attributes.

Table 10.2.2.8-1: *<remoteCSE>* RETRIEVE

***<remoteCSE>* RETRIEVE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.3
Processing at Receiver	According to clause 10.1.3
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content*:** attributes of the *<remoteCSE>* resource as the Originator requested
Processing at Originator after receiving Response	According to clause 10.1.3
Exceptions	According to clause 10.1.3

10.2.2.9 Update *<remoteCSE>*

This procedure shall be used for updating the attributes and the actual data of an *<remoteCSE>* resource.

Table 10.2.2.9-1: *<remoteCSE>* UPDATE

***<remoteCSE>* UPDATE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content*:** attributes of the *<remoteCSE>* resource as defined in clause 9.6.4 which need be updated
Processing at Originator before sending Request	According to clause 10.1.4
Processing at Receiver	According to clause 10.1.4 with the following specific processing: If the *descendantCSEs* attribute is updated, and the Receiver CSE has registered to another CSE, the Receiver CSE shall send an update request to its Registrar CSE to make the corresponding updates to the *descendantCSEs* attribute of the Receiver CSE's <remoteCSE> hosted by the Registrar CSE If the *pointOfAccess* attribute is updated and there are any messages in the buffer for store-and-forward procedure, Receiver shall send all buffered messages
Information in Response message	According to clause 10.1.4
Processing at Originator after receiving Response	According to clause 10.1.4
Exceptions	According to clause 10.1.4

10.2.2.10 Delete *<remoteCSE>*

This procedure shall be used for deleting the *<remoteCSE>* resource with all related information.

Table 10.2.2.10-1: *<remoteCSE>* DELETE

***<remoteCSE>* DELETE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.5
Processing at Receiver	According to clause 10.1.5 with the following specific processing: If the Receiver CSE has registered to another CSE, the Receiver CSE shall send an update request to its Registrar CSE to delete the CSE-IDs of the Originator CSE and the Originator CSE's descendants in the *descendantCSEs* attribute of the Receiver CSE's <remoteCSE> hosted by the Registrar CSE If the IN-CSE is the receiver and it has created an entry in the DNS to allow access to the CSE across multiple M2M domains, then it shall delete the entry from the DNS
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	According to clause 10.1.5
Exceptions	According to clause 10.1.5

The procedure for CSE Deregistration follows the procedure described in clause 10.1.5, but with some exceptions. Below is the detailed description on how to perform the CSE Deregistration and which part of the procedure deviates from the one described in clause 10.1.5.

The Deregistration procedure accompanies the deletion of two resources (a <remoteCSE> on the Hosting CSE and a <remoteCSE> on the Originator CSE) rather than one resource. The Deregistration procedure can be initiated by either Registree CSE or Registrar CSE.

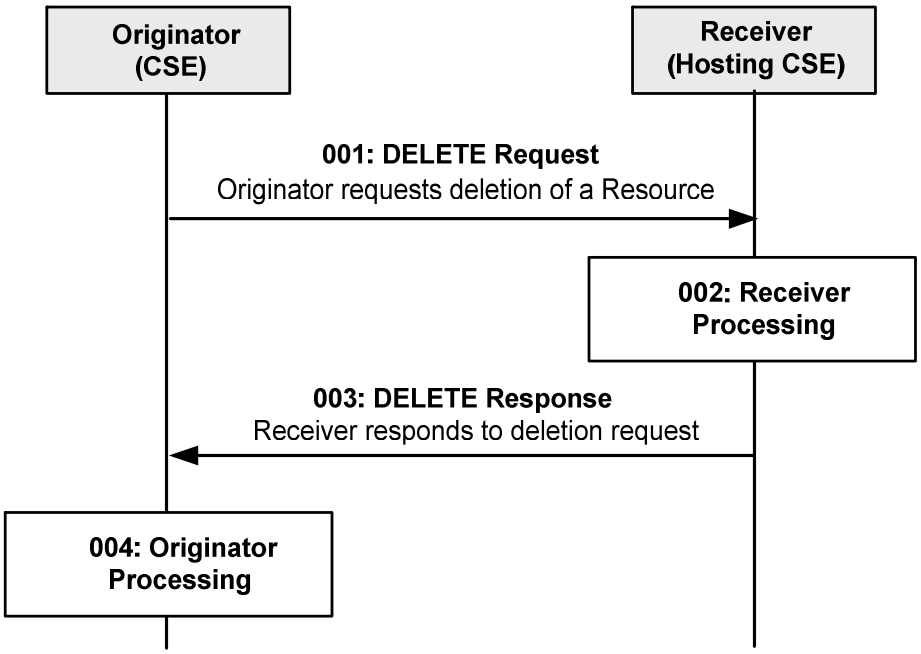


Figure 10.2.2.10-1: Procedure for DELETING a <remoteCSE> Resource

Step 001: See clause 10.1.5.

Step 002: See clause 10.1.5.

Step 003: See clause 10.1.5.

Step 004: The Originator, upon receipt of the DELETE response, shall delete a <remoteCSE> resource locally under its <CSEBase> resource.

General Exceptions:

All exceptions from 10.1.5 are applicable; in addition the following exception may occur:

1) If the Receiver rejects the DELETE request and responds with an error in the DELETE response, the Originator cannot perform the action described in the step 004.

10.2.2.11 Retrieve <*CSEBase*>

This procedure shall be used for retrieving the representation of the <*CSEBase*> resource with its attributes.

Table 10.2.2.11-1: <*CSEBase*> RETRIEVE

<*CSEBase*> RETRIEVE	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clauses 10.1.3 and 10.2.2.7
Processing at Receiver	According to clauses 10.1.3 and 10.2.2.7
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content*****:** attributes of the <*CSEBase*> resource as requested by the Originator
Processing at Originator after receiving Response	According to clauses 10.1.3 and 10.2.2.7 When this procedure is used during CSE Registration, a <*remoteCSE*> resource is created using the retrieved resource
Exceptions	According to clauses 10.1.3 and 10.2.2.7

10.2.3 Authorization

10.2.3.1 Introduction

This clause describes the procedures for creation, retrieval, update and deletion of the different types of authorization resources (i.e. <*accessControlPolicy*>, <*dynamicAuthorizationConsultation*>, <*role*> and <*token*>). These resources are used by a CSE to control access to other resources based on the different authorization methods as specified in the present document and in ETSI TS 118 103 [2].

When processing a request to a targeted resource, the Hosting CSE shall progress through the different types of authorization (if supported) as shown in figure 10.2.3.1-1.

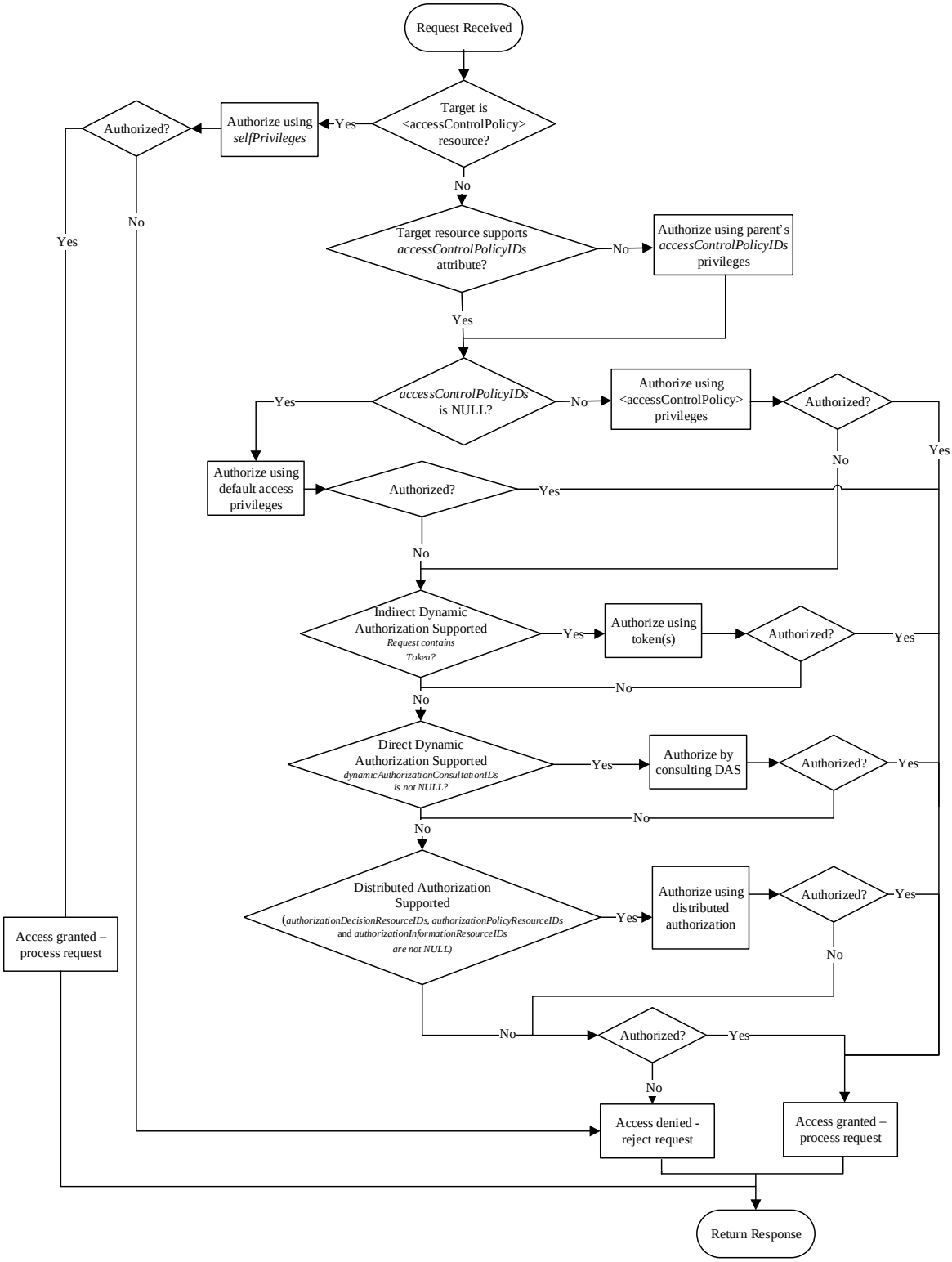


Figure 10.2.3.1-1: Different types of authorization flow chart

If the Hosting CSE receives a request targeting a resource of type <*accessControlPolicy*>, the Hosting CSE shall evaluate the set of access control rules configured within the *selfPrivileges* attribute and shall permit the operation if allowed by at least one access control rule. Otherwise the Hosting CSE shall deny access to the <*accessControlPolicy*> resource and return an error to the Originator. If a targeted resource is not of type <*accessControlPolicy*> and does not support an *accessControlPolicyIDs* attribute, the Hosting CSE shall evaluate the *accessControlPolicyIDs* of the targeted resource's parent. If a targeted resource is not of type <*accessControlPolicy*> and if the *accessControlPolicyIDs* is supported, the Hosting CSE shall evaluate the *accessControlPolicyIDs* of the targeted resource. If the *accessControlPolicyIDs* attribute is not NULL, the Hosting CSE shall evaluate the set of access control rules configured within the *privileges* attributes of each of the <*accessControlPolicy*> resources and shall permit the operation if allowed by at least one access control rule. If the *accessControlPolicyIDs* attribute is NULL, the Hosting CSE shall apply the default access privileges and grant access to the creator of the resource. Otherwise, if the request includes token information, the Hosting CSE may perform Indirect Dynamic Authorization if supported, as described in clause 11.5.3, and permit the operation if allowed. Otherwise, if the *dynamicAuthorizationConsultationIDs* attribute is not NULL, the Hosting CSE may perform Direct Dynamic Authorization if supported, as described in clause 11.5.2, and permit the operation if allowed. Otherwise, the Hosting CSE may perform Distributed Authorization if supported, as described in clause 11.6, and permit the operation if allowed.

10.2.3.2 Authorization using <*accessControlPolicy*>

This clause describes the procedures for creation, retrieval, update and deletion of the <*accessControlPolicy*> resource. The <*accessControlPolicy*> resource is used by a CSE to control access to other resources as specified in clauses 9.6.2 and 11.3.4.

10.2.3.3 Create <*accessControlPolicy*>

This procedure shall be used to create an <*accessControlPolicy*> resource.

Table 10.2.3.3-1: <*accessControlPolicy*> CREATE

<*accessControlPolicy*> CREATE	
Information in Request message	Same as clause 10.1.2.
Pre-Processing at Originator	Same as clause 10.1.2.
Processing at Receiver	Same as clause 10.1.2.
Information in Response message	Same as clause 10.1.2.
Post-Processing at Originator	Same as clause 10.1.2.
Exceptions	Same as clause 10.1.2.

10.2.3.4 Retrieve <*accessControlPolicy*>

This procedure shall be used to retrieve attributes and child resource information of the <*accessControlPolicy*> resource.

Table 10.2.3.4-1: <*accessControlPolicy*> RETRIEVE

<*accessControlPolicy*> RETRIEVE	
Information in Request message	Same as clause 10.1.3.
Pre-Processing at Originator	Same as clause 10.1.3.
Processing at Receiver	Addition to clause 10.1.3: • The Receiver shall check access control rules defined in *selfPrivileges* of the <*accessControlPolicy*> resource>.
Information in Response message	Same as clause 10.1.3.
Post-Processing at Originator	Same as clause 10.1.3.
Exceptions	Addition to clause 10.1.3.

10.2.3.5 Update <*accessControlPolicy*>

This procedure shall be used to update attributes information of the <*accessControlPolicy*> resource.

Table 10.2.3.5-1: <*accessControlPolicy*> UPDATE

<*accessControlPolicy*> UPDATE	
Information in Request message	Same as clause 10.1.4.
Pre-Processing at Originator	Same as clause 10.1.4.
Processing at Receiver	Addition to clause 10.1.4: • The Receiver shall check access control rules defined in *selfPrivileges* of the <*accessControlPolicy*> resource.
Information in Response message	Same as clause 10.1.4.
Post-Processing at Originator	Same as clause 10.1.4.
Exceptions	Addition to clause 10.1.4.

10.2.3.6 Delete <*accessControlPolicy*>

This procedure shall be used to delete the <*accessControlPolicy*> resource.

Table 10.2.3.6-1: <*accessControlPolicy*> DELETE

<*accessControlPolicy*> DELETE	
Information in Request message	Same as clause 10.1.5.
Pre-Processing at Originator	Same as clause 10.1.5.
Processing at Receiver	Addition to clause 10.1.5: • The Receiver shall check access control rules defined in *selfPrivileges* of the <*accessControlPolicy*> resource.
Information in Response message	Same as clause 10.1.5.
Post-Processing at Originator	Same as clause 10.1.5.
Exceptions	Addition to clause 10.1.5.

10.2.3.7 Authorization using <*dynamicAuthorizationConsultation*>

This clause describes the procedures for creation, retrieval, update and deletion of the <*dynamicAuthorizationConsultation*> resource. The <*dynamicAuthorizationConsultation*> resource is used by a CSE to control access to resources in a dynamic manner as specified in clause 11.5.

10.2.3.8 Create <*dynamicAuthorizationConsultation*>

This procedure shall be used to create a <*dynamicAuthorizationConsultation*> resource.

Table 10.2.3.8-1: <*dynamicAuthorizationConsultation*> CREATE

<dynamicAuthorizationConsultation> CREATE	
Information in Request message	Same as clause 10.1.2.
Pre-Processing at Originator	Same as clause 10.1.2.
Processing at Receiver	Same as clause 10.1.2.
Information in Response message	Same as clause 10.1.2.
Post-Processing at Originator	Same as clause 10.1.2.
Exceptions	Same as clause 10.1.2.

10.2.3.9 Retrieve <*dynamicAuthorizationConsultation*>

This procedure shall be used to retrieve attributes and child resource information of the <*dynamicAuthorizationConsultation*> resource.

Table 10.2.3.9-1: <*dynamicAuthorizationConsultation*> RETRIEVE

<dynamicAuthorizationConsultation> RETRIEVE	
Information in Request message	Same as clause 10.1.3.
Pre-Processing at Originator	Same as clause 10.1.3.
Processing at Receiver	Same as clause 10.1.3.
Information in Response message	Same as clause 10.1.3.
Post-Processing at Originator	Same as clause 10.1.3.
Exceptions	Addition to clause 10.1.3.

10.2.3.10 Update <*dynamicAuthorizationConsultation*>

This procedure shall be used to update attributes information of the <*dynamicAuthorizationConsultation*> resource.

Table 10.2.3.10-1: <*dynamicAuthorizationConsultation*> UPDATE

<dynamicAuthorizationConsultation> UPDATE	
Information in Request message	Same as clause 10.1.4.
Pre-Processing at Originator	Same as clause 10.1.4.
Processing at Receiver	Addition to clause 10.1.4.
Information in Response message	Same as clause 10.1.4.
Post-Processing at Originator	Same as clause 10.1.4.
Exceptions	Addition to clause 10.1.4.

10.2.3.11 Delete <*dynamicAuthorizationConsultation*>

This procedure shall be used to delete the <*dynamicAuthorizationConsultation*> resource.

Table 10.2.3.11-1: <*dynamicAuthorizationConsultation*> DELETE

<dynamicAuthorizationConsultation> DELETE	
Information in Request message	Same as clause 10.1.5.
Pre-Processing at Originator	Same as clause 10.1.5.
Processing at Receiver	Addition to clause 10.1.5.
Information in Response message	Same as clause 10.1.5.
Post-Processing at Originator	Same as clause 10.1.5.
Exceptions	Addition to clause 10.1.5.

10.2.3.12 Authorization using <*role*>

This clause describes the procedures for creation, retrieval, update and deletion of the <*role*> resource. The <*role*> resource is used to assign a role to an AE or CSE. The role is used to control access to resources in a role-based manner as specified in ETSI TS 118 103 [2].

10.2.3.13 Create <*role*>

This procedure shall be used for creating a <*role*> resource.

Table 10.2.3.13-1: *<role>* CREATE

***<role>* CREATE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***From:*** Identifier of the AE that initiates the request ***To:*** Address the resource where the <*role*> resource is intended to be created ***Content:*** The resource content shall provide the information as defined in clause 9.6.38
Processing at Originator before sending Request	According to clause 10.1.2
Processing at Receiver	According to clause 10.1.2
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content*:** Address of the created <role> resource, according to clause 10.1.2
Processing at Originator after receiving Response	According to clause 10.1.2
Exceptions	According to clause 10.1.2

10.2.3.14 Retrieve <*role*>

This procedure shall be used for retrieving the attributes of a <*role*> resource.

Table 10.2.3.14-1: *<role>* RETRIEVE

***<role>* RETRIEVE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content*:** void
Processing at Originator before sending Request	According to clause 10.1.3
Processing at Receiver	According to clause 10.1.3
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content*:** attributes of the <role> resource as defined in clause 9.6.38
Processing at Originator after receiving Response	According to clause 10.1.3
Exceptions	According to clause 10.1.3

10.2.3.15 Update <*role*>

This procedure shall be used for updating attributes of a <*role*> resource.

Table 10.2.3.15-1: *<role>* UPDATE

***<role>* UPDATE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.4
Processing at Receiver	According to clause 10.1.4
Information in Response message	According to clause 10.1.4
Processing at Originator after receiving Response	According to clause 10.1.4
Exceptions	According to clause 10.1.4

10.2.3.16 Delete <*role*>

This procedure shall be used for deleting an existing <*role*> resource.

Table 10.2.3.16-1: <*role*> DELETE

<*role*> DELETE	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.5
Processing at Receiver	According to clause 10.1.5
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	According to clause 10.1.5
Exceptions	According to clause 10.1.5

10.2.3.17 Authorization using <*token*>

This clause describes the procedures for creation, retrieval, update and deletion of the <*token*> resource. The <*token*> resource is used for storing a token that is issued to an AE or CSE. The token is used to control access to resources in a dynamic manner as specified in clause 11.5 of the present document and in ETSI TS 118 103 [2].

10.2.3.18 Create <*token*>

This procedure shall be used for creating a <*token*> resource.

Table 10.2.3.18-1: <*token*> CREATE

<*token*> CREATE	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***From:*** Identifier of the AE that initiates the request ***To:*** Address the resource where the <*token*> resource is intended to be created ***Content****:* The resource content shall provide the information as defined in clause 9.6.39
Processing at Originator before sending Request	According to clause 10.1.2
Processing at Receiver	According to clause 10.1.2
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content*****:** Address of the created <token> resource, according to clause 10.1.2
Processing at Originator after receiving Response	According to clause 10.1.2
Exceptions	According to clause 10.1.2

10.2.3.19 Retrieve <*token*>

This procedure shall be used for retrieving the attributes of a <*token*> resource.

Table 10.2.3.19-1: <*token*> RETRIEVE

<*token*> RETRIEVE	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content*****:** void
Processing at Originator before sending Request	According to clause 10.1.3
Processing at Receiver	According to clause 10.1.3
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content*****:** attributes of the <token> resource as defined in clause 9.6.39
Processing at Originator after receiving Response	According to clause 10.1.3
Exceptions	According to clause 10.1.3

10.2.3.20 Update <*token*>

This procedure shall be used for updating attributes of a <*token*> resource.

Table 10.2.3.20-1: <*token*> UPDATE

<*token*> UPDATE	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.4
Processing at Receiver	According to clause 10.1.4
Information in Response message	According to clause 10.1.4
Processing at Originator after receiving Response	According to clause 10.1.4
Exceptions	According to clause 10.1.4

10.2.3.21 Delete <*token*>

This procedure shall be used for deleting an existing <*token*> resource.

Table 10.2.3.21-1: <*token*> DELETE

<*token*> DELETE	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.5
Processing at Receiver	According to clause 10.1.5
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	According to clause 10.1.5
Exceptions	According to clause 10.1.5

10.2.3.22 Authorization using <*authorizationDecision*>

Each <*authorizationDecision*> resource represents an entrance of a Policy Decision Point (PDP) that is responsible for making access control decisions.

Multiple <*authorizationDecision*> resources can be created under one <*CSEBase*> resource. The access control policies associated to an <*authorizationDecision*> resource can be used to group access control decision requesters, i.e. which CSEs can retrieve access control decisions from a given <*authorizationDecision*> resource.

The resource specific attributes of an <*authorizationDecision*> resource can be classified into two categories according to their usages. One category is used for describing access control decisions, e.g. *decision* and *status* attributes. The other category is used for describing access control decision requests, e.g. *to*, *from*, *operation*, *roleIDs* and so on.

An UPDATE operation on an <*authorizationDecision*> resource may trigger an access control decision making process. After making an access control decision, the access control decision or error status is returned back to the decision requester via an UPDATE response.

The details of distributed authorization procedures are described in ETSI TS 118 103 [2].

10.2.3.23 Create <*authorizationDecision*>

This procedure shall be used for creating an <*authorizationDecision*> resource.

Table 10.2.3.23-1: *<authorizationDecision>* CREATE

***<authorizationDecision>* CREATE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content**:* The resource content shall provide the information as defined in clause 9.6.41, and all the values of resource specific attributes shall be set to null.
Processing at Originator before sending Request	According to clause 10.1.2
Processing at Receiver	According to clause 10.1.2
Information in Response message	According to clause 10.1.2
Processing at Originator after receiving Response	According to clause 10.1.2
Exceptions	According to clause 10.1.2

10.2.3.24 Retrieve <*authorizationDecision*>

This procedure shall be used for retrieving <*authorizationDecision*> resource.

Table 10.2.3.24-1: *<authorizationDecision>* RETRIEVE

***<authorizationDecision>* RETRIEVE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content*:** void
Processing at Originator before sending Request	According to clause 10.1.3
Processing at Receiver	According to clause 10.1.3
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content*:** attributes of the <*authorizationDecision*> resource as defined in clause 9.6.38
Processing at Originator after receiving Response	According to clause 10.1.3
Exceptions	According to clause 10.1.3

10.2.3.25 Update <*authorizationDecision*>

This procedure shall be used for updating attributes of an <*authorizationDecision*> resource.

Originator: The Originator shall request to obtain an access control decision by using UPDATE operation on an <*authorizationDecision*> resource. The access control decision request shall be specified with resource specific attributes except the *decision* attribute.

Receiver: The Receiver shall execute an access control decision making process according to the access control decision request provided in the UPDATE request and return the access control decision in the UPDATE response.

Table 10.2.3.25-1: *<authorizationDecision>* UPDATE

***<authorizationDecision>* UPDATE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply.
Processing at Originator before sending Request	According to clause 10.1.4 with the following additions: ***Content:*** The representation of an access control decision request constructed using updated attributes. See clause 7 in ETSI TS 118 103 [2] for the mandatory and optional parameters.
Processing at Receiver	According to clause 10.1.4 with the following additions: • Check the validity of the access control decision request constructed using updated attributes. See clause 7 in ETSI TS 118 103 [2] for the mandatory and optional parameters. • Obtain applicable access control policies and requested access control information, and then make an access control decision. See clause 7 in ETSI TS 118 103 [2] for more details. • Update the *decision* and *status* attributes with the access control decision evaluation result. See clause 7 in ETSI TS 118 103 [2] for possible access control decision evaluation results. • After sending the response message, all the resource specific attributes shall be deleted.
Information in Response message	According to clause 10.1.4
Processing at Originator after receiving Response	According to clause 10.1.4
Exceptions	According to clause 10.1.4

10.2.3.26 Delete *<authorizationDecision>*

This procedure shall be used for deleting an existing *<authorizationDecision>* resource.

Table 10.2.3.26-1: *<authorizationDecision>* DELETE

***<authorizationDecision>* DELETE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.5
Processing at Receiver	According to clause 10.1.5
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	According to clause 10.1.5
Exceptions	According to clause 10.1.5

10.2.3.27 Authorization using *<authorizationPolicy>*

Each *<authorizationPolicy>* resource represents an entrance of a Policy Retrieval Point (PRP) that is responsible for retrieving access control policies.

Multiple *<authorizationPolicy>* resources can be created under one *<CSEBase>* resource. The access control policies associated to an *<authorizationPolicy>* resource can be used to group access control policy requesters, i.e. which CSEs can retrieve access control policies from a given *<authorizationPolicy>* resource.

The resource specific attributes of an *<authorizationPolicy>* resource can be classified into two categories according to their usages. One category is used for describing access control policies, e.g. *policies* and *combiningAlgorithm* attributes. The other category is used for describing access control policy requests, e.g. *to* attribute.

An UPDATE operation on an *<authorizationPolicy>* resource may trigger an access control policy retrieving process. After obtaining the access control policies and policy combining algorithm, the access control policies and policy combining algorithm or error status is returned back to the policy requester via an UPDATE response.

The details of distributed authorization procedures are described in ETSI TS 118 103 [2].

10.2.3.28 Create <*authorizationPolicy*>

This procedure shall be used for creating an <*authorizationPolicy*> resource.

Table 10.2.3.28-1: <*authorizationPolicy*> CREATE

<*authorizationPolicy*> CREATE	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content**:* The resource content shall provide the information as defined in clause 9.6.42, and all the values of resource specific attributes shall be set to null
Processing at Originator before sending Request	According to clause 10.1.2
Processing at Receiver	According to clause 10.1.2
Information in Response message	According to clause 10.1.2
Processing at Originator after receiving Response	According to clause 10.1.2
Exceptions	According to clause 10.1.2

10.2.3.29 Retrieve <*authorizationPolicy*>

This procedure shall be used for retrieving <*authorizationPolicy*> resource.

Table 10.2.3.29-1: <*authorizationPolicy*> RETRIEVE

<*authorizationPolicy*> RETRIEVE	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content*****:** void
Processing at Originator before sending Request	According to clause 10.1.3
Processing at Receiver	According to clause 10.1.3
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content*****:** attributes of the <*authorizationPolicy*> resource as defined in clause 9.6.38
Processing at Originator after receiving Response	According to clause 10.1.3
Exceptions	According to clause 10.1.3

10.2.3.30 Update <*authorizationPolicy*>

This procedure shall be used for updating attributes of an <*authorizationPolicy*> resource.

Originator: The Originator shall request to obtain access control policies by using UPDATE operation on an <*authorizationPolicy*> resource. The access control policy request shall be specified with resource specific attributes except the *policies and combiningAlgorithm* attributes.

Receiver: The Receiver shall execute an access control policy retrieving process according to the access control policy request provided in the UPDATE request and return the access control policies and policy combining algorithm in the UPDATE response.

Table 10.2.3.30-1: *<authorizationPolicy>* UPDATE

***<authorizationPolicy>* UPDATE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply.
Processing at Originator before sending Request	According to clause 10.1.4 with the following additions: ***Content:*** The representation of an access control policy request constructed using updated attributes. See clause 7 in ETSI TS 118 103 [2] for the mandatory and optional parameters.
Processing at Receiver	According to clause 10.1.4 with the following additions: • Check the validity of the access control policy request constructed using updated attributes. See clause 7 in ETSI TS 118 103 [2] for the mandatory and optional parameters. • Obtain applicable access control policies and policy combining algorithm. See clause 7 in ETSI TS 118 103 [2] for more details. • Update the *policies* and *combiningAlgorithm* and *status* attributes with the access control policy retrieval result. See clause 7 in ETSI TS 118 103 [2] for possible policy combining algorithms. • After sending the response message, all the resource specific attributes shall be deleted.
Information in Response message	According to clause 10.1.4.
Processing at Originator after receiving Response	According to clause 10.1.4.
Exceptions	According to clause 10.1.4.

10.2.3.31 Delete *<authorizationPolicy>*

This procedure shall be used for deleting an existing *<authorizationPolicy>* resource.

Table 10.2.3.31-1: *<authorizationPolicy>* DELETE

***<authorizationPolicy>* DELETE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.5
Processing at Receiver	According to clause 10.1.5
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	According to clause 10.1.5
Exceptions	According to clause 10.1.5

10.2.3.32 Authorization using *<authorizationInformation>*

Each *<authorizationInformation>* resource represents an entrance of a Policy Information Point (PIP) that is responsible for retrieving access control information, e.g. a role or token.

Multiple *<authorizationInformation>* resources can be created under one *<CSEBase>* resource. The access control policies associated to an *<authorizationInformation>* resource can be used to group access control information requesters, i.e. which CSEs can retrieve access control information from a given *<authorizationInformation>* resource.

The resource specific attributes or child resources of an *<authorizationInformation>* resource can be classified into two categories according to their usages. One category is used for describing access control information, e.g. *<role>* and *<token>* resources. The other category is used for describing access control Information requests, e.g. *from, roleIDs* and *tokenIDs* attributes.

An UPDATE operation on an *<authorizationInformation>* resource may trigger an access control information retrieving process. After obtaining the access control information, the access control information or error status is returned back to the information requester via an UPDATE response.

The details of distributed authorization procedures are described in ETSI TS 118 103 [2].

10.2.3.33 Create <*authorizationInformation*>

This procedure shall be used for creating an <*authorizationInformation*> resource.

Table 10.2.3.33-1: <*authorizationInformation*> CREATE

<*authorizationInformation*> CREATE	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content**:* The resource content shall provide the information as defined in clause 9.6.43, and all the values of resource specific attributes shall be set to null, and any resource specific child resource shall not be created.
Processing at Originator before sending Request	According to clause 10.1.2.
Processing at Receiver	According to clause 10.1.2.
Information in Response message	According to clause 10.1.2.
Processing at Originator after receiving Response	According to clause 10.1.2.
Exceptions	According to clause 10.1.2.

10.2.3.34 Retrieve <*authorizationInformation*>

This procedure shall be used for retrieving <*authorizationInformation*> resource.

Table 10.2.3.34-1: <*authorizationInformation*> RETRIEVE

<*authorizationInformation*> RETRIEVE	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content*:** void
Processing at Originator before sending Request	According to clause 10.1.3
Processing at Receiver	According to clause 10.1.3
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content*:** attributes of the <*authorizationInformation*> resource as defined in clause 9.6.38
Processing at Originator after receiving Response	According to clause 10.1.3
Exceptions	According to clause 10.1.3

10.2.3.35 Update <*authorizationInformation*>

This procedure shall be used for updating attributes and child resources of an <*authorizationInformation*> resource.

Originator: The Originator shall request to obtain access control information by using UPDATE operation on an <*authorizationInformation*> resource. The access control information request shall be specified with resource specific attributes.

Receiver: The Receiver shall execute an access control information retrieving process according to the access control information request provided in the UPDATE request, and return the access control information in the UPDATE response. The access control information shall be specified with <*role*> and/or <*token*> child resources.

Table 10.2.3.35-1: <*authorizationInformation*> UPDATE

<*authorizationInformation*> UPDATE	
Information in Request message	All parameters defined in table 8.1.2-3 apply.
Processing at Originator before sending Request	According to clause 10.1.4 with the following additions: ***Content:*** The representation of an access control information request constructed using updated attributes. See clause 7 in ETSI TS 118 103 [2] for the mandatory and optional parameters.
Processing at Receiver	According to clause 10.1.4 with the following additions: • Check the validity of the access control information request constructed using updated attributes. See clause 7 in ETSI TS 118 103 [2] for the mandatory and optional parameters. • Obtain applicable access control information. The details of this process are described in ETSI TS 118 103 [2]. • Create <*role*> and/or <*token*> child resources and update the *status* attributes according to the access control information retrieval result. • After sending the response message, all the resource specific attributes and <*role*> and/or <*token*> child resources shall be deleted.
Information in Response message	According to clause 10.1.4 with the following additions: The <*role*> and/or <*token*> child resources shall also be returned.
Processing at Originator after receiving Response	According to clause 10.1.4.
Exceptions	According to clause 10.1.4.

10.2.3.36 Delete <*authorizationInformation*>

This procedure shall be used for deleting an existing <*authorizationInformation*> resource.

Table 10.2.3.36-1: <*authorizationInformation*> DELETE

<*authorizationInformation*> DELETE	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.5
Processing at Receiver	According to clause 10.1.5
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	According to clause 10.1.5
Exceptions	According to clause 10.1.5

10.2.4 Data management

10.2.4.1 Introduction

Data management is provided with different sets of Content Sharing Resources and those sets provide different functionalities that can be utilized depending on application services. The following lists each resource type set.

- container, contentInstance, latest, oldest
- flexContainer
- timeSeries, timeSeriesInstance, latest, oldest

10.2.4.2 Data management using <*container*> and <*contentInstance*>

A <container> resource provides the data instance container for the children <contentInstance> resources. An AE can configure the container (e.g. maxNrOfInstances) and a CSE provides meta information (e.g. currentNrOfInstances) of the container. A <contentInstance> resource is immutable once created and it stores application data within a single attribute (i.e. the *content*) regardless the data structure and format.

The <latest> and <oldest> virtual resources are used to easily retrieve and delete the latest and oldest <contentInstance> resource in a <container> resource, respectively, without any knowledge of all the <contentInstance> resources in the container.

10.2.4.3 Create *<container>*

This procedure shall be used for creating a *<container>* resource.

Table 10.2.4.3-1: *<container>* CREATE

***<container>* CREATE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content**:* The resource content shall provide the information as defined in clause 9.6.6
Processing at Originator before sending Request	According to clause 10.1.2
Processing at Receiver	According to clause 10.1.2
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content*:** Address of the created <container> resource, according to clause 10.1.2
Processing at Originator after receiving Response	According to clause 10.1.2
Exceptions	According to clause 10.1.2

10.2.4.4 Retrieve *<container>*

This procedure shall be used for retrieving the attributes of a *<container>* resource.

Table 10.2.4.4-1: *<container>* RETRIEVE

***<container>* RETRIEVE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content*:** void.
Processing at Originator before sending Request	According to clause 10.1.3.
Processing at Receiver	The Receiver shall verify the existence (including ***Filter Criteria*** checking, if it is given) of the target resource or the attribute and check if the Originator has appropriate privileges to retrieve information stored in the resource/attribute. When the child <contentInstance> resource has to be part of the response (ResultContent is child-resources or attributes+child-resources) and there is no *<contentInstance>* resource in the parent or if all existing ones are obsolete then 2 situations: a) There is a subscription on the <container> resource with the notificationEventType 'e)' set (table 9.6.8-3) so a notification is triggered, a timer shall be set and the Receiver shall delay the response until a <contentInstance> resource is available in the <container> resource, or until the timer expires; in that last case the Receiver shall respond with an error. If the ***Result Expiration Timestamp*** parameter is received from the Originator, the timer should be set to enforce this parameter, otherwise, the timer is set, based on the local policy configured at the Hosting CSE. b) There is no subscription on the <container> resource with the notificationEventType 'e)' set, then the Receiver shall respond with an error. Otherwise clause 10.1.3 applies.
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content*:** attributes of the <container> resource as defined in clause 9.6.6.
Processing at Originator after receiving Response	According to clause 10.1.3.
Exceptions	According to clause 10.1.3. In addition : a timer has expired. The Receiver responds with an error.

10.2.4.5 Update <*container*>

This procedure shall be used for updating the attributes and the actual data of a <*container*> resource.

Table 10.2.4.5-1: <*container*> UPDATE

<*container*> UPDATE	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content*:** attributes of the <container> resource as defined in clause 9.6.6 which need be updated
Processing at Originator before sending Request	According to clause 10.1.4
Processing at Receiver	According to clause 10.1.4
Information in Response message	According to clause 10.1.4
Processing at Originator after receiving Response	According to clause 10.1.4
Exceptions	According to clause 10.1.4

10.2.4.6 Delete <*container*>

This procedure shall be used for deleting a <*container*> resource.

Table 10.2.4.6-1: <*container*> DELETE

<*container*> DELETE	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.5
Processing at Receiver	According to clause 10.1.5.
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	According to clause 10.1.5
Exceptions	According to clause 10.1.5

10.2.4.7 Create <*contentInstance*>

This procedure shall be used for creating a <*contentInstance*> resource.

Table 10.2.4.7-1: *<contentInstance>* CREATE

***<contentInstance>* CREATE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content:*** The resource content shall provide the information as defined in clause 9.6.7.
Processing at Originator before sending Request	According to clause 10.1.2.
Processing at Receiver	According to clause 10.1.2. If the newly created <*contentInstance*> resource violates any of the policies defined in the parent <*container*> resource (e.g. *maxNrOfInstances* or *maxByteSize*), then the oldest <*contentInstance*> resources shall be removed from the <*container*> to enable the creation of the new <*contentInstance*> *resource*. If the hosting CSE has the capability to duplicate the actual data in semantic triples complying to an ontology that it supports, it may: 1) represent the actual data contained in the *content* attribute to semantic triples (e.g. RDF triples), 2) create a <*semanticDescriptor*> child resource for the <*contentInstance*> resource with its *descriptor* attribute set to these semantic triples generated in 1). • As an example to enable the hosting CSE to duplicate the actual data in semantic triples, the parent <*container*> resource contains sufficient semantic information in one of its <*semanticDescriptor*> child resources. Specifically, the *descriptor* attribute of the <*container*>/<*semanticDescriptor*> resource contains triples to describe semantic information about the actual data to be contained in any created <*contentInstance*>; as such, the hosting CSE is able to duplicate the actual data in semantic triples. For example, if the <*container*> is for storing readings from a temperature sensor (i.e. each <*contentInstance*> corresponds to a different reading), the *descriptor* attribute can contain triples to describe, for example, the type of actual data is temperature reading, the encoding of the actual data is a Base64-encoded string, and the unit of the actual data is Celsius.
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: • ***Content:*** Address of the created <*contentInstance*> resource, according to clause 10.1.2.
Processing at Originator after receiving Response	According to clause 10.1.2.
Exceptions	According to clause 10.1.2.

10.2.4.8 Retrieve <*contentInstance*>

This procedure shall be used for retrieving the attributes of a <*contentInstance*> resource.

Table 10.2.4.8-1: *<contentInstance>* RETRIEVE

***<contentInstance>* RETRIEVE**	
Information in Request message	According to clause 10.1.3.
Processing at Originator before sending Request	According to clause 10.1.3.
Processing at Receiver	According to clause 10.1.3. If the *disableRetrieval* attribute of the parent <*container*> resource was set as 'TRUE', then the RETRIEVE request shall be rejected (see note).
Information in Response message	All parameters defined in table 8.1.3-1 apply with specific details for: ***Content:*** Attributes of the <*contentInstance*> resources as defined in clause 9.6.7.
Processing at Originator after receiving Response	According to clause 10.1.3.
Exceptions	According to clause 10.1.3.
NOTE: Notification regarding <*subscription*> on the parent <container> resource shall be.	

10.2.4.9 Update <*contentInstance*>

The Update operation shall not apply to <*contentInstance*> resource.

10.2.4.10 Delete <*contentInstance*>

This procedure shall be used for deleting a <*contentInstance*> resource residing under a <*container*> resource.

Table 10.2.4.10-1: <*contentInstance*> DELETE

<*contentInstance*> DELETE	
Information in Request message	All parameters defined in table 8.1.2-3 apply.
Processing at Originator before sending Request	According to clause 10.1.5.
Processing at Receiver	According to clause 10.1.5. The Receiver shall delete the <*contentInstance*> resource.
Information in Response message	According to clause 10.1.5.
Processing at Originator after receiving Response	According to clause 10.1.5.
Exceptions	According to clause 10.1.5.

10.2.4.11 Retrieve <*latest*>

If *locationID* of <*container*> is configured, the procedure specified in clause 10.2.9.7 shall apply.

If *locationID* of <*container*> is not configured, this procedure shall apply to the latest <*contentInstance*> resource among all existing <*contentInstance*> resources in the parent <*container*> resource. If there is no <*contentInstance*> resource in the parent, then the Receiver shall response with an error.

This procedure is the same as the procedures in clause 10.2.4.8 <*contentInstance*> RETRIEVE.

10.2.4.12 Delete <*latest*>

This procedure shall apply to the latest <*contentInstance*> resource among all existing <*contentInstance*> resources in the parent <*container*> resource. If there is no <*contentInstance*> resource in the parent, then the Receiver shall response with an error.

After deletion, the <*latest*> contentInstance will point to the latest <*contentlnstance*> among all remaining <*contentInstance*> resources in the parent <*container*> resource. This procedure is the same as the procedures in clause 10.2.4.10 <*contentInstance*> DELETE.

10.2.4.13 Retrieve <*oldest*>

This procedure shall apply to the oldest <*contentInstance*> resource among all existing <*contentInstance*> resources in the parent <*container*> resource. If there is no <*contentInstance*> resource in the parent, then the Receiver shall response with an error.

This procedure is the same as the procedures in clause 10.2.4.8 <*contentInstance*> RETRIEVE.

10.2.4.14 Delete <*oldest*>

This procedure shall apply to the oldest <*contentInstance*> resource among all existing <*contentInstance*> resources in the parent <*container*> resource. If there is no <*contentInstance*> resource in the parent, then the Receiver shall response with an error.

After deletion, the <*oldest*> contentInstance will point to the oldest <*contentlnstance*> among all remaining <*contentInstance*> resources in the parent <*container*> resource.

This procedure is the same as the procedure in as clause 10.2.4.10 <*contentInstance*> DELETE.

10.2.4.15 Data management using <*flexContainer*>

A <flexContainer> resource can store application data in custom attributes and provide limited meta information (e.g. *contentSize*) compared to a <container> resource. With the custom attributes, application data can be separately stored in different attributes that can be leveraged to discover the resource with the attribute value or subscribe to specific attribute(s). A <flexContainer> resource can have one or more child(ren) <flexContainers> resource(s) in multiple levels.

10.2.4.16 Create <*flexContainer*>

This procedure shall be used for creating a <*flexContainer*> resource.

Table 10.2.4.16-1: <*flexContainer*> CREATE

<*flexContainer*> CREATE	
Information in Request message	According to clause 10.1.2
Processing at Originator before sending Request	According to clause 10.1.2
Processing at Receiver	According to clause 10.1.2
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content*:** Address of the created <*flexContainer*> resource, according to clause 10.1.2
Processing at Originator after receiving Response	According to clause 10.1.2
Exceptions	According to clause 10.1.2 with the following addition: • The parent resource type of this newly created <flexContainer> resource shall follow the definition in clause 9.6.1.2.2 (Specializations of <*flexContainer*>).

10.2.4.17 Retrieve <*flexContainer*>

This procedure shall be used for retrieving the attributes of a <*flexContainer*> resource.

Table 10.2.4.17-1: <*flexContainer*> RETRIEVE

<*flexContainer*> RETRIEVE	
Information in Request message	All parameters defined in table 8.1.2-2 apply with the specific details for: ***Content*:** void
Processing at Originator before sending Request	According to clause 10.1.3
Processing at Receiver	According to clause 10.1.3
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content*:** attributes of the <*flexContainer*> resource as defined in clause 9.6.6
Processing at Originator after receiving Response	According to clause 10.1.3
Exceptions	According to clause 10.1.3

10.2.4.18 Update <*flexContainer*>

This procedure shall be used for updating the attributes and the actual data of a <*flexContainer*> resource.

Table 10.2.4.18-1: <*flexContainer*> UPDATE

<*flexContainer*> UPDATE	
Information in Request message	All parameters defined in table 8.1.2-2 apply with the specific details for: ***Content*:** attributes of the <*flexContainer*> resource as defined in clause 9.6.6 which need be updated
Processing at Originator before sending Request	According to clause 10.1.4
Processing at Receiver	According to clause 10.1.4
Information in Response message	According to clause 10.1.4
Processing at Originator after receiving Response	According to clause 10.1.4
Exceptions	According to clause 10.1.4

10.2.4.19 Delete <*flexContainer*>

This procedure shall be used for deleting a <*flexContainer*> resource.

Table 10.2.4.19-1: <*flexContainer*> DELETE

<*flexContainer*> DELETE	
Information in Request message	All parameters defined in table 8.1.2-2 apply
Processing at Originator before sending Request	According to clause 10.1.5
Processing at Receiver	According to clause 10.1.5
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	According to clause 10.1.5
Exceptions	According to clause 10.1.5

10.2.4.20 Data management using <*timeSeries*> and <*timeSeriesInstance*>

<timeSeries> and <timeSeriesInstance> resources are similar in function to <container> and <contentInstance> resources, however a <timeSeries> provides missing data detection while monitoring the *dataGenerationTime* attribute of a <timeSeriesInstance> resource. If a <timeSeries> resource is subscribed to, a missing data detection event results in a notification to the subscriber.

10.2.4.21 Create <*timeSeries*>

This procedure shall be used for creating a <*timeSeries*> resource.

Table 10.2.4.21-1: <*timeSeries*> CREATE

<*timeSeries*> CREATE	
Information in Request message	All parameters defined in table 8.1.2-2 apply with the specific details for: ***Content:*** The resource content shall provide the information as defined in clause 9.6.36
Processing at Originator before sending Request	According to clause 10.1.2
Processing at Receiver	According to clause 10.1.2: • If the *periodicInterval* attribute is set and the *missingDataDetect* attribute is TRUE, the Hosting CSE shall begin the procedure defined in 10.2.4.29
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content*:** Address of the created <*timeSeries*> resource, according to clause 10.1.2
Processing at Originator after receiving Response	According to clause 10.1.2
Exceptions	According to clause 10.1.2

10.2.4.22 Retrieve <*timeSeries*>

This procedure shall be used for retrieving the attributes of a <*timeSeries*> resource.

Table 10.2.4.22-1: <*timeSeries*> RETRIEVE

<*timeSeries*> RETRIEVE	
Information in Request message	All parameters defined in table 8.1.2-2 apply with the specific details for: ***Content*:** void
Processing at Originator before sending Request	According to clause 10.1.3
Processing at Receiver	According to clause 10.1.3
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content*:** attributes of the <*timeSeries*> resource as defined in clause 9.6.36
Processing at Originator after receiving Response	According to clause 10.1.3
Exceptions	According to clause 10.1.3

10.2.4.23 Update <*timeSeries*>

This procedure shall be used for updating the attributes in a <*timeSeries*> resource.

Table 10.2.4.23-1: <*timeSeries*> UPDATE

<*timeSeries*> UPDATE	
Information in Request message	All parameters defined in table 8.1.2-2 apply with the specific details for: ***Content*:** attributes of the <*timeSeries*> resource as defined in clause 9.6.36 which need be updated
Processing at Originator before sending Request	According to clause 10.1.4
Processing at Receiver	According to clause 10.1.4
Information in Response message	According to clause 10.1.4
Processing at Originator after receiving Response	According to clause 10.1.4
Exceptions	According to clause 10.1.4

10.2.4.24 Delete <*timeSeries*>

This procedure shall be used for deleting a <*timeSeries*> resource residing under a <*timeSeries*> resource.

Table 10.2.4.24-1: <*timeSeries*> DELETE

<*timeSeries*> DELETE	
Information in Request message	All parameters defined in table 8.1.2-2 apply.
Processing at Originator before sending Request	According to clause 10.1.5.
Processing at Receiver	According to clause 10.1.5.
Information in Response message	According to clause 10.1.5.
Processing at Originator after receiving Response	According to clause 10.1.5.
Exceptions	According to clause 10.1.5.

10.2.4.25 Create <*timeSeriesInstance*>

This procedures shall be used for creating a <*timeSeriesInstance*> resource.

Table 10.2.4.25-1: <*timeSeriesInstance*> CREATE

<*timeSeriesInstance*> CREATE	
Information in Request message	All parameters defined in table 8.1.2-2 apply with the specific details for: ***Content:*** The resource content shall provide the information as defined in clause 9.6.37.
Processing at Originator before sending Request	According to clause 10.1.2.
Processing at Receiver	According to clause 10.1.2. If the newly created <*timeSeriesInstance*> resource violates any of the policies defined in the parent <*timeSeries*> resource (i.e. *maxInstanceAge,maxNrOfInstances* or *maxByteSize*), then the <*timeSeriesInstance*> resource with the oldest *dataGenerationTime* attribute shall be removed to enable the creation of the new <*timeSeriesInstance*> resource.
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content:*** Address of the created <*timeSeriesInstance*> resource, according to clause 10.1.2.
Processing at Originator after receiving Response	According to clause 10.1.2.
Exceptions	According to clause 10.1.2.

10.2.4.26 Retrieve <*timeSeriesInstance*>

This procedure shall be used for retrieving the attributes of a <*timeSeriesInstance*> resource.

Table 10.2.4.26-1: <*timeSeriesInstance*> RETRIEVE

<*timeSeriesInstance*> RETRIEVE	
Information in Request message	According to clause 10.1.3.
Processing at Originator before sending Request	According to clause 10.1.3.
Processing at Receiver	According to clause 10.1.3.
Information in Response message	According to clause 10.1.3.
Processing at Originator after receiving Response	According to clause 10.1.3.
Exceptions	According to clause 10.1.3.

10.2.4.27 Update <*timeSeriesInstance*>

The Update operation shall not apply to <*timeSeriesInstance*> resource.

10.2.4.28 Delete <*timeSeriesInstance*>

This procedure shall be used for deleting a <*timeSeriesInstance*> resource residing under a <*timeSeries*> resource.

Table 10.2.4.28-1: <*timeSeriesInstance*> DELETE

<*timeSeriesInstance*> DELETE	
Information in Request message	All parameters defined in table 8.1.2-2 apply.
Processing at Originator before sending Request	According to clause 10.1.5.
Processing at Receiver	According to clause 10.1.5.
Information in Response message	According to clause 10.1.5.
Processing at Originator after receiving Response	According to clause 10.1.5.
Exceptions	According to clause 10.1.5.

10.2.4.29 Procedure for Time Series Data Detecting and Reporting

In the case that the *periodicInterval* is set and the *missingDataDetect* is TRUE, the Hosting CSE shall monitor the Time Series Data based on its *periodicInterval*. When the Hosting CSE detects a missing data point, the *dataGenerationTime* of the missing data point is inserted into the *missingDataList* attribute and the *missingDataCurrentNr* shall be increased by one. When the *missingDataCurrentNr* reaches the *missingDataMaxNr*, the oldest *dataGenerationTime* shall be removed from *missingDataList* to enable the insertion of the new missing data point information.

When an AE wants to be informed of the number of missing data points in a given renewable time duration, the AE should request the creation of a <*subscription*> resource and set the *missingData* in the *eventNotificationCriteria* conditions to specify the reporting policy. This enables the AE to keep track of the number of missing data points and the corresponding time-stamps over a predefined but renewable duration (i.e. the "window duration" of the *missingData condition*).

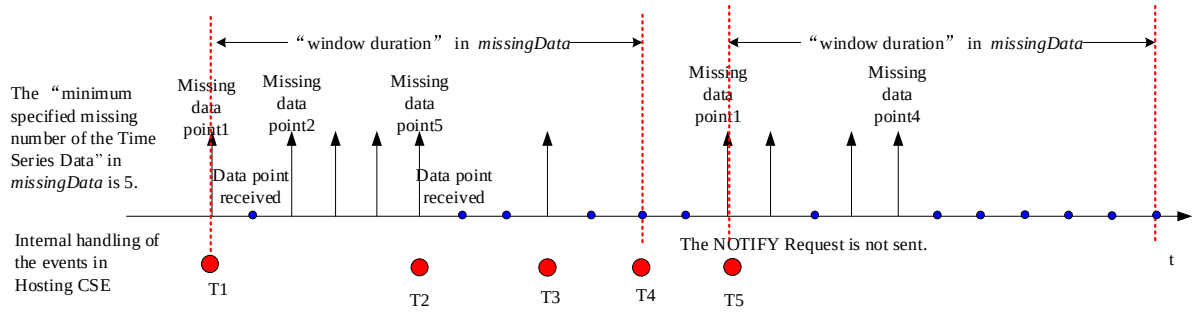
When the Hosting CSE reports missing data points, it shall check the *missingData* condition in the subscription resources created for that purpose.

When the first missing data point is detected (i.e. a detection of the first discontinuous time-stamp) following the creation of a subscription, the Hosting CSE shall start a timer associated with that subscription and start counting the number of missing data points. The timer is set according to the "window duration" in the subscription's *missingData* condition. The reporting policy is governed by the rules below:

- If the total number of missing data points becomes equal to the "minimum specified missing number of the Time Series Data" specified in the subscription's *missingData* condition before the timer expires, a NOTIFY request shall be sent including the "number of missing data points" that have been detected since the start of the subscription's timer. The missing data points counter shall continue counting while the timer continues to run (since it did not expire). A similar NOTIFY request shall be sent for each subsequent missing data point detected until the timer expires (see next bullet for behavior when the timer expires).

- If the timer expires, the missing data points counter is reset back to 0. The timer is restarted upon detection of next missing data.
- The reset of the timer and the missing data points counter upon timer expiry shall continue until such time as the subscription is cancelled or terminated.
- If no missing data points have been detected at all during the life time of a subscription, then no timer shall be started at all. But once a timer is started triggered by the first missing data point, then the above rules in the previous bullets shall apply.

Figure 10.2.4.29-1 depicts the above rules.



T1: When the first missing data point is detected the timer is started and the number of the missing data points is counted.
T2: The NOTIFY Request is sent when the total number of missing data points becomes equal to or greater than the value in the *missingData* attribute.
T3: The NOTIFY Request is sent again.
T4: At the end of the "window duration" the missing data points counter is reset back to 0.
T5: The "window duration" timer is restarted when the next missing data point is detected.

Figure 10.2.4.29-1: Time Series Data Detecting and Reporting Mechanism

10.2.5 Request message handling

10.2.5.1 Introduction

A request can be processed in blocking, non-blocking synchronous or non-blocking asynchronous mode as illustrated in clauses 8.2.1 and 8.2.2. Sometimes, more than one request can be batched together by an intermediate CSE and delivered to another CSE via the implicit creation of <*delivery*> resources.

For deployment scenarios where requests cannot be directly forwarded to a targeted AE or CSE (e.g. the AE or CSE is deployed behind NAT/Firewall), the targeted AE or CSE can perform long polling to retrieve requests from its Registrar CSE.

To support end-to-end secure communications, a request can be secured and encapsulated within a notify message to prevent tampering of the request.

10.2.5.2 Non-blocking communication management

Unlike blocking request handling, non-blocking request handling involves the implicit creation and use of <*request*> resources by a Receiver CSE to process the request. The following clauses illustrate <*request*> resource handling procedures.

10.2.5.3 Create <*request*>

As specified in clause 9.6.12, creation of a <*request*> resource shall only be done on a Receiver CSE implicitly when a Registree AE or a Registree/Registrar CSE of the Receiver CSE issues a request to the Receiver CSE for targeting any other resource type or requesting a notification in non-blocking mode. Therefore, the creation procedure of a <*request*> resource cannot be initiated explicitly by an Originator.

The specific conditions to create a <*request*> resource are as follows: When an AE or CSE issues a request for targeting any other resource type or requesting a notification in non-blocking mode , i.e. the ***Response Type*** parameter of the request is set to either '*nonBlockingRequestSynch'* or '*nonBlockingRequestAsynch'*, and the Receiver CSE supports the <*request*> resource type as indicated by the *supportedResourceType* attribute of the <*CSEBase*> resource representing the Receiver CSE.

The Receiver CSE of a non-blocking Request that was issued by either:

- a Registrar AE of the Receiver CSE; or
- a Registree/Registrar CSE of the Receiver CSE;

is the Hosing CSE for the <*request*> resource that shall be associated with the non-blocking request.

The Hosting CSE shall follow the procedure outlined in this clause.

Step 001: The Receiver shall:

1) Assign values to the *resourceID* and *resourceName* attributes of the <*request*> resource to be created.
2) Assign a value to the following common attributes specified in clause 9.6.1.3:
   a) *parentID*;
   b) *creationTime*;
   c) *expirationTime*: The Receiver shall assign a value that is consistent with the ***Request Expiration Timestamp***, ***Result Expiration Timestamp*** and ***Result Persistence*** parameters effective for the associated non-blocking request that implied the creation of this <request> resource (within the restriction of the Receiver policies). If a value consistent with the ***Request Expiration Timestamp***, ***Result Expiration* Timestamp** and ***Result Persistence*** parameters effective for the associated non-blocking request that implied the creation of this <request> resource cannot be supported, due to either policy or subscription restrictions, the Receiver will assign a new value;
   d) *lastModifiedTime*: which is equals to the creationTime;
   e) *stateTag*;
   f) *accessControlPolicyIDs*: Populate with the resource identifier of an <accessControlPolicy> that contains the following:
      i) In the *privileges* attribute:
         1) Allow RETRIEVE, UPDATE and DELETE operations for the Hosting CSE.
         2) Allow RETRIEVE and DELETE operations for the Originator, i.e. the value of the ***From*** parameter.
      ii) In the *selfPrivileges* attribute:
         1) Allow UPDATE operations for the Originator, i.e. the value of the ***From*** parameter.
3) Assign any other RO (Read Only) attributes of <*request*> resource type within the restriction of the Receiver policies:
   a) Operation: Value of the parameter ***Operation*** in the associated non-blocking request that implied the creation of this <*request*> resource;
   b) Target: Value of the parameter ***To*** in the associated non-blocking request that implied the creation of this <*request*> resource;
   c) Originator: Value of the parameter ***From*** in the associated non-blocking request that implied the creation of this <*request*> resource;
   d) *requestIdentifier*: Value of the parameter ***Request Identifier*** in the associated non-blocking request that implied the creation of this <*request*> resource;

e) *metaInformation*: The content of this attribute is set to information in any other optional parameters described in clause 8.1. given in the associated non-blocking request that implied the creation of this <*request*> resource;

f) content: Value of the parameter ***Content*** - if any - in the associated non-blocking request that implied the creation of this <*request*> resource;

g) *requestStatus*: Information on the initial status of the associated non-blocking request that implied the creation of this <*request*> resource. The initial value of this attribute shall be identical to the status that is contained in the Acknowledgement response message of the associated non-blocking request. Possible values for status information contained in this attribute are specified in ETSI TS 118 104 [3]. The value of this attribute is subject to changes according to the progress in processing of the non-blocking request that implied the creation of this <*request*> resource;

h) *operationResult*: Initially empty. This attribute will be used for representing the result of the originally requested operation - if any - in line with the ***Result Content*** parameter in the associated non-blocking request that implied the creation of this <*request*> resource.

Step 002: The Receiver shall create the <*request*> resource.

Table 10.2.5.3-1: *<request>* CREATE

***<request>* CREATE**	
Information in Request message	Not applicable. For *<request>* resources, explicit creation via a Request message shall not be supported
Pre-Processing at Originator	Not applicable. There is no Originator. *<request>* resources are only created implicitly
Processing at Receiver	Different to the non-registration CREATE procedure described in clause 10.1.2, see outlined steps described in the present clause above
Information in Response message	Not applicable. Since *<request>* resources shall not be created explicitly, no response messages will be sent after creation. However, the address of a *<request>* resource will be passed back as a reference to the Originator of an associated non-blocking request that implied the creation of this *<request>* resource
Post-Processing at Originator	None
Exceptions	None

10.2.5.4 Retrieve *<request>*

This procedure shall be used for retrieving the attributes of a <*request*> resource.

Table 10.2.5.4-1: *<request>* RETRIEVE

***<request>* RETRIEVE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content*****:** Void
Pre-Processing at Originator	According to clause 10.1.3 with the following specific processing: Originator needs to retrieve information about an associated previously issued non-blocking request
Processing at Receiver	According to clause 10.1.3 with the following specific processing: The Receiver shall provide the content of the addressed *<request>* resource or the addressed attributes thereof
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content*****:** Attributes of the *<request>* resource as defined in clause 9.6.12
Post-Processing at Originator	According to clause 10.1.3
Exceptions	According to clause 10.1.3 According to clause 10.1.2 with the following: • The Originator CSE is not authorized to retrieve the *<request>* resource or the addressed parts of it • The addressed *<request>* resource does not exist

10.2.5.5 Update *<request>*

For a *<request>* resource explicit update requests shall not be supported. Changes in the attributes of a *<request>* resource can only be done by the Hosting CSE due to changes of the status of the associated non-blocking request that implied the creation of this *<request>* resource or due to reception of an operation result in response to the associated non-blocking request that implied the creation of this *<request>* resource.

10.2.5.6 Delete *<request>*

This procedure shall be used for deleting an existing *<request>* resource. Deletion of an existing *<request>* resource shall terminate any further processing of an associated pending non-blocking request that implied the creation of this *<request>* resource if the pending request was not already completed or forwarded.

Table 10.2.5.6-1: *<request>* DELETE

<request> DELETE	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Pre-Processing at Originator	According to clause 10.1.5 with the following: Originator needs to cancel a previously issued non-blocking request that is still pending, i.e. it has not yet been completed or Originator needs to remove the *<request>* resource representing the context of an already completed non-blocking request
Processing at Receiver	According to clause 10.1.5: • Receiver CSE checks if the associated non-blocking request process is still pending. If so, it stops that request process • Receiver CSE removes the addressed *<request>* resource
Information in Response message	According to clause 10.1.5 with the following specific information: Successful Response message indicating that the associated non-blocking request process was stopped as requested or the context of an already completed associated non-blocking request was deleted
Post-Processing at Originator	According to clause 10.1.5
Exceptions	According to clause 10.1.5 with the following: • The Originator CSE is not authorized to delete the *<request>* resource • The addressed *<request>* resource does not exist

10.2.5.7 Request delivery aggregation

In this introduction an example for delivering information from a source CSE to a target CSE via the use of the <delivery> resource is explained.

The information flow depicted in figure 10.2.5.7-1 defines the exchange of Requests/Responses for processing an original request targeting a resource that is not hosted on the Registrar CSE of the request Originator. The following assumptions hold:

- Originator is AE1.
- AE1 is registered with CSE1, i.e. CSE1 is the Registrar CSE for AE1.
- The original Request is an UPDATE to a remote resource hosted on CSE3, i.e. CSE3 is the Hosting CSE for the target resource.
- UPDATE options in the original Request are selected such that no feedback after completion of the update operation was requested, i.e. AE1 decided that it does not need to hear back from CSE3; this is expressed by setting the ***Result Content*** information to "nothing", see clause 8.1.2.
- Delivery related parameters included in the original UPDATE request (may be set via CMDH policies): ***Request Expiration Timestamp***, ***Event Category***, ***Delivery Aggregation*** and ***Result Persistence***:
  - ***Request Expiration Timestamp*** indicates how long the forwarding of the request can last.
  - ***Event Category*** indicates the event category that should be used by CMDH to handle this request.

- ***Result Persistence*** indicates how long after the request has expired, the local request context should still be available for retrieving status or result information.
- ***Delivery Aggregation*** would be set to ON indicating that <*delivery*> resource shall be used for forwarding the request.

- CSE1 is the CSE of an Application Service Node.
- CSE1 is registered with CSE2 and interacts with CSE2 via the reference point Mcc(1).
- CSE2 is the CSE of a Middle Node.
- CSE2 is registered with CSE3 and interacts with CSE3 via the reference point Mcc(2)
- CSE3 is the CSE of an Infrastructure Node.

The Originator AE1 shall get a confirmation from CSE1 when the original Request is accepted. The response informs AE1 that CSE1 has accepted the Request and has accepted responsibility to execute on the requested operation. Furthermore, AE1 has expressed by setting ***Result Content*** to "nothing" that no result of the requested operation is expected to come back from CSE3. With the provided reference (Req-Ref in figure 10.2.5.7-1. AE1 can retrieve the status of the issued request at a later time, for instance to find out if the request was already forwarded to CSE2 or if it is still waiting for being forwarded on CSE1. Before accepting the request from AE1, CSE1 has also verified if the delivery related parameters expressed by AE1 (settings of ***Request Expiration Timestamp*** and ***Event Category***) are in line with provisioned CMDH policies. AE1 may not be authorized to use certain values for ***Request Expiration Timestamp*** or ***Event Category***.

In line with the delivery related parameters, CSE1 is generating a local <*delivery*> resource on CSE1 and attempts to forward the content of it in line with provisioned CMDH policies at a suitable time and via a suitable connection to CSE2 by requesting the creation of a <*delivery*> resource on CSE2. In this example case, the *lifespan* attribute of this delivery resource is set to the same value as the ***Request Expiration Timestamp*** parameter expressed by AE1. In general - i.e. also in cases where more than one original request is aggregated into a single create request for a <*delivery*> resource - the *lifespan* and *eventCat* attributes of the created <*delivery*> resource shall be set consistent with the ***Request Expiration Timestamp*** and ***Event Category*** parameters in the set of original requests. See the attribute definitions in clause 9.6.11.

CSE1 shall use a blocking request for requesting creation of a <*delivery*> resource on CSE2.

When CSE2 has accepted the incoming request from CSE1, CSE1 may delete the *aggregatedRequest* attribute of the local <*delivery*> resource. Furthermore - if the expiration time of the local <*delivery*> resource is not exhausted - the Registrar CSE shall update *deliveryMetaData* attribute of the local <*delivery*> resource to indicate that it has been forwarded to CSE2.

When CSE2 has accepted the request to create a local <*delivery*> resource, it shall attempt to forward it to CSE3. In line with the delivery related parameters, CSE2 shall create a local <*delivery*> resource on CSE2 and shall attempt to forward it in line with provisioned CMDH policies at a suitable time and via a suitable connection to CSE3 by requesting the creation of a <*delivery*> resource on CSE3.

CSE2 shall use a blocking request for requesting creation of a <*delivery*> resource on CSE3.

When CSE3 has accepted the incoming request from CSE2, CSE2 may delete the *aggregatedRequest*attribute of the local <*delivery*> resource. Furthermore - if the expiration time of the local <*delivery*> resource is not exhausted - the Registrar CSE shall update *deliveryMetaData* attribute of the local <*delivery*> resource to indicate that it has been forwarded to CSE3.

When CSE3 has accepted the request to create a local <*delivery*> resource, it shall determine that the target of the delivery was CSE3 itself. Therefore it shall forward internally the original request contained in the *aggregatedRequest* attribute of the <*delivery*> resource.

Within CSE3, functions that are responsible for checking and executing local access to resources in CSE3 will execute the originally requested UPDATE operation. If successful, the targeted resource will be updated with the content provided by the Originator.

Since in the depicted case no result needed to be sent back to the Originator, the processing for the requested operation is then completed.

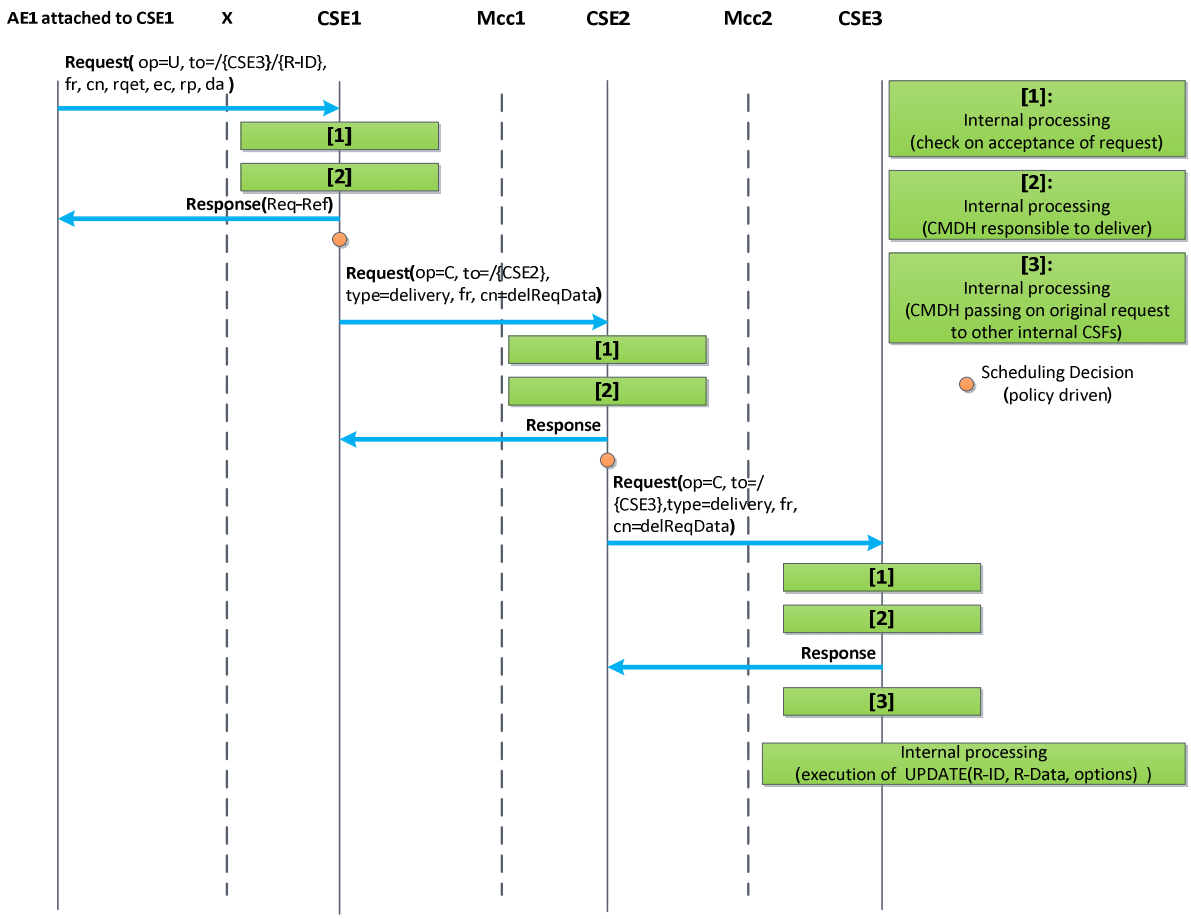


Figure 10.2.5.7-1: CMDH information flow for 2 hops - no result needs to be returned after operation completes

The following procedures shall be triggered by requesting the corresponding operations on a <*delivery*> resource:

- Initiate the delivery of one or more original request(s) stored for later forwarding from one CSE to another CSE:
  - Request a CREATE operation for a <*delivery*> resource from an issuing CSE to a receiving CSE.
  - The original request(s) need to be contained in the "*aggregatedRequest*" attribute of the <*delivery*> resource.
  - If successful, the receiving CSE takes the responsibility to further execute on the delivery process for the original Request.
  - If not successful, the issuing CSE cannot assume that the receiving CSE will carry out the delivery of the original request.
- Get information about the status of a pending delivery process for an original request:
  - Request a RETRIEVE operation of the content of a <*delivery*> resource representing a pending delivery or part of it.
  - The status of the pending forwarding process is reflected the "*deliveryMetaData*" attribute defined in the <*delivery*> resource.
- Change parameters of pending delivery process:
  - Request an UPDATE operation on applicable attributes of the <*delivery*> resource representing the pending delivery.

- For instance the time allowed for completion of a delivery process could be modified by updating the "*lifespan*" attribute of an existing <*delivery*> resource.

- Cancel a pending delivery request:
  - Request a DELETE operation of a <*delivery*> resource that represents a pending delivery process.

10.2.5.8 Create <*delivery*>

This procedure shall be used for requesting a CSE to take responsibility to deliver the provided data to a target CSE in line with CMDH parameters and provisioned CMDH policies in case <*delivery*> resource based CMDH processing is used. If indicated by the Originator, the Receiver shall confirm the acceptance of delivery responsibility by a successful Response.

Originator: The Originator of a Create request for a <*delivery*> resource can only be a CSE. The Originator needs to provide the content of a <*delivery*> resource type together with the Create request or can Update it after a successful creation of the <*delivery*> resource with empty *aggregatedRequest* attribute. Otherwise the Receiver cannot accept the Create Request. The Originator shall use a blocking request for issuing the Create request to the Receiver.

Receiver: The receiver of a Create request for a <*delivery*> resource is a Registrar or Registree CSE of the Originator and it shall check the access control policies to assure the Originator is authorized to request a delivery procedure. The Receiver of the Create Request shall further check whether the provided attributes of the <*delivery*> resource that is requested to be created represents a valid request for forwarding data to a target CSE. If the Originator of the Create request is authorized and the Request is valid, the Receiver shall check whether it can actually satisfy the requested delivery in line with provisioned CMDH policies and requested *eventCat* and *lifespan* attributes of the <*delivery*> resource. If all these checks are positive, the Receiver shall create the requested <*delivery*> resource and assumes responsibility for delivering the requested data to the target CSE as soon as the content of the *aggregatedRequest* attribute is available. In case an operation result is expected by the Originator, the Receiver shall confirm acceptance of the responsibility by indicating a successful creation of the <*delivery*> resource. If the Receiver CSE is the target CSE of the requested delivery, it shall forward the content of the delivered data - which represents one or more forwarded original request(s) - to the internal functions that handle incoming requests and continue processing of the forwarded request(s).

Table 10.2.5.8-1: <*delivery*> CREATE

<*delivery*> CREATE	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***From:*** CSE only ***Content:*** The resource content shall provide the information as defined in clause 9.6.11 ***Response Type:*** Shall be set to "blockingRequest" which means a blocking request is issued
Processing at Originator before sending Request	According to clause 10.1.2 with the following specific processing: The Originator needs to provide the content of a <*delivery*> resource type together with the Create request or can Update it after a successful creation of the <*delivery*> resource with empty *aggregatedRequest*attribute. Otherwise the Receiver cannot accept the Create Request. The Originator shall use a blocking request for issuing the Create request to the Receiver
Processing at Receiver	According to clause 10.1.2 with the following specific processing: • Check whether the provided attributes of the <*delivery*> resource that is requested to be created represents a valid request for delivering data to a target CSE • Check whether Receiver CSE can actually satisfy the requested delivery in line with provisioned policies and requested delivery parameters • If all checks are positive, the receiver shall create the requested <*delivery*> resource and assumes responsibility for delivering the provided data to the target CSE • If the Receiver CSE is the target CSE of the requested delivery, it shall forward the content of the delivered data to the internal CSFs that will interpret the delivered data as a forwarded request(s) from a remote Originator

***<delivery>* CREATE**	
Information in Response message	All parameters defined in table 8.1.3-1 apply, with the following specific information: In case the Originator CSE has not asked for a Result of the requested Operation (***Result Content*** set to "nothing"), the Response only contains an Acknowledgement indicator. This only indicates that the Receiver CSE received the Request. It does NOT indicate whether the Receiver CSE was able to take on responsibility for delivery of the data In case the Originator CSE asked for the status of the requested Operation to be contained in the Result of the requested Operation (***Result Content*** not set to "nothing"), the Receiver CSE shall respond with a Success or Failure indicator In case the Originator CSE asked for the status of the requested Operation and the address of the created Resource to be contained in the Result of the Request, the Receiver CSE shall respond with a Success indicator including the address of the created <*delivery*> resource in case it has taken on responsibility to deliver the data to the target CSE or with Failure indicator including an error indication otherwise
Processing at Originator after receiving Response	According to clause 10.1.2 with the following specific processing: The Originator CSE shall update the local <*delivery*> resource to reflect the new status of the delivery process (e.g. '{Receiver-CSE-ID} accepted delivery responsibility') In case the Originator CSE got a Success indicator as a Response, it shall stop any further delivery attempts. In that case or if there was no indication of a need to provide a result of the operation, the Originator CSE may delete the content of the *'aggregatedRequest'* attribute of the local <*delivery*> resource In case the Originator CSE got a Failure indicator as a response, it may initiate further delivery attempts in line with CMDH policies and delivery parameters and depending on the reason for Failure In case the Receiver CSE is the target CSE of the delivery, the Receiver CSE needs to execute on the forwarded request contained in the delivered data
Exceptions	According to clause 10.1.2 with the following: • The Originator CSE is not authorized to request a delivery procedure on the Receiver CSE • The provided content of the <*delivery*> resource is not in line with the specified structure • The provided content of the <*delivery*> resource represents a request for delivery that is not consistent (e.g. lifespan attribute already expired) • The provided content of the <*delivery*> resource represents a request for delivery that cannot be met by the Receiver CSE within the limits of the provided delivery parameters and the provisioned CMDH policies on the Receiver CSE

10.2.5.9 Retrieve <*delivery*>

This procedure shall be used for requesting a CSE to provide information on a previously created <*delivery*> resource which represents delivery of data to a target CSE.

Table 10.2.5.9-1: *<delivery>* RETRIEVE

***<delivery>* RETRIEVE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content:*** void
Processing at Originator before sending Request	According to clause 10.1.3 with the following specific processing: Originator needs to retrieve information about a previously issued delivery
Processing at Receiver	According to clause 10.1.3 with the following specific processing: The Receiver shall provide the content of the addressed <*delivery*> resource or the addressed attributes thereof
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content:*** attributes of the <*delivery*> resource as defined in clause 9.6.11
Processing at Originator after receiving Response	According to clause 10.1.3
Exceptions	According to clause 10.1.3 with the following: • The Originator CSE is not authorized to retrieve the <*delivery*> resource or the addressed parts of it

	• The addressed <*delivery*> resource does not exist

10.2.5.10 Update <*delivery*>

This procedure shall be used for requesting a CSE to update information on a previously created <*delivery*> resource which represents a pending delivery of data to a target CSE. The update may have impact on further processing of the delivery.

Table 10.2.5.10-1: <*delivery*> UPDATE

<*delivery*> UPDATE	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: • Address of the <*delivery*> resource • Content of a <*delivery*> resource in line with the definition in clause 9.6.11 representing a valid request for delivery of data to a target CSE
Processing at Originator before sending Request	According to clause 10.1.4 with the following specific processing: • Originator needs to modify information about a previously issued delivery that is still pending, i.e. it has not yet been forwarded to another CSE
Processing at Receiver	According to clause 10.1.4 with the following specific processing: • Receiver CSE checks if the requested changes to the delivery process can actually be accomplished • If possible, the Receiver CSE modifies the previously established delivery process and changes the respective content of the <*delivery*> resource
Information in Response message	According to clause 10.1.4
Processing at Originator after receiving Response	According to clause 10.1.4
Exceptions	According to clause 10.1.4 with the following: • The Originator CSE is not authorized to modify the <*delivery*> resource or the addressed parts of it • The addressed <*delivery*> resource does not exist • The responsibility for the further processing of the delivery process represented by the addressed <*delivery*> process was already forwarded to another CSE

10.2.5.11 Delete <*delivery*>

This procedure shall be used for requesting a CSE to cancel a pending delivery of data to a target CSE or to delete the <*delivery*> resource of an already executed delivery.

Table 10.2.5.11-1: *<delivery>* DELETE

***<delivery>* DELETE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.5 with the following: • Originator needs to cancel a previously issued delivery that is still pending, i.e. it has not yet been forwarded to another CSE or Originator needs to remove the *<delivery>* resource representing an already executed delivery
Processing at Receiver	According to clause 10.1.5: • Receiver CSE checks if the corresponding delivery process is still pending. If so, it stops that delivery process • Receiver CSE removes the addressed *<delivery>* resource and stop the corresponding delivery process if it is still pending
Information in Response message	According to clause 10.1.5 with the following specific information: • Successful Response messages indicate that the delivery process was stopped as requested
Processing at Originator after receiving Response	According to clause 10.1.5
Exceptions	According to clause 10.1.5 with the following: • The Originator CSE is not authorized to delete the *<delivery>* • The addressed *<delivery>* resource does not exist

10.2.5.12 Request message polling

An AE or a CSE that is request unreachable cannot receive a request from other entities directly. Instead this AE/CSE can retrieve requests that others sent to this AE/CSE once it created *<pollingChannel>* resource on a request reachable CSE.

This clause consists of manipulation procedures of *<pollingChannel>* resource (clauses 10.2.5.13 to 10.2.5.16), re-targeting request to *<pollingChannel>* resource (clause 10.2.5.17), the long polling procedure to retrieve requests from *<pollingChannel>* resource (clause 10.2.5.18) and the responding to the request received by long polling (clause 10.2.5.19). This is depicted in figure 10.2.5.12-1.

Figure 10.2.5.12-1 depicts the case when the Originator sent a request("req2") to the Target as a blocking request. The request can be any of the requests defined in clause 10.2 (e.g. <container> resource creation on the Target CSE). As defined in clause 10.2.5.18, polling response contains the "req2" in step 0004. Also as per clause 10.2.5.19, in step 005 the "req3" contains the "resp2", which is the response to the "req2" in step 002 and step 004, in the "req3". Finally the "resp2" is forwarded to the Originator in step 006.

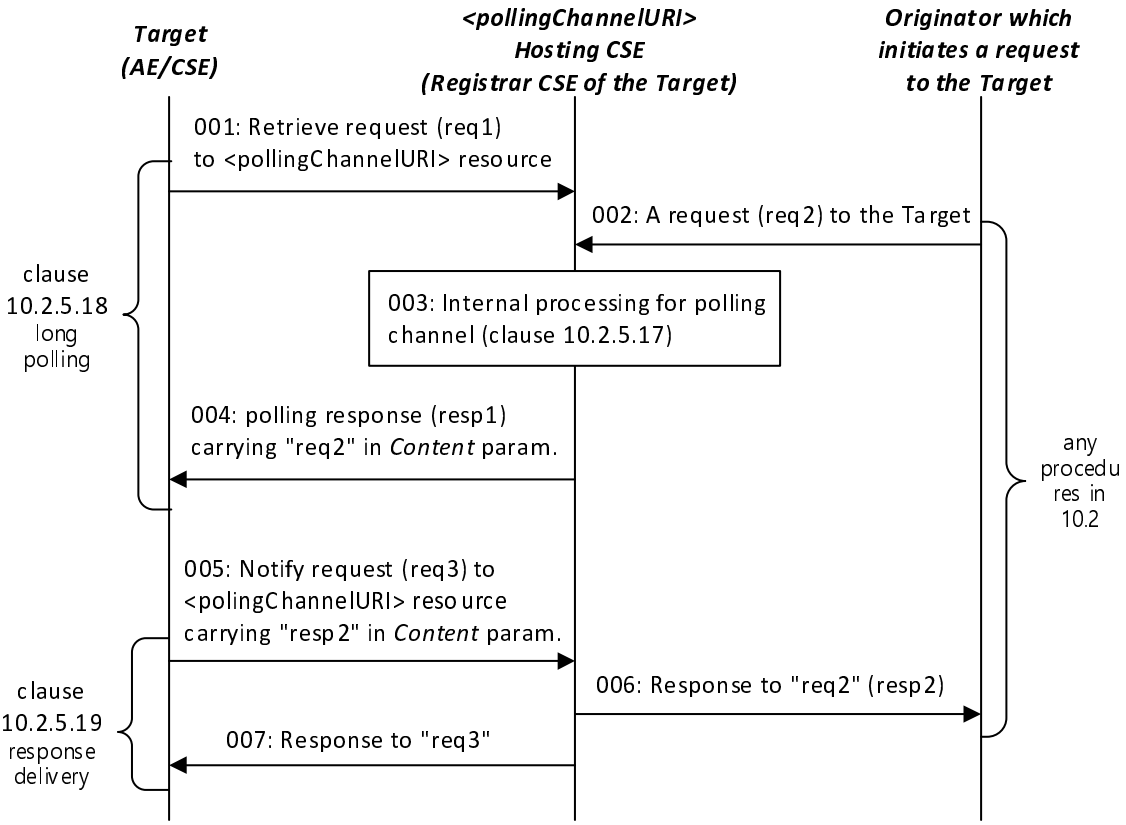


Figure 10.2.5.12-1: Request/response delivery via polling channel

10.2.5.13 Create <*pollingChannel*>

Table 10.2.5.13-1: <*pollingChannel*> CREATE

<*pollingChannel*> CREATE	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***To:*** Address of <*AE*> or <*remoteCSE*> resource ***Content:*** attributes of the <*pollingChannel*> resource as defined in clause 9.6.21
Processing at Originator before sending Request	According to clause 10.1.2 with the following additions: • If an AE is the Originator, it shall address the <*AE*> resource that it already created. Otherwise, if a CSE is the Originator, it shall address the <*remoteCSE*> resource that it already created
Processing at Receiver	According to clause 10.1.2 with the replacement for sub-step 1) of step 002 as follows: • The Hosting CSE shall check if the Originator ID is the same as the CSE-ID or AE-ID of the parent resource which is the <remoteCSE> or <AE> resource If the check fails, the request shall be rejected
Information in Response message	According to clause 10.1.2
Processing at Originator after receiving Response	The Originator should send a retrieve request to the <pollingChannelURI> resource
Exceptions	According to clause 10.1.2

10.2.5.14 Retrieve <*pollingChannel*>

This procedure is used to retrieve a <*pollingChannel*> resource and an AE/CSE can be an Originator.

Table 10.2.5.14-1: *<pollingChannel>* RETRIEVE

***<pollingChannel>* RETRIEVE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content:*** void
Processing at Originator before sending Request	According to clause 10.1.3
Processing at Receiver	According to clause 10.1.3 with the following for step 002: • For access privilege checking, the Hosting CSE shall check if the Originator ID is the same as the CSE-ID or AE-ID of the parent resource which is the <remoteCSE> or <AE> resource, respectively. If the check fails, the request shall be rejected
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content:*** attributes of the *<pollingChannel>* resource as defined in clause 9.6.21
Processing at Originator after receiving Response	According to clause 10.1.3
Exceptions	According to clause 10.1.3

10.2.5.15 Update *<pollingChannel>*

This procedure is used to update a *<pollingChannel>* resource and an AE/CSE can be an Originator.

Table 10.2.5.15-1: *<pollingChannel>* UPDATE

***<pollingChannel>* UPDATE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content:*** attributes of the *<pollingChannel>* resource as defined in clause 9.6.21
Processing at Originator before sending Request	According to clause 10.1.4
Processing at Receiver	According to clause 10.1.4 with the following for step 002: • For access privilege checking, the Hosting CSE shall check if the Originator ID is the same as the CSE-ID or AE-ID of the parent resource which is the <remoteCSE> or <AE> resource, respectively. If the check fails, the request shall be rejected
Information in Response message	According to clause 10.1.4
Processing at Originator after receiving Response	According to clause 10.1.4
Exceptions	According to clause 10.1.4

10.2.5.16 Delete *<pollingChannel>*

This procedure is used to delete a *<pollingChannel>* resource and an AE/CSE can be an Originator.

Table 10.2.5.16-1: *<pollingChannel>* DELETE

***<pollingChannel>* DELETE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.5
Processing at Receiver	According to clause 10.1.5 for step 002: • For access privilege checking, the Hosting CSE shall check if the Originator ID is the same as the CSE-ID or AE-ID of the parent resource which is the <remoteCSE> or <AE> resource, respectively. If the check fails, the request shall be rejected
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	According to clause 10.1.5
Exceptions	According to clause 10.1.5

10.2.5.17 Internal Processing for Polling Channel

This procedure is used to forward a request to a request-unreachable AE or CSE(i.e. *requestReachability* attribute of its <AE> or <remoteCSE> resource is set to FALSE) which has created a <pollingChannel> resource as a child of its <AE> or <remoteCSE> resource. When a <pollingChannel> Hosting CSE receives a request towards the AE or CSE, it shall forward the request to the AE or CSE in the ***Content*** parameter of the response to polling response (see clause 10.2.5.18). If there is no pending polling request from the AE or CSE, then the <pollingChannel> Hosting CSE shall store the request and forward it when it receives the polling request. When the stored request expires according to its ***Request Expiration Timestamp*** parameter the Hosting CSE shall return an error to the entity that initiated the request.

10.2.5.18 Long Polling on Polling Channel

This procedure is originated by a request-unreachable entity to poll requests from a polling channel. Once the Originator starts long polling on a polling channel by sending a RETRIEVE request, the Receiver who is the *<pollingChannel>* Hosting CSE holds the request until it has any requests to return to the Originator. If the request expires and there is no available request to return, the Receiver shall send the response with a status indicating a timeout has occurred to inform the Originator that a new polling request should be generated again.

Table 10.2.5.18-1: *<pollingChannelURI>*RETRIEVE

Long Polling RETRIEVE	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***To:*** Address of *<pollingChannelURI>* child resource of the *<pollingChannel>* resource
Processing at Originator before sending Request	According to clause 10.1.3
Processing at Receiver	According to clause 10.1.3 with the following privilege check for step 002: • The Hosting CSE shall check if the Originator ID is the same as the CSE-ID or AE-ID of the grant parent *<remoteCSE>* or *<AE>* resource, respectively • The Hosting CSE shall check if there is any request to be returned to the Originator. If there is any, the Hosting CSE shall generate the response containing the request(s) for the Originator. If none, the Hosting CSE shall wait for any request for the Originator to be reached at the polling channel until the request expiration time
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content:*** request message(s) targeting the Originator
Processing at Originator after receiving Response	If the Originator receives the response from the Receiver that the long polling request is expired, the Originator should send a new long polling request
Exceptions	If the long polling request is expired at the Receiver, the Receiver shall send an unsuccessful response to the Originator

10.2.5.19 Delivering the response to the request sent over polling channel

When a Registree AE or CSE receives a response from a long polling request (clause 10.2.5.18), the Registree AE or CSE shall generate a response to each request primitive contained in the ***Content*** of the long polling response. The Registree AE or CSE shall send each of the responses to the received requests in in the ***Content*** parameter of a new Notify request to the <pollingChannelURI> Hosting CSE (one Notify request for each primitive received in the long polling response).

When the Hosting CSE receives a Notify request targeting the <pollingChannelURI> resource (Figure 10.2.5.12-1, req3), the Hosting CSE shall send the response, contained in the ***Content*** parameter of the Notify request, to the entity that sent the associated request (Figure 10.2.5.12-1, req2) to the Hosting CSE. The associated request is the request that the Hosting CSE received (Figure 10.2.5.12-1, step 002) and forwarded to the Registree AE or CSE (Figure 10.2.5.12-1, step 004) over the polling channel. The association shall be done by matching the ***Request Identifier*** parameter of the request delivered in <pollingChannelURI> Retrieve response and the ***Request Identifier*** parameter of the response delivered in the ***Content*** parameter in a <pollingChannelURI> Notify request.

Table 10.2.5.19-1: *<pollingChannelURI>* NOTIFY

***<pollingChannelURI>* NOTIFY**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***To:*** Address of *<pollingChannelURI>* resource ***Content:*** The response to the request contained in <pollingChannelURI> Retrieve response
Processing at Originator before sending Request	Originator shall handle and generate the response to each of the requests contained in the <pollingChannelURI> Retrieve response
Processing at Receiver	The Hosting CSE shall send the response contained in the Content parameter of Notify request to the entity that sent the associated request to the Hosting CSE
Information in Response message	All parameters defined in table 8.1.3-1 apply
Processing at Originator after receiving Response	According to clause 10.1.6
Exceptions	If the Originator of the <pollingChannelURI> Notify is not the AE-ID of the <AE> resource or CSE-ID of the <remoteCSE> resource which is the grandparent resource of the <pollingChannelURI> resource, then the Hosting CSE shall reject the request with access privilege error information

10.2.5.20 End-to-end secure communication

Multi-hop end-to-end secure communication is supported by encapsulating an end-to-end secured request within a notify message. Depending on the Receiver entity, a notification target can be an <AE> or <CSEBase> resource.

10.2.5.21 End-to-AE communication

This procedure shall be used for sending a Notify request to an *<AE>* resource for the case when the AE represented as the *<AE>* resource is the final target of the request contained in the notification. When a notification is received at the <AE> resource Hosting CSE, the Hosting CSE shall perform a notify re-targeting (clause 10.2.5.23). In this description, the Receiver is the CSE hosting the <AE> resource.

Table 10.2.5.21-1: *<AE>* NOTIFY

***<AE>* NOTIFY**	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.6
Processing at Receiver	The Hosting CSE shall re-target the Notify request to the AE according to clause 10.2.5.23
Information in Response message	According to clause 10.1.6
Processing at Originator after receiving Response	According to clause 10.1.6
Exceptions	According to clause 10.1.6

10.2.5.22 End-to-CSE communication

This procedure shall be used for sending a Notify request to a <*CSEBase*> resource for the case when the CSE represented as the <*CSEBase*> resource is the final target of the request contained in the notification. In this description, the Receiver is the Hosting CSE of the <CSEBase> resource.

Table 10.2.5.22-1: <*CSEBase*> NOTIFY

<*CSEBase*> NOTIFY	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.6
Processing at Receiver	According to clause 10.1.6
Information in Response message	According to clause 10.1.6
Processing at Originator after receiving Response	According to clause 10.1.6
Exceptions	According to clause 10.1.6

10.2.5.23 Notification Re-targeting

A Notify request to an AE is sent by targeting <*AE*> resource on a Hosting CSE. If the Hosting CSE verifies access control privilege of the Originator, the Hosting CSE shall re-target the request to the address specified as AE-PoA (i.e. pointOfAccess attribute of <*AE*> resource). The AE-PoA may be initially configured in the <*AE*> resource when the AE registers to the Registrar CSE. If the <*AE*> resource does not contain an AE-PoA, an active communication link, if available, can be used for re-targeting. If neither of them is available, the request cannot be re-targeted to the AE.

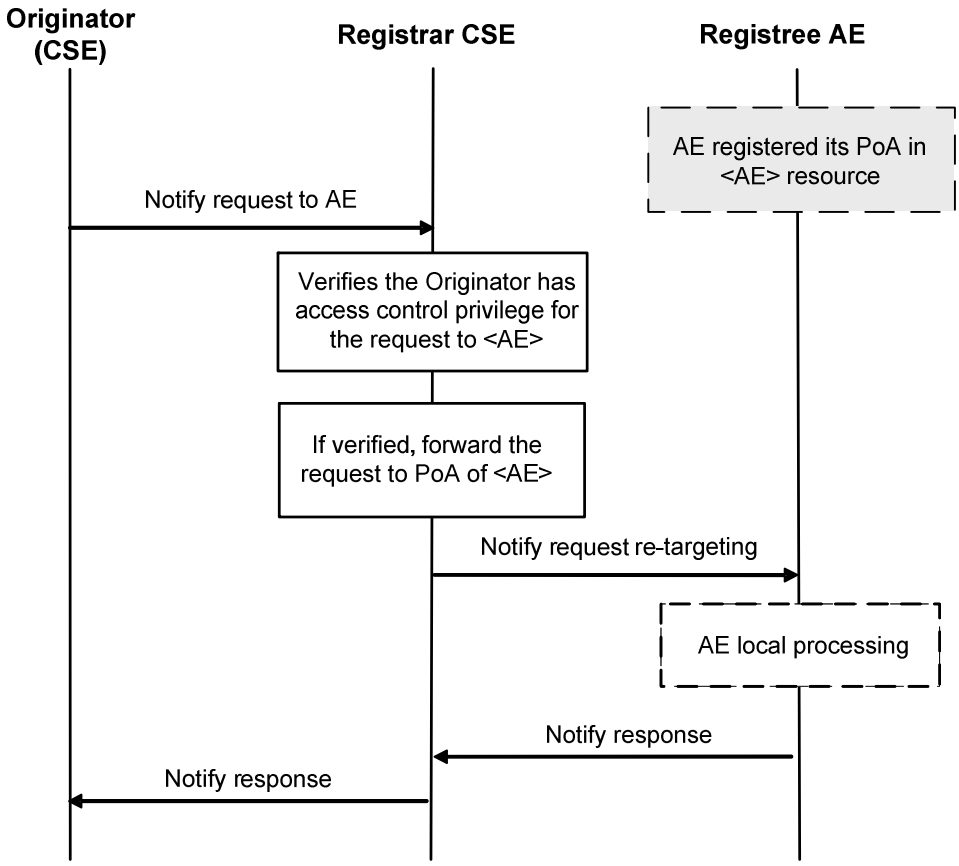


Figure 10.2.5.23-1: Re-targeting a notification request to an AE

10.2.6 Discovery

10.2.6.1 Discovery without Result Content parameter

This is the resource discovery procedure which returns matching resource identifiers. Note that the returned information is the difference compared to the other discovery mechanism in the present document which involves the ***Result Content*** parameter (clause 10.2.6.2).

The resource discovery procedures allow discovering of resources residing on a CSE. The use of the ***Filter Criteria*** parameter allows limiting the scope of the results.

Resource discovery shall be accomplished using the RETRIEVE method by an Originator which shall also include the root of where the discovery begins: e.g. <*CSEBase*>. The unfiltered result of the resource discovery procedure includes all the child resources under the root of where the discovery begins, which the Originator has a Discover access right on. The unfiltered results do not include any resources whose status is marked as "INACTIVE", as well as any child resources of these "INACTIVE" resources. For the allowed *Result Content* parameter options for Discovery related RETRIEVE see clause 8.1.2.

Filter criteria conditions may be provided as parameters to the RETRIEVE method. The filter criteria conditions describe the rules for resource discovery, e.g. resource types, creation time and matching string. The filter criteria can also contain the parameters for specifying the maximum number of discovered resources included in the response, the maximum limit on the number of levels in the resource tree (starting from the target resource) that the Hosting CSE shall perform the discovery request upon and an offset for specifying the number of discovered resources the Hosting CSE shall skip over and not include within the response. Table 8.1.2-2 describes the ***Filter Criteria*** parameter.

A match shall happen when a resource matches the configured filter criteria conditions and the Originator has a Discover access right on the resource. A discovery operation shall skip over resources that cannot be discovered due to access rights but shall include their descendants, if the Originator has discovered rights on them. A successful response contains a list for the matched resources addressable in any of the forms expressed in clause 9.3.1 if matches are found. If no matches are found, a successful response returns no matched resources. If ***Desired Identifier Result Type*** parameter is specified in a discovery request, the Hosting CSE shall choose the resource identifier format specified by the ***Desired Identifier Result Type*** parameter.

The discovery results may be modified by the Hosting CSE to restrict the scope of discoverable resources according to the Originator's access control policy or M2M service subscription.

The Hosting CSE may also implement a configured upper limit on the size of the answer. In such a case when the Originator and the Hosting CSE have different upper limits, the smaller of the two shall apply.

The Hosting CSE shall return the discovery results as a list of resource identifiers using the identifier format specified by the ***Desired Identifier Result Type*** parameter.

This procedure shall be used for the discovery of resources under <*CSEBase*> that match the provided ***Filter Criteria*** parameter. The discovery result shall be returned to the Originator using a successful Response message.

Table 10.2.6.1-1: Discovery procedure via Retrieve Operation

***<resource>* RETRIEVE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: For the allowed *Result Content* parameter options for Discovery related RETRIEVE see clause 8.1.2. ***To:*** Address of the root of where the discovery begins. ***Filter Criteria:*** Filter criteria for searching and expected returned result. The *filterUsage* parameter shall be set in this case. ***Desired Identifier Result Type:*** optional, format of discovery results returned (see clause 8.1.2 for options applicable to Discovery, and how results shall be displayed).
Processing at Originator before sending Request	According to clause 10.1.3 with the following: • Setup the RETRIEVE operation in the Request. • Include the conditions in the filter criterion to limit the scope of the discovery results. • Specify the desired format of returned discovery results.
Processing at Receiver	According to clause 10.1.3 with the following specific processing: • Checks the validity of the Request (e.g. format of ***Filter Criteria***). • May change the filter criteria according to local policies. • Searches matched resources as per the DISCOVER privileges from the addressed resource hierarchy. Any resources whose status is marked as "INACTIVE" are not searched, as well as any child resources of these "INACTIVE" resources. • Limits the discovery result according to the upper limit on the size of the answer. The Hosting CSE shall use the appropriate addressing (see clause 9.3.1) form for each element included in the list in accordance with the incoming request. If ***Filter Criteria*** is provided in the request, the Hosting CSE uses it identifying the resources whose attributes match the ***Filter Criteria***. The Hosting CSE shall respond to the Originator with the appropriate list of discovered resources in the Hosting CSE. If the ***Filter Criteria*** includes *filterUsage* element set to "IPEOnDemandDiscovery", the target is the <AE> resource and the Hosting CSE has no match from the discovery of existing resources, then the Hosting CSE shall send a NOTIFY request containing the ***Filter Criteria*** to the AE(i.e. *pointOfAccess* of the <AE> resource) and the Originator ID of this discovery request. When the CSE gets the successful NOTIFY response with the resource address(es) which are created under the <AE> resource, then the CSE shall check the DISCOVER privilege and return the address(es) to the Originator. When the CSE gets the unsuccessful NOTIFY response, then the CSE shall send the ***Response Status Code*** in the NOTIFY response to the Originator. The Hosting CSE may modify the ***Filter Criteria*** including upper limit provided by the Originator or the discovery results based on the local policies. If the size of the result list is bigger than the upper limit or the scope of discoverable resources, according to the Originator's access control policy or service subscription has been modified by the Hosting CSE, the full list is not returned. Instead, an incomplete list is returned and an indication is added in the response for warning the requestor.
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: • Contains the address list of discovered resources expressed in any of the methods depicted in clause 9.3.1. The address list may be empty if no result matching the filter criterion is discovered. • Contains an incomplete list warning if the full list is not returned.
Processing at Originator after receiving Response	According to clause 10.1.3.
Exceptions	According to clause 10.1.3, with the following: • The request contains invalid parameters. • The on-demand discovery was rejected by the requested M2M Application.

10.2.6.2 Discovery with Result Content parameter

When a Retrieve request contains the ***Result Content*** parameter set to "attributes+child-resource-references" or "child-resource-references", the Hosting CSE returns child resource references which are child resource identifiers. A Retrieve request with the ***Result Content*** parameter can contain a ***Filter Criteria*** parameter for filtering child/descendant resources. The Hosting CSE checks for RETRIEVE privileges of the Originator to determine whether the matching resource identifier can be returned.

The Hosting CSE shall return the Retrieve results as a list of child resource references using the identifier format specified by the ***Desired Identifier Result Type*** parameter.

10.2.7 Group management

10.2.7.1 Introduction

This clause describes different procedures for managing membership verification, creation, retrieval, update and deletion of the information associated with a <*group*> resource. Bulk management of all group member resources by invoking the corresponding operations upon the virtual resource <*fanOutPoint*> of a <*group*> resource are detailed.

This clause also describes the use of the virtual resource <*semanticFanOutPoint*> which is supported only if the group hosting CSE supports semantic discovery functionality. This virtual resource, shall be the target of RETRIEVE requests only.

Figure 10.2.7.1-1 illustrates in a generic way how addressing a request to the <*fanOutPoint*> virtual resource on the group Hosting CSE results in a fanning out of the request to the group member resources. The procedures in the figure apply to clauses 10.2.7.6 to 10.2.7.9.

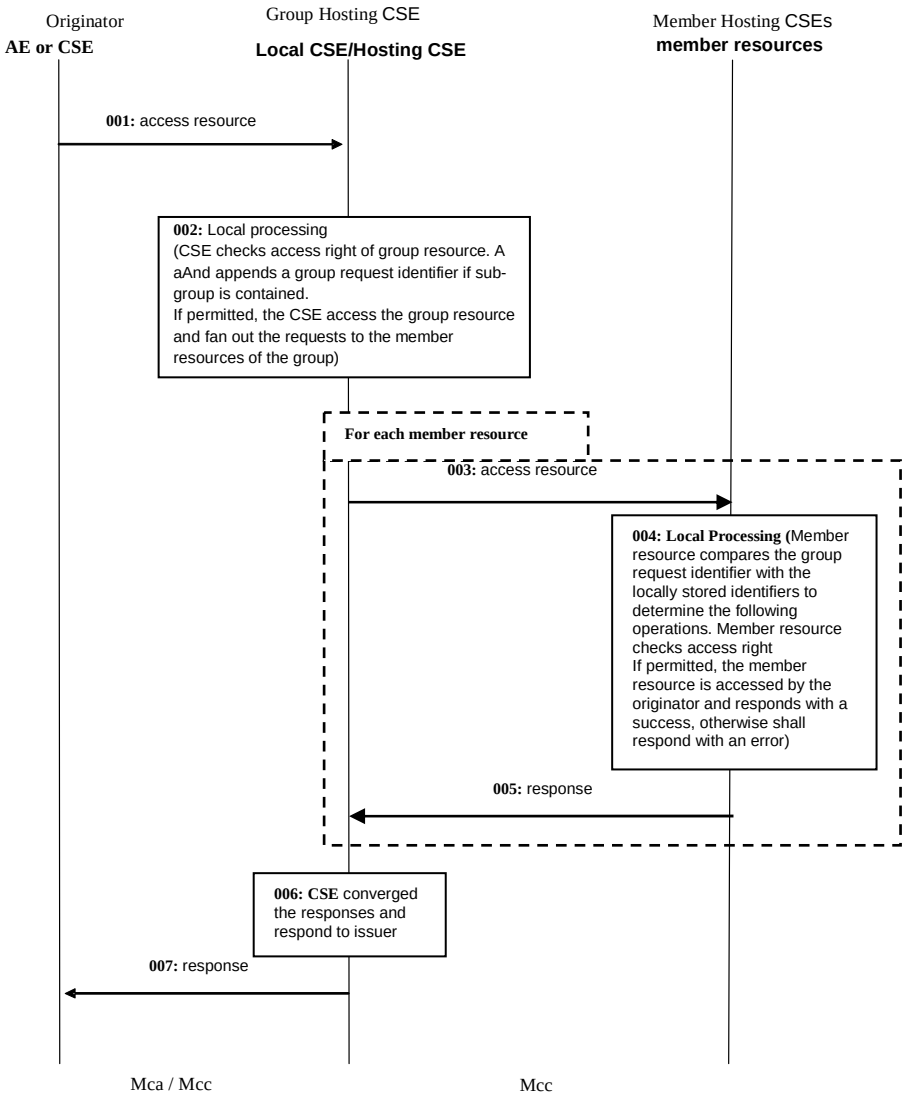


Figure 10.2.7.1-1: Group content management procedures

In step 002, if the group resource, whose *fanOutPoint* virtual sub-resource is addressed by the request contains a <group> resource suffixed with /fopt as a member resource, the Group Hosting CSE shall address the <*fanOutPoint*> virtual sub-resource of the member <group> resource in step 003 so that the member <group> resource Hosting CSE can fan out the request to its members correspondingly.

In step 004, the CSE hosting the member resource shall check to see if the request has a ***Group Request Identifier***. If it does, this CSE shall check this identifier against the list of ***Group Request Identifiers*** that it has stored locally. If a match is found, it shall ignore the current request and respond with an error. If no match is found, it shall proceed with the request and store the ***Group Request Identifier*** locally until the expiration of the request expiration time or local policy.

10.2.7.2 Create <*group*>

This procedure shall be used for creating a <*group*> resource.

Table 10.2.7.2-1: <*group*> CREATE

<*group*> CREATE	
Information in Request message	***From:*** Identifier of the AE or the CSE that initiates the Request. ***To:*** The address of the <*CSEBase*>, <*AE*>, *or* <*remoteCSE*> where the <*group*> resource is intended to be Created. ***Content:*** The representation of the <*group*> resource for which the attributes are described in clause 9.6.13.
Processing at Originator before sending Request	The Originator shall request to Create a <*group*> resource by using the CREATE operation. The request shall address <*CSEBase*>, <*remoteCSE*> *or* <*AE*> resource of a Hosting CSE. The Request shall also provide *memberIDs* and may provide *expirationTime* attributes. For members which are of type <group>, the originator shall suffix the '/fopt' to that 'memberID' during group creation if the originator wants to fan-out the group request to each member of that sub-<group>, else originator shall not suffix the '/fopt' to that 'memberID'. The Originator may be an AE or a CSE.
Processing at Receiver	For the CREATE procedure, the Receiver shall: • Check if the Originator has CREATE permissions on the target resource. • Check the validity of the provided attributes. • Validate that there are no duplicate members present in the *memberIDs* attribute. • Validate that the resource type of every member on each member Hosting CSE conforms to the *memberType* attribute in the request, if the *memberType* attribute of the <*group*> resource is not 'mixed'. Set the *memberTypeValidated* attribute to TRUE upon successful validation. • If *specializationType* attribute is set and the *memberType* attribute of the <*group*> resource is not 'mixed', then validate that the resource type of every member on each member Hosting CSE conforms to the specialized resource type set in *specializationType* attribute. Set the *memberTypeValidated* attribute to TRUE upon successful validation. • Upon successful validation of the provided attributes, create a new group resource in the Hosting CSE. If the CSE supports semantic discovery functionality, the Hosting CSE shall also create and set the *semanticSupportIndicator* attribute to TRUE. • If the registree Member Hosting CSEs and the Group Hosting CSE supports the same type of multicast communication, the Group Hosting CSE shall perform the procedures as specified in clause 10.2.7.13.1. • Conditionally, in the case that the group resource contains temporarily. unreachable Hosting CSE of sub-group resources as member resource, set the *memberTypeValidated* attribute of the <*group*> resource to FALSE. • Respond to the Originator with the appropriate generic Response with the representation of the <*group*> resource if the *memberTypeValidated* attribute is FALSE, and the address of the created <*group*> resource if the CREATE was successful. • As soon as any Hosting CSE that hosts the unreachable resource becomes reachable, the *memberType* validation procedure shall be performed. If the *memberType* validation fails, the Hosting CSE shall deal with the <*group*> resource according to the policy defined by the *consistencyStrategy* attribute of the <*group*> resource provided in the request. or by default if the attribute is not provided.
Information in Response message	The representation of the <*group*> resource if the *memberTypeValidated* attribute is FALSE.

<*group*> CREATE	
Processing at Originator after receiving Response	None.
Exceptions	No change from the basic procedure in clause 10.1.2.

10.2.7.3 Retrieve <*group*>

This procedure shall be used for retrieving <*group*> resource.

Table 10.2.7.3-1: <*group*> RETRIEVE

<*group*> RETRIEVE	
Information in Request message	***From:*** Identifier of the AE or the CSE that initiates the Request ***To:*** The address of the <*group*> resource
Processing at Originator before sending Request	The Originator shall request to obtain <*group*> resource information by using the RETRIEVE operation. The request shall address the specific <*group*> resource of a Hosting CSE. The Originator may be an AE or a CSE
Processing at Receiver	No change from the basic procedure in clause 10.1.3
Information in Response message	No change from the basic procedure in clause 10.1.3
Processing at Originator after receiving Response	None
Exceptions	No change from the basic procedure in clause 10.1.3

10.2.7.4 Update <*group*>

This procedure shall be used for updating an existing <*group*> resource.

Table 10.2.7.4-1: <*group*> UPDATE

<*group*> UPDATE	
Information in Request message	***From:*** Identifier of the AE or the CSE that initiates the Request. ***To:*** The address of the <*group*> resource.
Processing at Originator before sending Request	The Originator shall request to update attributes of an existing <*group*> resource by using an UPDATE operation. The Request shall address the specific <*group*> resource of a CSE. If originator intends to update memberIDs attribute, for members which are of type <group>, originator shall suffix the '/fopt' to that 'memberID' during group update if the originator wants to fan-out the group request to each member of that sub-<group>, else originator shall not suffix the '/fopt' to that 'memberID'. The Originator may be an AE or a CSE.

***<group>* UPDATE**	
Processing at Receiver	The UPDATE procedure shall be: • Check if the Originator has UPDATE permissions on the *<group>* resource. • Check the validity of provided attributes. • Validate that there are no duplicated members present in the *memberIDs* attribute. • Validate that the resource type of every member on each member Hosting CSE conforms to the *memberType* attribute in the request, if the *memberType* attribute of the *<group>* resource is not 'mixed'. Set the *memberTypeValidated* attribute to TRUE upon successful validation. • Validate that the resource type of every member on each member Hosting CSE conforms to the specialized resource type set in *specializationType* attribute, if the *memberType* attribute of the *<group>* resource is not 'mixed'. Set the *memberTypeValidated* attribute to TRUE upon successful validation. • Upon successful validation of the provided attributes, update the *<group>* resource in the Hosting CSE. Conditionally, if the *memberIDs* attribute changes and the group includes Multicast Group Information, Group Hosting CSE shall update or delete the Multicast Group Information according to the new group members and trigger the update or delete of <localMulticastGroup> on each member Hosting CSEs. The procedure is specified in the clause 10.2.7.15 or 10.2.7.16 correspondingly. • Conditionally, in the case that the *<group>* resource contains temporarily unreachable Hosting CSE of sub-group resources as members resource set the *memberTypeValidated* attribute of the *<group>* resource to FALSE. • Respond to the Originator with the appropriate generic response with the representation of the *<group>* resource if the *memberTypeValidated* attribute is FALSE, and the address of the created *<group>* resource if the UPDATE is successful. • As soon as any Hosting CSE that hosts unreachable resource becomes reachable, the *memberType* validation procedure shall be performed. If the *memberType* validation fails, the Hosting CSE shall deal with the *<group>* resource according to the policy defined by the *consistencyStrategy* attribute of the *<group>* resource provided in the request, or by default if the attribute is not provided.
Information in Response message	The representation of the <group> resource if the *memberTypeValidated* attribute is FALSE.
Processing at Originator after receiving Response	None.
Exceptions	No change from the basic procedure in clause 10.1.4.

10.2.7.5 Delete <*group*>

This procedure shall be used for deleting an existing <*group*> resource.

Table 10.2.7.5-1: <*group*> DELETE

<*group*> DELETE	
Information in Request message	***From:*** Identifier of the AE or the CSE that initiates the Request ***To:*** The address of the <*group*> resource
Processing at Originator before sending Request	The Originator shall request to delete an existing <*group*> resource by using the DELETE operation. The request shall address the specific <*group*> resource of a Hosting CSE. The Originator may be an AE or a CSE This operation shall also delete the child virtual resources <*fanOutPoint*> and <*semanticFanOutPoint*>
Processing at Receiver	Besides the basic procedure in clause 10.1.5, the receiver shall: • Check if the group includes Multicast Group information, and perform the multicast message fan out procedure to delete the <localMulticastGroup> of member hosting CSEs as specified in the clause 10.2.7.16
Information in Response message	No change from the basic procedure in clause 10.1.5
Processing at Originator after receiving Response	None
Exceptions	No change from the basic procedure in clause 10.1.5

10.2.7.6 Create <*fanOutPoint*>

This procedure does not create a new <fanOutPoint> resource. It is targeted at a <fanOutPoint> itself, and is used to create a new child resource under each member of the <group> that corresponds to that <fanOutPoint>.

Table 10.2.7.6-1: <*fanOutPoint*> CREATE

<*fanOutPoint*> CREATE	
Information in Request message	***From:*** Identifier of the AE or the CSE that initiates the Request ***To:*** The address of the <*fanOutPoint*> virtual resource ***Content:*** The representation of the resource the Originator intends to create ***Group Request Identifier:*** The group request identifier (present if the request has been forwarded from another <group>). ***Response Type:*** If the parameter is set to BlockingSynch, it indicates that the group hosting CSE shall return the aggregated response once. Otherwise if the parameter is set to nonBlockingRequestSynch, nonBlockingRequestAsynch or flexBlocking, it indicates that the Group Hosting CSE shall return the aggregated response in a batched mode ***Result Expiration Time:*** Indicates the maximum time limit in which the Group Hosting CSE has to respond the aggregated response.
Processing at Originator before sending Request	The Originator shall request to create the resource that have the same content in all member resources belonging to an existing <*group*> resource by using a CREATE operation. The Request may address the virtual child resource <*fanOutPoint*> of the specific <*group*> resource of a group Hosting CSE. The request may also address the address that results from appending a relative address to the <*fanOutPoint*> address in order to create the resources that have the same content under the corresponding child resources represented by the relative address with respect to all member resources. The Originator may be an AE or CSE.

***<fanOutPoint>* CREATE**	
Processing at Group Hosting CSE	For the CREATE procedure, the Group Hosting CSE shall: • Check if the Originator has CREATE privilege in the <*accessControlPolicy*> resource referenced by the *membersAccessControlPolicyIDs* in the <*group*> resource. In the case members *membersAccessControlPolicyIDs* is not provided the access control policy defined for the <*group*> resource shall be used. • Upon successful validation, obtain the IDs of all member resources from the attribute *membersIDs* of the addressed <*group*> resource. • If the group includes Multicast Group Information, the group Hosting CSE shall perform the procedure as described in clause 10.2.7.13.2. • If the request contains a ***Group Request Target Members*** parameter, it shall check whether all members contained in this parameter are a subset of the *memberIDs* attribute of the addressed <group> resource. If true, the request shall be fanned out to the members contained in this parameter only. • Generate fan out requests addressing the obtained address (appended with the relative address if any) to the member hosting CSEs as indicated in figure 10.2.7.1-1 which are not in the multicast group. The ***From*** parameter in the fanout request is set to the ID of the Originator from the request from the original Originator. The ***Response Type*** parameter in the fanout request may be set by the group hosting CSE differently according to its local policy. • In the case that any of these target addresses involves a further <fanOutPoint> and the request to be fanned out does not contain a ***Group Request Identifier*** already, generate a unique group request identifier, include it in all the requests to be fanned out and store this group request identifier locally. • If the group Hosting CSE determines that multiple member resources belong to one CSE according to the IDs of the member resources and no multicast group exists for these members, it may converge the requests accordingly before sending out. This may be accomplished by the group Hosting CSE creating a <*group*> resource on the member Hosting CSE to collect all the members on that member Hosting CSE. • After receiving the responses from the member hosting CSEs, respond to the Originator with the aggregated results and the associated members list. Depending on the ***Response Type***, the Group Hosting CSE shall: - **blockingRequest:** respond with the aggregated responses before the ***Result Expiration Time*** reaches and discard the member responses received after. - **nonBlockingRequestSynch:** prepare the *operationResult* of the <request> resource and indicate that if all the member responses have been aggregated by setting the *requestStatus* of the <request> resource before the ***Result Expiration Time*** reaches. There may be multiple updates of the *operationResult* attribute. - **nonBlockingRequestAsynch:** notify with the aggregated response from all or part of the members before the ***Result Expiration Time*** reaches. There may be more than one notifications. - **flexBlocking:** continue aggregate the member response until the group hosting CSE determines to send the aggregated responses, if all member responses has been aggregated, respond the aggregated response as in the blockingRequest case. Otherwise, respond an acknowledgement together with the current aggregated member responses and the reference to the created <request> resource. Then continue aggregate and deliver the remaining member response to the Originator as defined in the nonBlockingRequestSynch or the nonBlockingRequestAsynch case. - After the ***Result Expiration Time***, there shall not be any further updates to the aggregated responses. (See note)
Processing at Member Hosting CSE	For the CREATE procedure, the Member Hosting CSE shall: • Check if the request has a ***Group Request Identifier*** and if so, process it as described in step 004 of clause 10.2.7.1. • Check if the original Originator has the CREATE permission on the addressed resource. Upon successful validation, perform the create procedures for the corresponding type of addressed resource as described in other sub-clauses of clause 10.2. • Send the corresponding response to the Group Hosting CSE.
Information in Response message	Converged responses from member hosting CSEs.

***<fanOutPoint>* CREATE**	
Processing at Originator after receiving Response	None.
Exceptions	• Same request with identical ***Group Request Identifier*** received. • Originator does not have CREATE permission to access the *<fanOutPoint>* resource. • Members in the ***Group Request Target Members*** request parameter are not included in the *memberIDs* attribute of the addressed <group> resource.
NOTE: If ***Result Expiration Time*** is not provided in the original request from the Originator, the group hosting CSE may decide the timer based on its local policy.	

10.2.7.7 Retrieve *<fanOutPoint>*

This procedure shall be used for retrieving the content of all member resources belonging to an existing *<group>* resource.

Table 10.2.7.7-1: *<fanOutPoint>* RETRIEVE

***<fanOutPoint>* RETRIEVE**	
Information in Request message	***From:*** Identifier of the AE or the CSE that initiates the Request ***To:*** The address of the *<fanOutPoint>* virtual resource ***Content:*** The representation of the resource the Originator intends to retrieve ***Group Request Identifier:*** The group request identifier (present if the request has been forwarded from another <group>). ***Response Type:*** If the parameter is set to BlockingSynch, it indicates that the group hosting CSE shall return the aggregated response once. Otherwise if the parameter is set to nonBlockingRequestSynch or nonBlockingRequestAsynch, it indicates that the Group Hosting CSE shall return the aggregated response in a batched mode. ***Result Expiration Time:*** Indicates the maximum time limit in which the Group Hosting CSE has to respond the aggregated response.
Processing at Originator before sending Request	The Originator shall request to obtain the resource or specific attributes of all member resources belonging to an existing *<group>* resource by using a RETRIEVE operation. The request may address the virtual child resource *<fanOutPoint>* of the specific *<group>* resource of a group Hosting CSE. The request may also address the address that results from appending a relative address to the *<fanOutPoint>* address in order to retrieve the corresponding attributes or child resources represented by the relative address with respect to all member resources. The Originator may be an AE or CSE.
Processing at Group Hosting CSE	For the RETRIEVE procedure, the Group Hosting CSE shall: • Check if the Originator has RETRIEVE permission in the *<accessControlPolicy>* resource referenced by the *membersAccessControlPolicyIDs* in the addressed *<group>* resource. In the case *membersAccessControlPolicyIDs* is not provided, the access control policy defined for the group resource shall be used • Upon successful validation, obtain the IDs of all member resources from the *membersIDs* attribute of the addressed *<group>* resource • If the group includes Multicast Group information, the group Hosting CSE shall perform the procedure as described in clause 10.2.7.13.2 • If the request contains a ***Group Request Target Members*** parameter, it shall check whether all members contained in this parameter are a subset of the *memberIDs* attribute of the addressed <group> resource. If true, the request shall be fanned out to the members contained in this parameter only. • Generate fan out requests addressing the obtained address (appended with the relative address if any) to the member hosting CSEs as indicated in figure 10.2.7.1-1. The ***From*** parameter in the fanout request is set to the ID of the Originator from the request from the original Originator. The ***Response Type*** parameter in the fanout request may be set by the group hosting CSE differently according to its local policy • In the case that any of these target addresses involves a further <fanOutPoint> and the request to be fanned out does not contain a ***Group Request Identifier*** already, generate a unique group request identifier, include it in all the requests to be fanned out and store this group request identifier locally.

***<fanOutPoint>* RETRIEVE**	
	• If the group hosting CSE determines that multiple member resources belong to one CSE according to the IDs of the member resources and no multicast group exists for these members, it may converge the requests accordingly before sending out. This may be accomplished by the group Hosting CSE creating a *<group>* resource on the member Hosting CSE to collect all the members on that member Hosting CSE • After receiving the responses from the member hosting CSEs, respond to the Originator with the aggregated results and the associated members list. Depending on the ***Response Type***, the Group Hosting CSE shall: - BlockingRequest: respond with the aggregated responses before the ***Result Expiration Time*** reaches and discard the member responses received after. - nonBlockingRequestSynch: prepare the *operationResult* of the <request> resource and indicate that if all the member responses have been aggregated by setting the *requestStatus* of the <request> resource before the ***Result Expiration Time*** reaches. There may be multiple updates of the *operationResult* attribute. - nonBlockingRequestAsynch: notify with the aggregated response from all or part of the members before the ***Result Expiration Time*** reaches. There may be more than one notifications. - flexBlocking: continue aggregate the member response until the group hosting CSE determines to send the aggregated responses. If all member responses has been aggregated, respond the aggregated response as in the blockingRequest case. Otherwise, respond an acknowledgement together with the current aggregated member responses and the reference to the created <request> resource. Then continue aggregate and deliver the remaining member response to the Originator as defined in the nonBlockingRequestSynch or the nonBlockingRequestAsynch case. - After the ***Result Expiration Time***, there shall not be any further updates to the aggregated responses. (See note)
Processing at Member Hosting CSE	For the RETRIEVE procedure, the Member Hosting CSE shall: • Check if the request has a ***Group Request Identifier*** and if so, process it as described in step 004 of clause 10.2.7.1. • Check if the original Originator has the RETRIEVE permission on the addressed resource. Upon successful validation, perform the retrieve procedures for the corresponding type of addressed resource as described in other sub-clauses of clause 10.2. • Send the corresponding response to the group Hosting CSE.
Information in Response message	Converged responses from member hosting CSEs.
Processing at Originator after receiving Response	None.
Exceptions	• Same request with identical ***Group Request Identifier*** received. • Originator does not have RETRIEVE permission to access the *<fanOutPoint>* resource. • Members in the ***Group Request Target Members*** request parameter are not present in the *memberIDs* attribute of the addressed <group> resource.
NOTE: If ***Result Expiration Time*** is not provided in the original request from the Originator, the group hosting CSE may decide the timer based on its local policy.	

10.2.7.8 Update *<fanOutPoint>*

This procedure shall be used for updating the content of all member resources belonging to an existing *<group>* resource.

Table 10.2.7.8-1: *<fanOutPoint>* UPDATE

***<fanOutPoint>* UPDATE**	
Information in Request message	***From:*** Identifier of the AE or the CSE that initiates the Request ***To:*** The address of the <*group*> resource ***Content:*** The representation of the resource the Originator intend to Update ***Group Request Identifier:*** The group request identifier (present if the request has been forwarded from another <group>). ***Response Type:*** If the parameter is set to BlockingSynch, it indicates that the group hosting CSE shall return the aggregated response once. Otherwise if the parameter is set to nonBlockingRequestSynch or nonBlockingRequestAsynch, it indicates that the Group Hosting CSE shall return the aggregated response in a batched mode ***Result Expiration Time:*** Indicates the maximum time limit in which the Group Hosting CSE has to respond the aggregated response.
Processing at Originator before sending Request	The Originator shall request to update all member resources belonging to an existing <*group*> resource with the same data by using a UPDATE operation. The request may address the virtual child resource <*fanOutPoint*> of the specific <*group*> resource of a group Hosting CSE. The request may also address the address that results from appending a relative address to the <*fanOutPoint*> in order to update the corresponding child resources represented by the relative address with respect to all <*members*> resources. The Originator may be an AE or CSE
Processing at Group Hosting CSE	For the UPDATE procedure, the Group Hosting CSE shall: • Check if the Originator has UPDATE permission in the <*accessControlPolicy*> resource referenced by the *membersAccessControlPolicyIDs* in the group resource. In the case members *membersAccessControlPolicyIDs* is not provided the access control policy defined for the group resource shall be used • Upon successful validation, obtain the IDs of all member resources from the attribute *membersIDs* of the addressed <*group*> resource • If the group includes Multicast Group information, the group Hosting CSE shall perform the procedure as described in clause 10.2.7.13.2 • If the request contains a ***Group Request Target Members*** parameter, it shall check whether all members contained in this parameter are a subset of the *memberIDs* attribute of the addressed <group> resource. If true, the request shall be fanned out to the members contained in this parameter only. • Generate fan out requests addressing the obtained address (appended with the relative address if any) to the member hosting CSEs as indicated in figure10.2.7.1-1 which are not in the multicast group. The ***From*** parameter in the fanout request is set to the ID of the Originator from the request from the original Originator. The ***Response Type*** parameter in the fanout request may be set by the group hosting CSE differently according to its local policy • In the case that any of these target addresses involves a further <fanOutPoint> and the request to be fanned out does not contain a ***Group Request Identifier*** already, generate a unique group request identifier, include it in all the requests to be fanned out and store this group request identifier locally. • If the group Hosting CSE determines that multiple member resources belong to one CSE according to the IDs of the member resources and no multicast group exists for these members, it may converge the requests accordingly before sending out. This may be accomplished by the group Hosting CSE creating a <*group*> resource on the member Hosting CSE to collect all the members on that member Hosting CSE • After receiving the responses from the member hosting CSEs, respond to the Originator with the aggregated results and the associated members list. Depending on the ***Response Type***, the Group Hosting CSE shall: - BlockingRequest: respond with the aggregated responses before the ***Result Expiration Time*** reaches and discard the member responses received after. - nonBlockingRequestSynch: prepare the *operationResult* of the <request> resource and indicate that if all the member responses have been aggregated by setting the *requestStatus* of the <request> resource before the ***Result Expiration Time*** reaches. There may be multiple updates of the *operationResult* attribute. - nonBlockingRequestAsynch: notify with the aggregated response from all or part of the members before the ***Result Expiration Time*** reaches. There may be more than one notifications.

***<fanOutPoint>* UPDATE**	
	- flexBlocking: continue aggregate the member response until the group hosting CSE determines to send the aggregated responses, if all member responses has been aggregated, respond the aggregated response as in the blockingRequest case. Otherwise, respond an acknowledgement together with the current aggregated member responses and the reference to the created <request> resource. Then continue aggregate and deliver the remaining member response to the Originator defined in the nonBlockingRequestSynch or the nonBlockingRequestAsynch case. - After the ***Result Expiration Time***, there shall not be any further updates to the aggregated responses. (See note)
Processing at Member Hosting CSE	For the UPDATE procedure, the Member Hosting CSE shall: • Check if the request has a ***Group Request Identifier*** and if so, process it as described in step 004 of clause 10.2.7.1. • Check if the original Originator has the UPDATE permission on the addressed resource. Upon successful validation, perform the update procedures for the corresponding type of addressed resource as described in other sub-clauses of clause 10.2. • Send the corresponding response to the group Hosting CSE.
Information in Response message	Converged responses from member hosting CSEs.
Processing at Originator after receiving Response	None.
Exceptions	• Same request with identical ***Group Request Identifier*** received. • Originator does not have UPDATE permission to access the *<fanOutPoint>* resource. • Members in the ***Group Request Target Members*** request parameter are not present in the *memberIDs* attribute of the addressed <group> resource.
NOTE: If ***Result Expiration Time*** is not provided in the original request from the Originator, the group hosting CSE may decide the timer based on its local policy.	

10.2.7.9 Delete *<fanOutPoint>*

This procedure shall be used for deleting the content of all member resources belonging to an existing *<group>* resource.

Table 10.2.7.9-1: *<fanOutPoint>* DELETE

***<fanOutPoint>* DELETE**	
Information in Request message	***From:*** Identifier of the AE or the CSE that initiates the Request. ***To:*** The address of the *<fanOutPoint>* virtual resource. ***Content:*** The representation of the resource the Originator intends to delete ***Group Request Identifier:*** The group request identifier (present if the request has been forwarded from another <group>). ***Response Type:*** If the parameter is set to BlockingSynch, it indicates that the group hosting CSE shall return the aggregated response once. Otherwise if the parameter is set to nonBlockingRequestSynch or nonBlockingRequestAsynch, it indicates that the Group Hosting CSE shall return the aggregated response in a batched mode. ***Result Expiration Time:*** Indicates the maximum time limit in which the Group Hosting CSE has to respond the aggregated response.
Processing at Originator before sending Request	The Originator shall request to delete all member resources belonging to an existing *<group>* resource by using a DELETE operation. The request may address the virtual child resource *<fanOutPoint>* of the specific *<group>* resource of a group Hosting CSE. The request may also address the address that results from appending a relative address to the *<fanOutPoint>* in order to delete the corresponding child resources represented by the relative address with respect to all member resources. The Originator may be an AE or a CSE.
Processing at Group Hosting CSE	For the DELETE procedure, the *<group>* Hosting CSE shall: • Check if the Originator has DELETE permission in the *<accessControlPolicy>* resource referenced by the *membersAccessControlPolicyIDs* in the *<group>* resource. In the case *membersAccessControlPolicyIDs* is not provided the access control policy defined for the group resource shall be used.

***<fanOutPoint>* DELETE**	
	• Upon successful validation, obtain the IDs of all member resources from the attribute *membersIDs* of the addressed <*group*> resource. • If the group includes Multicast Group information, the group Hosting CSE shall perform the procedure as described in clause 10.2.7.13.2. • If the request contains a ***Group Request Target Members*** parameter, it shall check whether all members contained in this parameter are a subset of the *memberIDs* attribute of the addressed <group> resource. If true, the request shall be fanned out to the members contained in this parameter only. • Generate fan out requests addressing the obtained address (appended with the relative address if any) to the member hosting CSEs as indicated in figure 10.2.7.1-1 which are not in the multicast group. The ***From*** parameter in the fanout request is set to the ID of the Originator from the request from the original Originator. The ***Response Type*** parameter in the fanout request may be set by the group hosting CSE differently according to its local policy. • In the case that any of these target addresses involves a further <fanOutPoint> and the request to be fanned out does not contain a ***Group Request Identifier*** already, generate a unique group request identifier, include it in all the requests to be fanned out and store this group request identifier locally. • If the <*group*> Hosting CSE determines that multiple member resources belong to one CSE according to the IDs of the member resources and no multicast group exists for these members, it may converge the requests accordingly before sending out. This may be accomplished by the group Hosting CSE creating a <*group*> resource on the member Hosting CSE to collect all the members on that member Hosting CSE. • After receiving the responses from the member hosting CSEs, respond to the Originator with the aggregated results and the associated members list. Depending on the ***Response Type***, the Group Hosting CSE shall: - BlockingRequest: respond with the aggregated responses before the ***Result Expiration Time*** reaches and discard the member responses received after. - nonBlockingRequestSynch: prepare the *operationResult* of the <request> resource and indicate that if all the member responses have been aggregated by setting the *requestStatus* of the <request> resource before the ***Result Expiration Time*** reaches. There may be multiple updates of the *operationResult* attribute. - nonBlockingRequestAsynch: notify with the aggregated response from all or part of the members before the ***Result Expiration Time*** reaches. There may be more than one notifications. - flexBlocking: continue aggregate the member response until the group hosting CSE determines to send the aggregated responses, if all member responses has been aggregated, respond the aggregated response as in the blockingRequest case. Otherwise, respond an acknowledgement together with the current aggregated member responses and the reference to the created <request> resource. Then continue aggregate and deliver the remaining member response to the Originator as defined in the nonBlockingRequestSynch or the nonBlockingRequestAsynch case. - After the ***Result Expiration Time***, there shall not be any further updates to the aggregated responses. (See note)
Processing at Member Hosting CSE	For the DELETE procedure, the Members Hosting CSE shall: • Check if the request has a ***Group Request Identifier*** and if so, process it as described in step 004 of clause 10.2.7.1. • Check if the original Originator has the DELETE permission on the addressed resource. Upon successful validation, perform the delete procedures for the corresponding type of addressed resource as described in other sub-clauses of clause 10.2. • Send the corresponding response to the Group Hosting CSE.
Information in Response message	Converged responses from members hosting CSEs
Processing at Originator after receiving Response	None

***<fanOutPoint>* DELETE**	
Exceptions	• Same request with identical ***Group Request Identifier*** received • Originator does not have DELETE permission to access the *<fanOutPoint>* resource • Members in the ***Group Request Target Members*** request parameter are not present in the *memberIDs* attribute of the addressed <group> resource
NOTE: If ***Result Expiration Time*** is not provided in the original request from the Originator, the group hosting CSE may decide the timer based on its local policy.	

10.2.7.10 Subscribe and Un-Subscribe *<fanOutPoint>* of a group

The following procedure shall be used for receiving information about modifications of all member resources belonging to an existing *<group>* resource.

Table 10.2.7.10-1: *<fanOutPoint>* Subscribe

***<fanOutPoint>* Subscribe**	
Information in Request message	***From:*** Identifier of the AE or CSE that initiates the request. ***To:*** The address of the <fanOutPoint> resource appended with the ID of the *<subscription>* resource to be created ***Group Request Identifier:*** The group request identifier (present if the request has been forwarded from another <group>).
Processing at Originator before sending Request	The Originator shall request to create a subscription resource under all member resources belonging to an existing *<group>* resource by using a CREATE operation. The request may address the virtual child resource *<fanOutPoint>* of the specific *<group>* resource of a group Hosting CSE. The request may also address the address that results from appending a relative address to the *<fanOutPoint>* in order to create the corresponding subscription to the resource represented by the relative address with respect to all member resources. In both cases the targeted resource(s) shall be the parents of the newly created <subscription> resource(s). The request shall include *notificationForwardingURI* attribute if the Originator wants the group Hosting CSE to aggregate the notifications. The request shall include the required information and may include the optional information as described in subscription management clause 10.2.10. The Originator may be an AE or a CSE.
Processing at Group Hosting CSE	The *<group>* Hosting CSE shall: • Check if the Originator has CREATE privilege in the *<accessControlPolicy>* resource referenced by the *membersAccessControlPolicyIDs* in the group resource. In the case *membersAccessControlPolicyIDs* is not provided the access control policy defined for the group resource shall be used. • If the subscription resource in the request contains an *notificationForwardingURI* attribute, assign a URI to replace the *notificationURI* of the subscription resource which will be used to receive notifications from member hosting CSEs. The ID of the *<group>* resource shall be set to the *groupID* attribute of the *<subscription>* resource. The group Hosting CSE shall maintain the mapping of the generated *notificationURI* and the former *notificationURI.* • Upon successful validation, obtain the IDs of all member resources from the attribute membersIDs of the addressed <group> resource. • If the group includes Multicast Group Information, the group Hosting CSE shall perform the procedure as described in clause 10.2.7.13.2. • If the request contains a Group Request Target Members parameter, it shall check whether all members contained in this parameter are a subset of the memberIDs attribute of the addressed <group> resource. If true, the request shall be fanned out to the members contained in this parameter only. • Generate fan out requests addressing the obtained address (appended with the relative address if any) to the member hosting CSEs as indicated in figure 10.2.7.1-1 which are not in the multicast group. The ***From*** parameter in the fanout request is set to the ID of the Originator from the request from the original Originator. The ***Response Type*** parameter in the fanout request may be set by the group hosting CSE differently according to its local policy. • In the case that any of these target addresses involves a further <fanOutPoint> and the request to be fanned out does not contain a ***Group Request Identifier*** already, generate a unique group request identifier, include it in all the requests to be fanned out and store this group request identifier locally.

***<fanOutPoint>* Subscribe**	
	• If the group Hosting CSE determines that multiple members resources belong to one CSE according to the IDs of the member resources, it may converge the requests accordingly before sending out. This may be accomplished by the *<group>* Hosting CSE creating a *<group>* resource on the member Hosting CSE to collect all the members on that member Hosting CSE. • After receiving the responses from the member hosting CSEs, respond to the Originator with the aggregated results and the associated members list.
Processing at Member Hosting CSE	For the subscribe/un-subscribe procedure, the Member Hosting CSE shall treat the request received from the group Hosting CSE as a normal SUBSCRIBE request on the addressed member resource as if it comes from the original Originator. Therefore the member Hosting CSE shall: Check if the request has a ***Group Request Identifier*** and if so, process it as described in step 004 of clause 10.2.7.1. • Check if the original Originator has the READ permission on the members resource. • Upon successful validation, perform the subscribe procedures for the corresponding type of member resource as described in clause 10.2.10 • Send the corresponding response to the group Hosting CSE.
Information in Response message	Converged responses from member hosting CSEs
Processing at Originator after receiving Response	None
Exceptions	• Same request with identical ***Group Request Identifier*** received • Originator does not have CREATE permission to access the *<fanOutPoint>* resource • Members in the ***Group Request Target Members*** request parameter are not present in the *memberIDs* attribute of the addressed <group> resource.

Un-subscribing to the members of a <group> resource uses the "Delete <fanOutPoint>" procedure defined in clause 10.2.7.9.

A typical example of how the subscription is established is as follows. The Originator is creating subscription resource on Member-1 resource, Member-2 resource and Member-3 resource. Member-2 resource and Member-3 resource are members of Group-2 resource. Member-1 resource and Group-2 resource are members of Group-1 resource. In this case, Group-2 resource is the sub-group of Group-1 resource.

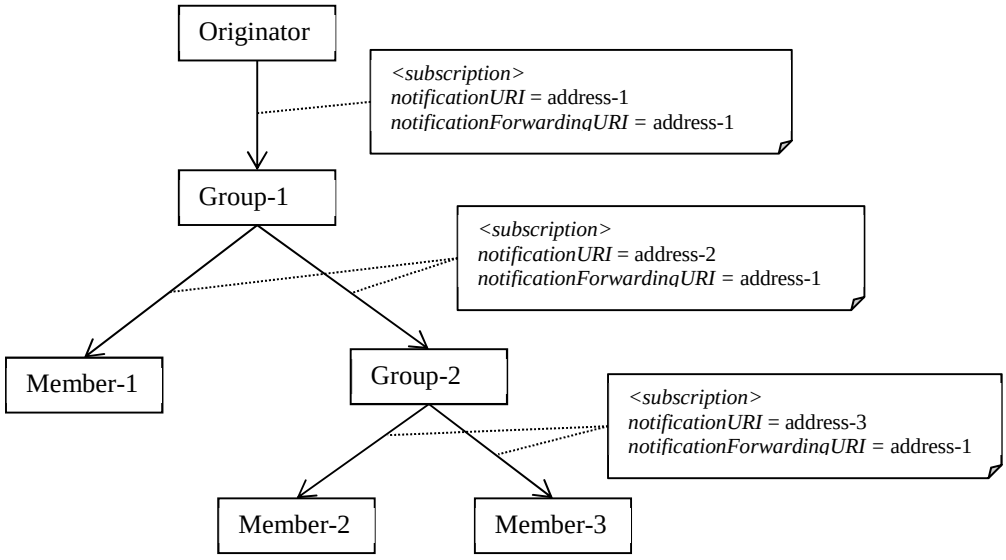


Figure 10.2.7.10-1: Example of subscription through group

Originator sends the <*subscription*> resource creation request to <*fanOutPoint*> of Group-1 resource. The Originator intends the Group-1 Hosting CSE to aggregate the notifications, thus, the Originator sets the *notificationForwardingURI* identical with *notificationURI* which is address-1 which is the address where the notification is supposed to be sent.

On receiving the request, the Group-1 Hosting CSE fans out the <*subscription*> creation request to address Member-1 resource and <*fanOutPoint*> resource of Group-2 resource. As *notificationForwardingURI* is set by the Originator, the Group-1 Hosting CSE allocates address-2 to receive aggregated notifications and put address-2 in the *notificationURI* of <*subscription*> resource to be fanned out.

On receiving the request, the Group-2 Hosting CSE fans out the <*subscription*> creation request to address Member-2 resource and Member-3 resource. As *notificationForwardingURI* is set, the Group-2 Hosting CSE allocates address-3 to receive aggregated notifications and put address-3 in the *notificationURI* of <*subscription*> resource to be fanned out. The mapping between address-2 and address-3 is maintained by the Group-2 Hosting CSE.

On receiving the request by any of the Member Hosting CSE, <*subscription*> resource is created.

10.2.7.11 Aggregate the Notifications by group

This procedure shall be used for the group Hosting CSE to aggregate the notifications from member hosting CSEs and forward the aggregated notification to the subscriber.

Table 10.2.7.11-1: Aggregation of Notifications by group

Aggregate Notifications by group	
Information in Request message	The same as table 10.2.10.7-1
Processing at Originator before sending Request (Member Hosting CSE)	Whenever the resource that is subscribed-to is modified in a way that matches the policies as is specified in clause 9.6.8, notification needs to be sent to the subscriber, the Members Hosting CSE shall: • Notify the subscriber at the notificationURI and include the *notificationForwardingURI* in the notification, if it exists
Processing at Group Hosting CSE	For the notification procedure, the Group Hosting CSE shall: • On receiving the notifications from the member hosting CSEs at the notificationURI generated by the group Hosting CSE during fanning out the <*subscription*> creation request, validate if the notification is sent from its member resource and contains a *notificationForwardingURI* attribute. • Upon successful validation, aggregate the notifications which have the same *notificationForwardingURI* for the duration specified in *notifyAggregation* or until the number of notifications specified in *notifyAggregation* are received, whichever occurs first. • Send the aggregated notification to the subscriber according to the *notificationForwardingURI* in the notification. In the case the addressed group is the member of another group through which the subscription is created the notification shall be sent to the notificationURI assigned by the group hosting CSE which contains the addressed group as the sub-group according to the mapping of the *notificationURI* maintained by the addressed group. • Wait for the response. After receiving the response, split the response and respond to the members hosting CSEs separately • The group Hosting CSE may stop aggregating the notifications when the expirationTime of the corresponding subscription expires
Processing at Member Hosting CSE	The subscriber shall treat every notification extracted from the aggregated notification as a separate notification received from the subscribed resource and generate corresponding responses. The subscriber shall aggregate the responses to these notifications and send the aggregated response to the group Hosting CSE
Information in Response message	According to clause 10.1.6
Processing at Originator after receiving Response	According to clause 10.1.6
Exceptions	According to clause 10.1.6

The example of aggregating notification following example in clause 10.2.7.10 is as follows.

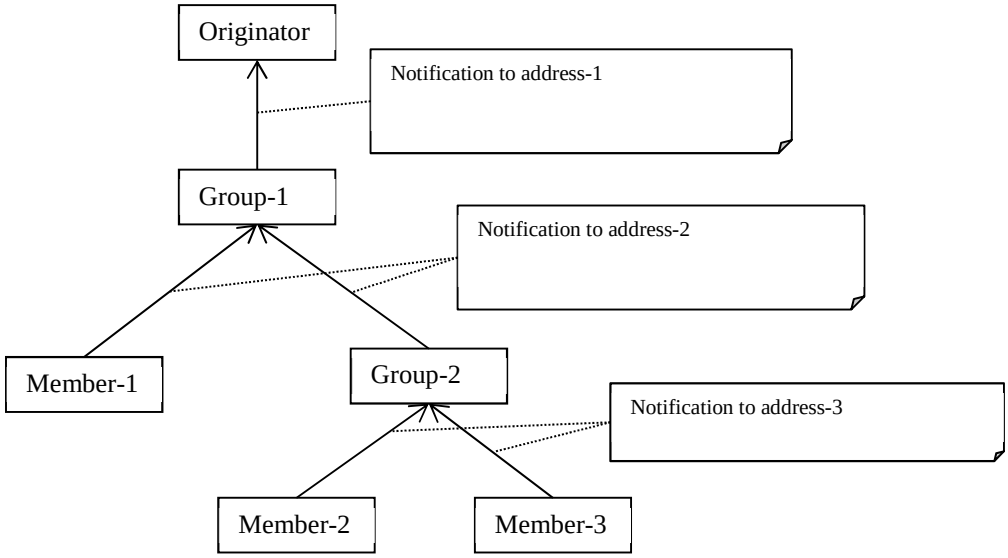


Figure 10.2.7.11-1: Example of aggregating notifications

Member Hosting CSEs sends notifications to their corresponding Group Hosting CSEs. In this example, Member-2 Hosting CSE and Member-3 Hosting CSE send notifications to address-3 which is the address allocated by the Group-2 Hosting CSE. Member-1 Hosting CSE sends notifications to address-2 which is the address allocated by the Group-1 Hosting CSE.

On receiving notifications by the Group-2 Hosting CSE at address-3, Group-2 Hosting CSE aggregates the notification that has *notificationForwardingURI* as address-1, as Group-2 Hosting CSE maintains the mapping of address-3 and address-2, and sends the aggregated notification to address-2.

On receiving notifications by the Group-1 Hosting CSE at address-2, Group-1 Hosting CSE aggregates the notification that has *notificationForwardingURI* as address-1, as Group-1 Hosting CSE receives <*subscription*> resource with *notificationURI* address-1, Group-1 Hosting CSE send the aggregated notification to address-1.

10.2.7.12 Retrieve <*semanticFanOutPoint*>

This procedure shall be used for performing semantic discovery or semantic query procedure using the descriptor content of all member <*semanticDescriptor*> resources belonging to an existing <*group*> resource. The Hosting CSE support of semantic discovery functionality via <*semanticFanOutPoint*> virtual resource is indicated by the *semanticSupportIndicator* attribute set to TRUE.

For detailed descriptions of the semantic discovery procedure see oneM2M TS-0034 [14], clause 6.2.

10.2.7.13 Multicast Group Management Procedures

10.2.7.13.0 Introduction

In case the multicasting is used to fan-out group operations to members of a <*group*> resource, the Group Hosting CSE shall maintain information specified in table 10.2.7.13.0-1 pertaining to the multicast group(s) that are applicable to a <*group*> resource, such as the mapping relationship between the member resources and the multicast address(es) and the target URI(s) in the fan out requests.

For each multicast group of a <group> resource, there may be multiple records of Multicast Group Information. The Group Hosting CSE should manage and index the multiple records by internal identifier for each Multicast Group Information. The definition of the internal identifier is out of scope of the present document.

Table 10.2.7.13.0-1: Multicast Group Information managed by Group Hosting CSE

Information	Multiplicity	Description
groupID	1	Indicates the <*group*> resource identifier which the Multicast Group Information belongs to.
multicastType	1	• Indicates the underlying networks multicast capability of the group members, the value is the same as the attribute *multicastCapability* of the <remoteCSE>. See clause 9.6.4.
externalGroupID	0..1	See table 9.6.44-2.
multicastAddress	1	See table 9.6.44-2.
multicastGroupFanoutTarget	1	See table 9.6.44-2.
memberList	1(L)	See table 9.6.44-2.
groupServiceServerAddress	0..1	It shall be present only when the multicastType is 3GPP_MBMS_group, and it is the address of 3GPP group service server (e.g. SCEF).
TMGI	0..1	The Temporary Mobile Group Identity is allocated to identify the MBMS bearer service as specified in ETSI TS 123 246 [i.32]. It is used to communicate with 3GPP networking with *externalGroupID* together.
TMGIExpiration	0..1	Indicates the duration of the TMGI expiration time as specified in ETSI TS 123 468 [i.33].
responseTimeWindow	0..1	Defines the upper bound on the amount of delay the Member Hosting CSE shall wait before sending a response message. The Member Hosting CSE shall wait a randomized time that is less than the value of this attribute. This randomized delay helps prevent network congestion caused by multiple Member Hosting CSEs responding at the same time as one another.

10.2.7.13.1 Multicast Group Information and <localMulticastGroup> Creation Procedures

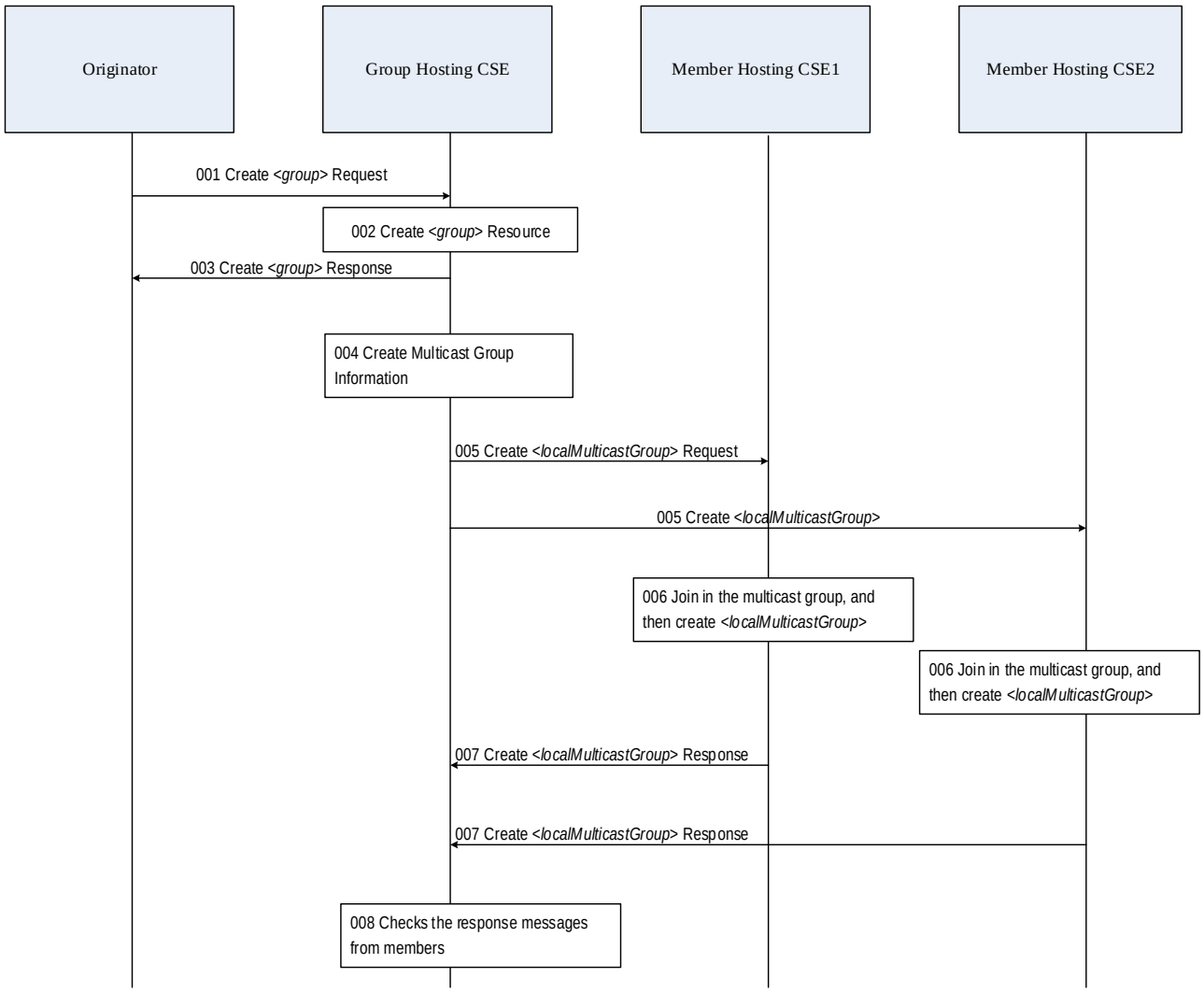


Figure 10.2.7.13.1-1: Multicast Group Information and <*localMulticastGroup*> creation procedures

Figure 10.2.7.13.1-1 illustrates how the Multicast Group Information and <*localMulticastGroup*> resource works on the Group Hosting CSE and the group members Hosting CSEs.

Step 001: The Originator sends a group resource creation request to the Group Hosting CSE which includes member resource identifier list consisting of multiple resources hosted on Member Hosting CSEs.

For example, different member resources identifiers are:

- /CSExx1/aa
- /CSExx1/bb
- /CSExx2/cc
- /CSExx2/dd

Step 002: The Group Hosting CSE shall create the group resource as requested.

Step 003: The Group Hosting CSE shall return the group creation response to the Originator.

Step 004: The Group Hosting CSE shall check the *multicastCapability* attribute of each Member Hosting CSE's <*remoteCSE*> resource to determine whether it could create a multicast group. If no member hosting CSEs support multicast capability or no more than one Member Hosting CSE supports the same multicast capability, then the Group Hosting CSE shall not create a multicast group.

If at least two Member Hosting CSEs support the same multicast capability, the Group Hosting CSE determines to create multicast group, and performs all the actions: assign the multicast type based on the multicast capability, and allocate multicast address and multicast address type to the member resources of the multicast group. For a guide to an allocation scheme of IPv4 and IPv6 multicast address spaces, reference the specification documents such as IETF RFC 3171 [i.34] and IETF RFC 4291 [i.35].

- If the *multicastType* is 3GPP_MBMS_group, and the Member Hosting CSEs have the same *externalGroupID*, the Group Hosting CSE shall set the *externalGroupID* to the same value as the *externalGroupID* of the Member Hosting CSEs, set the *responseTarget* to the *CSE-ID* of the <*CSEBase*> resource of the Group Hosting CSE, set the *memberList* to the members defined in the request, allocate a virtual fan out target for *multicastGroupFanoutTarget* according to the member resource identifiers, and establish the mapping relationship between the fanout target and the member resource identifiers.

NOTE 1: Additional details for creating a 3GPP MBMS group are specified in clause 7.7.3.1 of ETSI TS 118 126 [15].

- If the *multicastType* is IP_multicast_group, the Group Hosting CSE shall set the *memberList* to the members defined in the request, allocate a virtual fan out target for *multicastGroupFanoutTarget* according to the member resource identifiers, establish the mapping relationship between the fanout target and the member resource identifiers and set the Group Hosting CSE *resourceID* as the *responseTarget*.

NOTE 2: The current group based multicast can only be used when Member Hosting CSE and Group Hosting CSE have a registration relationship. For example, the Group Hosting IN-CSE can create a multicast group for Member Hosting MN/ASN-CSEs, and cannot create a multicast group for Member Hosting CSEs which are not registered to the IN-CSE.

NOTE 3: The current group based multicast can only be applied between CSEs. Multicast to AEs is FFS.

NOTE 4: The *externalGroupID* is pre-provisioned in the operator's network. The service provider and the operator need ensure that the externalGroupID assigned by the operator matches the *externalGroupID* attribute of each Member Hosting CSE's <*CSEBase*> resource.

The Group Hosting CSE shall create Multicast Group Information locally and establish a mapping relationship between the multicast address and the fanout target according to the member resource identifiers on the Member Hosting CSE. The *multicastGroupFanoutTarget* should be set to / {groupHostingCSE-ID}/ {fanout-segment}. The *multicastGroupFanoutTarget* should be uniquely assigned by the Group Hosting CSE.

{fanout-segment} is a string assigned by the Group Hosting CSE.

In this example, the Multicast Group Information is illustrated in figure 10.2.7.13.1-2.

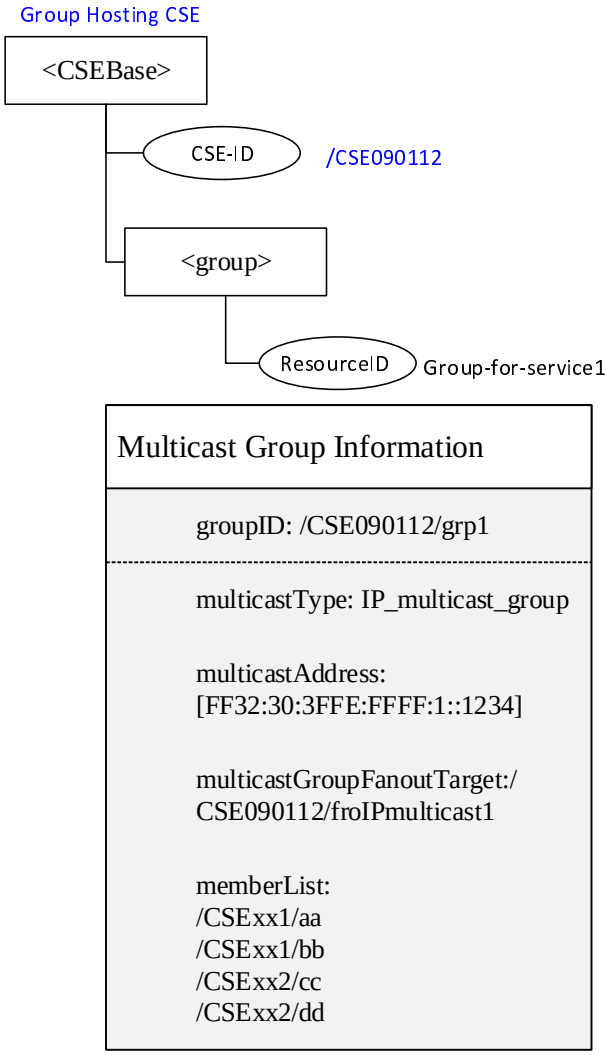


Figure 10.2.7.13.1-2: Multicast Group Information

Step 005: The Group Hosting CSE shall send <*localMulticastGroup*> creation request which carries a mapping relationship between the *multicastGroupFanoutTarget* and the member resources to the Member Hosting CSEs to advertise them to join the multicast group corresponding to the multicast address in unicast mode. If the Group Hosting CSE determines that multiple members resources belong to one Member Hosting CSE according to the IDs of the members resources, it may converge the requests accordingly before sending out. In this example, the Group Hosting CSE should send two group advertisement messages to the two Member Hosting CSEs respectively.

Step 006: The Member Hosting CSEs receive the creation request and shall use a multicast management protocol such as MLD or IGMP to join the multicast group corresponding to the multicast address indicated in the multicast group advertisement.

The Member Hosting CSEs shall create the <*localMulticastGroup*> after successfully joining the multicast group and store a mapping relationship between the *multicastGroupFanoutTarget* and the multicast address, and a mapping relationship between the member list and the multicast address.

In this example, both the ASN-CSE1 and ASN-CSE2 create each <*localMulticastGroup*> resource to record to which multicast group its local member resources have joined.

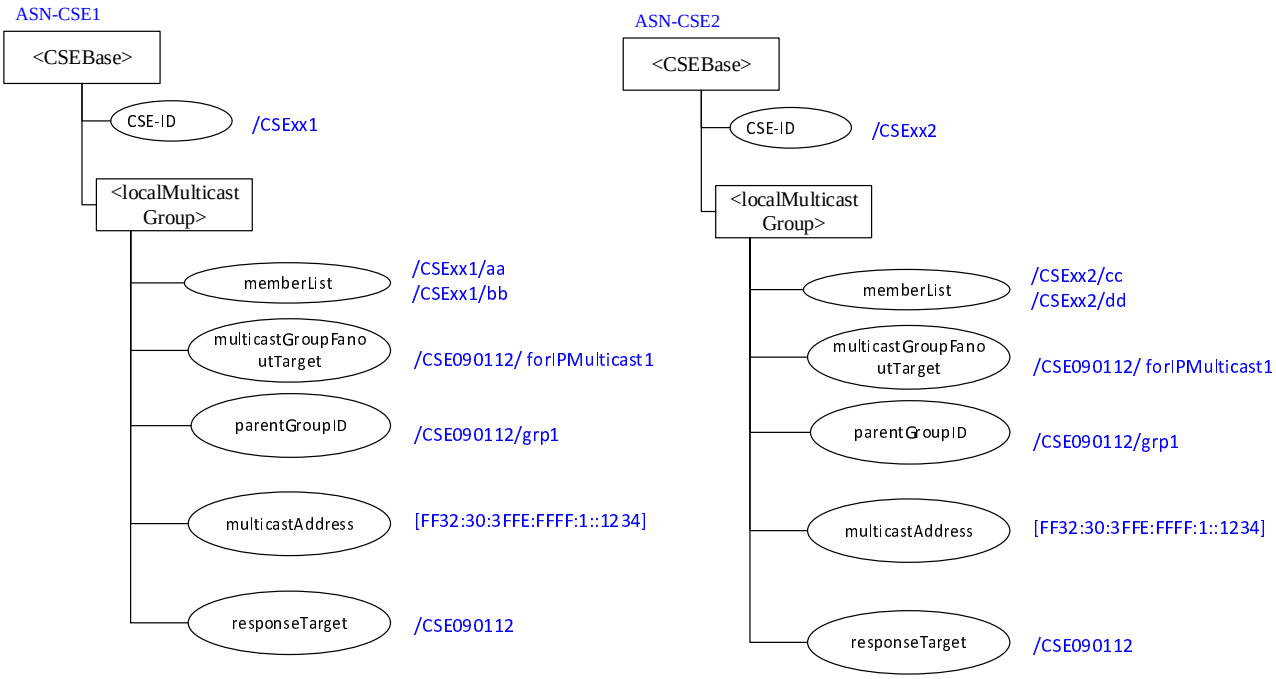


Figure 10.2.7.13.1-3: <localMulticastGroup> resource in Member Hosting CSE

Step 007: The members Hosting CSEs shall send the response to the Group Hosting CSE.

Step 008: The Group Hosting CSE shall check the response messages from Member Hosting CSEs, if at least two members respond successfully, the Group Hosting CSE shall keep Multicast Group Information locally. Otherwise, the Group Hosting CSE shall delete the information.

NOTE 5: The Group Hosting CSE may create one or more sets of Multicast Group Information according to the member resources of the group resource created by the Originator.

10.2.7.13.2 Multicast Group member Fan out Procedures

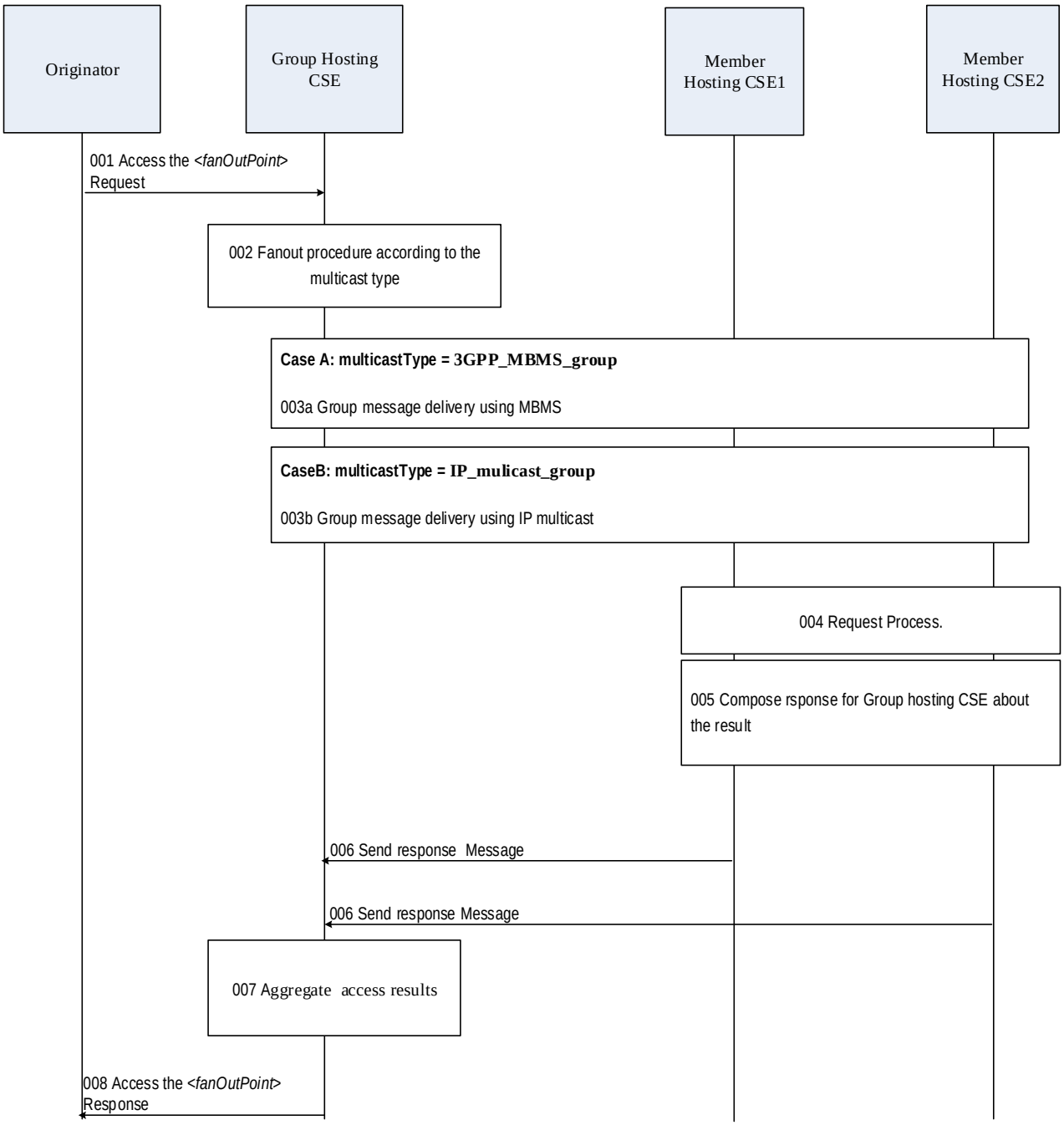


Figure 10.2.7.13.2-1: Multicast Group Information fan out procedures

Step 001: The Originator sends an access <*fanOutPoint*> request carrying the group resource identifier for accessing member resources to the Group Hosting CSE.

Step 002: The Group Hosting CSE shall check whether there is Multicast Group Information, according to the group resource identifier. If the group has Multicast Group Information, check the multicast type:

Case A): The multicast type is 3GPP_MBMS_group.

- **Step 003a:** The Group Hosting CSE shall get the *externalGroupID* based on the *groupID* in the Multicast Group Information, then send request message to the group service Server (e.g. SCEF) to perform TMGI allocation and Group Message request of the 3GPP MBMS group message procedure, as specified in the ETSI TS 118 126 [15].

Case B): The multicast type is IP_multicast_group.

- **Step 003b:** The Group Hosting CSE shall send the member resource access request to the Member Hosting CSEs in multicast mode according to the multicast address of the multicast group, which includes the *multicastGroupFanoutTarget* as the fan out target address corresponding to the member resource in the group resource. If ***Request Expiration Timestamp*** is not in the request from the Originator, the Group Hosting CSE shall set the ***Request Expiration Timestamp*** in the multicast request according to the local policy.

Step 004: The Member Hosting CSE receives from the multicast address a member resource access request, which carries the fan out target matching the *multicastGroupFanoutTarget* of a <*localMulticastGroup*> resource that contains the same multicast address. It shall determine that the local member resource identifiers as the final targets of the request by the mapping relationship between the *memberList* and *multicastGroupFanoutTarget* in the <*localMulticastGroup*> resource, then replaces the *multicastGroupFanoutTarget* with the determined member resource identifiers and executes the operation indicated by the resource access request.

The Member Hosting CSEs shall not return any Acknowledgement when receiving the message from the multicast address.

Step 005: The Member Hosting CSEs shall compose the response message for Group Hosting CSE about the access result: set ***To*** parameter value to the *responseTarget* of <*localMulticastGroup*> resource according to the ***Response Type*** in the request; and set ***From*** parameter value to CSE-ID of Member Hosting CSE.

Step 006: The Member Hosting CSEs shall send the response message including the access results to the Group Hosting CSE.

Step 007: The Group Hosting CSE shall determine a multicast response message according to the ***From*** and the ***Request Identifier*** in message, then aggregate the group member resource access results in the response messages from Member Hosting CSEs.

The Group Hosting CSE shall not return any response after parsing the Notification message content which is response message to the multicast request.

Step 008: The Group Hosting CSE returns the aggregated group member resource access result to the Originator.

10.2.7.14 Create <*localMulticastGroup*>

This procedure shall be used for creating <*localMulticastGroup*> resource.

Table 10.2.7.14-1: <*localMulticastGroup*> CREATE

<*localMulticastGroup*> CREATE	
Information in Request message	***From:*** Identifier of the group Hosting CSE that initiates the Request ***To:*** The address of the <*localMulticastGroup*> resource. ***Content:*** The representation of the <*localMulticastGroup*> resource for which the attributes are described in clause 9.6.44.
Processing at Originator before sending Request	Besides the basic procedures described in clause 10.1.2.The Multicast Group Information shall be created and maintained by the group Hosting CSE. The group Hosting CSE shall configure the group Hosting CSE as the only entity who has the CRUD privileges to the <localMulticastGroup> by configuring the corresponding <accessControlPolicy> resource.
Processing at Receiver	Besides the basic procedures described in clause 10.1.2, the receiver shall also comply with the multicast management protocol such as MLD or IGMP to join the multicast group and create the <localMulticastGroup> resource as specified in clause 10.2.7.13.1.
Information in Response message	According to clause 10.1.2.
Processing at Originator after receiving Response	None
Exceptions	If the Receiver responds with an error, the group Hosting CSE shall delete the receiver's information from the Multicast Group Information maintained locally.

10.2.7.15 Retrieve <*localMulticastGroup*>

This procedure shall be used for retrieving <*localMulticastGroup*> resource.

Table 10.2.7.15-1: <*group*> RETRIEVE

<*group*> RETRIEVE	
Information in Request message	***From:*** Identifier of the CSE that initiates the Request ***To:*** The address of the <*localMulticastGroup*> resource
Processing at Originator before sending Request	The Originator shall request to obtain <*localMulticastGroup*> resource information by using the RETRIEVE operation. The request shall address the specific <*localMulticastGroup*> resource of a Hosting CSE.
Processing at Receiver	No change from the basic procedure in clause 10.1.3
Information in Response message	No change from the basic procedure in clause 10.1.3
Processing at Originator after receiving Response	None
Exceptions	No change from the basic procedure in clause 10.1.3

10.2.7.16 Update <*localMulticastGroup*>

This procedure shall be used for updating <*localMulticastGroup*> resource.

Table 10.2.7.16-1: <*localMulticastGroup*> UPDATE

<*localMulticastGroup*> UPDATE	
Information in Request message	***From:*** Identifier of the CSE that initiates the Request ***To:*** The address the <*localMulticastGroup*> resource ***Content:*** The representation of the <*localMulticastGroup*> resource for which the attributes are described in clause 9.6.44
Processing at Originator before sending Request	According to clause 10.1.4
Processing at Receiver	According to clause 10.1.4
Information in Response message	According to clause 10.1.4
Processing at Originator after receiving Response	None
Exceptions	None

10.2.7.17 Delete <*localMulticastGroup*>

This procedure shall be used for deleting an existing <*localMulticastGroup*> resource.

Table 10.2.7.17-1: <*localMulticastGroup*> DELETE

<*localMulticastGroup*> DELETE	
Information in Request message	***From:*** Identifier of the CSE that initiates the Request ***To:*** The address of the <*localMulticastGroup*> resource
Processing at Originator before sending Request	According to clause 10.1.5
Processing at Receiver	Besides the basic procedures described in clause 10.1.5, the receiver shall also comply with the multicast management protocol such as MLD or IGMP to leave the multicast group
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	None
Exceptions	According to clause 10.1.5.

10.2.8 Device management

10.2.8.1 Introduction

This clause describes the procedures for managing device capabilities on MNs (e.g. M2M Gateways), ASNs and ADNs (e.g. M2M Devices), as well as devices that reside within an M2M Area Network.

Resources maintaining information and relationships that are specific to Device Management are termed Device Management Resources. This clause details the creation, retrieval, update and deletion of the information associated with the following Device Management Resources: <node>, <mgmtObj>, <mgmtCmd> and its child resource <execInstance>.

These operations are used in both Device Management options available in oneM2M: one utilizing existing technology protocols (e.g. BBF TR-069 [i.2], OMA-DM [i.3], and LWM2M [i.4]) and another utilizing the native oneM2M protocols. Clause 6.2.4 details the Device Management (DMG) CSF supporting this functionality.

10.2.8.2 Node management

This clause describes node management procedures over Mca and Mcc reference points, using the <*node*> resource which represents information about M2M Nodes that can be utilized in Device Management and other operations.

M2M Nodes represented by the <node> resource are: MN-CSE, ASN-CSE, ADN and NoDN. Zero, one or more <*node*> resources may be used to represent each M2M Node, as follows.

- A <*node*> resource representing a MN-CSE or a ASN-CSE is hosted by the represented CSE or the registrar CSE. The *hostedCSELink* attribute of the resource allows to find the <CSEBase> or <remoteCSE> resource representing the MN-CSE or ASN-CSE represented by the <node> resource. All <*node*> resources hosted on M2M Node's CSE may be announced to associated IN-CSEs.

- A <*node*> resource representing an ADN is hosted by the registrar CSE. The *hostedAELink* attribute of the resource allows to find the <AE> resources representing the AEs residing on the node ADN.

- A <*node*> resource representing a NoDN is hosted by a CSE with DMG capabilities used to perform Device Management operations on the NoDN. If the NoDN is an interworked device, the *hostedServiceLink* attribute of the resource allows to find the <*flexContainer*> resources representing the services hosted on the NoDN.

An entity co-located with a CSE on an ASN or MN which is managed using oneM2M Device Management shall be represented by the same <*node*> resource

10.2.8.3 Create <*node*>

This procedure shall be used for creating a <*node*> resource.

NOTE: The creation of the <*node*> resource is on discretion of the Originator.

Table 10.2.8.3-1: <*node*> CREATE

<*node*> CREATE	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content:*** The representation of the <node> resource described in clause 9.6.18 The following attributes from clause 9.6.18 are mandatory for the request: • *resourceType* which shall be set to the appropriate tag that identify the <*node*> resource as defined in clause 9.6.1.3
Processing at Originator before sending Request	According to clause 10.1.2
Processing at Receiver	According to clause 10.1.2
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: • ***Content*****:** Address of the created <*node*> resource, according to clause 10.1.2
Processing at Originator after receiving Response	According to clause 10.1.2
Exceptions	According to clause 10.1.2

10.2.8.4 Retrieve <*node*>

This procedure shall be used for retrieving the attributes of a <*node*> resource.

Table 10.2.8.4-1: <*node*> RETRIEVE

<*node*> RETRIEVE	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content*****:** Void
Processing at Originator before sending Request	According to clause 10.1.3
Processing at Receiver	According to clause 10.1.3
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content*****:** Attributes of the <*node*> resource as defined in clause 9.6.18

Processing at Originator after receiving Response	According to clause 10.1.3
Exceptions	According to clause 10.1.3

10.2.8.5 Update <*node*>

This procedure shall be used for updating the attributes and the actual data of a <*node*> resource and its child resources.

Table 10.2.8.5-1: <*node*> UPDATE

<*node*> UPDATE	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content***: attributes of the <*node*> resource as defined in clause 9.6.18 which need be updated, with the exception of the Read Only (RO) attributes cannot be modified
Processing at Originator before sending Request	According to clause 10.1.4
Processing at Receiver	According to clause 10.1.4 with the following: • The Receiver shall check whether the provided attributes of the <node> resource represent a valid request for updating <*node*> resource
Information in Response message	According to clause 10.1.4
Processing at Originator after receiving Response	According to clause 10.1.4
Exceptions	According to clause 10.1.4

10.2.8.6 Delete <*node*>

This procedure shall be used for deleting an existing <*node*> resource.

Table 10.2.8.6-1: <*node*> DELETE

<*node*> DELETE	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.5
Processing at Receiver	According to clause 10.1.5
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	According to clause 10.1.5
Exceptions	According to clause 10.1.5

10.2.8.7 Device management using <*mgmtObj*>

This clause describes the management procedures over Mca and Mcc reference points. If technology specific protocols are used for management, different operations addressing a <*mgmtObj*> resource (or its attributes or child resources) shall be translated by IN-CSE or MN-CSE into technology specific requests performed on the mapped technology specific data model object on the managed entity. In this case, the <mgmtObj> resources are hosted on the IN-CSE or MN-CSE. Although management requests by the AE are agnostic to the technology specific protocol, the <*mgmtObj*> resource exposes information about the technology specific protocol. AEs have the capability to retrieve this information within the *objectIDs* attribute of the <*mgmtObj*> resource.

In the scenario where the <*mgmtObj*> resource does not utilize an external management technology but instead uses the M2M Service Layer to perform the management request, the <*mgmtObj*> resource is hosted on the CSE of the managed entity when the managed entity is an ASN, MN or IN. If the managed entity is an ADN node or the managed entity is co-located on an ASN, MN or IN, the <mgmtObj> resource is hosted on the registrar CSE of the managed entity. The <*mgmtObj*> resource and its parent <*node*> resource hosted on node's CSE may be announced to associated IN-CSEs.

In the scenario where the managed entity is an NoDN, the managed entities' <*mgmtObj*> resources are hosted by a CSE with DMG capabilities used to perform Device Management operations on the NoDN.

10.2.8.8 Create <*mgmtObj*>

This procedure shall be used to create a specific <*mgmtObj*> resource in the Hosting CSE to expose the corresponding management function of a managed entity (i.e. M2M Device/Gateway) over the Mca reference point. Depending on the data model being used, the created <*mgmtObj*> resource may be a partial or complete mapping from the technology specific data model object on the managed entity. If such a technology specific data model object is missing from the managed entity, it shall be added to the managed entity. Further operations performed on the created <*mgmtObj*> resource shall be converted by the Hosting CSE into a corresponding technology specific request performed on the mapped technology specific data model object on the managed entity using technology specific protocol (e.g. OMA-DM [i.3] or BBF TR-069 [i.2]).

Besides the generic create procedure defined in clause 10.1.2, the procedure in table 10.2.8.8-1 shall be used when management is performed using technology specific protocols.

If the management is performed by service layer entities, the procedure is the same as generic create procedure defined in clause 10.1.2. In this case, local APIs (drivers) on the managed entity is required to monitor the change of the <mgmtObj> resource and reflect the change to the managed entity.

Table 10.2.8.8-1: <*mgmtObj*> CREATE

<*mgmtObj*> CREATE	
Information in Request message	***From:*** Identifier of the AE or the CSE that initiates the Request ***To:*** The address of the <*node*> where the <*mgmtObj*> resource is intended to be Created ***Content:*** The representation of the <*mgmtObj*> resource for which the attributes are described in clause 9.6.15
Processing at Originator before sending Request	The Originator shall be an AE, or a CSE: • The Originator is a CSE: In this case, the CSE first collects the original technology specific data model object (the management tree structure or also the value of the tree nodes if needed) of the local device and transforms the object into the <*mgmtObj*> resource representation, then requests the Hosting CSE to create the corresponding <*mgmtObj*> resource. • The Originator is an AE: In this case, the AE requests the Hosting CSE to add the corresponding technology specific data model object to the managed entity by creating an <mgmtObj> resource in the Hosting CSE (See notes 1 and 2)
Processing at Receiver	For the CREATE operation, besides the common create operation defined in clause 10.1.2, the Receiver shall: • If the Originator is an AE: Check if there is existing management session between the management server and the managed entity. If not, request the management server to establish a management session towards the managed entity. Send the technology specific request to the managed entity or to the management server to add the corresponding technology specific data model object to the managed entity based on technology specific protocol • Maintain the mapping relationship between the created <*mgmtObj*> resource and the technology specific data model object on the managed entity • Respond to the Originator with the appropriate responses based on the technology specific response. It shall also provide in the response the address of the created new resource
Information in Response message	Error code if the new technology specific data model object is not created
Processing at Originator after receiving Response	None
Exceptions	• The creation of the technology specific data model object is not allowed • The created technology specific data model object already exists • Corresponding technology specific data model object cannot be added to the managed entity for some reason (e.g. not reachable, memory shortage)
NOTE 1: The CSE can create the <*mgmtObj*> resource locally by itself. The details are out of scope. In this case, the Hosting CSE first collects the original technology specific data model object on the managed entity via technology specific protocol (e.g. OMA DM [i.3], BBF TR-069 [i.2] or LWM2M [i.4]), then	

	transforms the object into the *<mgmtObj>* resource representation and create the *<mgmtObj>* resource locally in the CSE.
NOTE 2:	The *<mgmtObj>* resource can be created in the Hosting CSE by other offline provisioning means which are out of scope.

10.2.8.9 Retrieve *<mgmtObj>*

This procedure shall be used to retrieve information from an existing *<mgmtObj>* resource. Besides the generic retrieve procedure defined in clause 10.1.3, the procedure in table 10.2.8.9-1 shall be used when management is performed using technology specific protocols. If the management is performed by service layer entities, the procedure is the same as generic retrieve procedure defined in clause 10.1.3.

Table 10.2.8.9-1: *<mgmtObj>* RETRIEVE

***<mgmtObj>* RETRIEVE**	
Information in Request message	***From:*** Identifier of the AE or the CSE that initiates the Request ***To:*** The address of the *<mgmtObj>* resource
Processing at Originator before sending Request	None
Processing at Receiver	For the RETRIEVE operation, besides the common retrieve operation defined in clause 10.1.3, the Receiver shall: • If the Originator is an AE and if the requested information of the *<mgmtObj>* resource is not available, identify the corresponding technology specific data object on the managed entity according to the mapping relationship that the CSE maintains. Check if there is an existing management session between the management server and the managed entity. If not, request the management server to establish a management session towards the managed entity. Send the technology specific request to get the corresponding technology specific data model object from the managed entity based on the external management technology, then return the result to the Originator based on the technology specific response
Information in Response message	Error code if the new technology specific data model object cannot be retrieved
Processing at Originator after receiving Response	None
Exceptions	• Corresponding technology specific data model object data cannot be retrieved from the managed entity (e.g. technology specific data model object not found)

10.2.8.10 Update *<mgmtObj>*

This procedure shall be used to update information of an existing *<mgmtObj>* resource. Besides the generic update procedure defined in clause 10.1.4, the procedure in table 10.2.8.10-1 shall be used when management is performed using technology specific protocol. If the management is performed by service layer entities, the procedure is the same as generic update procedure defined in clause 10.1.4. In this case, local APIs (drivers) on the managed entity is required to monitor the change of the <mgmtObj> resource and reflect the change to the managed entity.

Table 10.2.8.10-1: *<mgmtObj>* UPDATE

***<mgmtObj>* UPDATE**	
Information in Request message	***From:*** Identifier of the AE or the CSE that initiates the Request ***To:*** The address of the *<mgmtObj>* resource ***Content:*** The representation of the *<mgmtObj>* resource for which the attributes are described in clause 9.6.15
Processing at Originator before sending Request	None
Processing at Receiver	For the UPDATE operation, besides the common update operation defined in clause 10.1.4, the Receiver shall: • If the Originator is an AE, identify the corresponding technology specific data model object on the managed entity according to the mapping relationship it maintains. Check if there is an existing management session between the management server and the managed entity. If not, request the management server to establish a management session towards the managed entity. Send

***<mgmtObj>* UPDATE**	
	the technology specific request to update the corresponding technology specific data model object in the managed entity accordingly based on technology specific protocol • Respond to the Originator with the appropriate response based on the technology specific response from the external management technology
Information in Response message	Error code if the technology specific data model object cannot be updated
Processing at Originator after receiving Response	None
Exceptions	• Corresponding technology specific data model object cannot be updated to managed entity (e.g. not reachable, technology specific data model object not found)

10.2.8.11 Delete <*mgmtObj*>

This procedure shall be used to delete an existing <*mgmtObj*> resource. An Originator uses this procedure to remove the corresponding technology specific data model object (e.g. an obsolete software package) from the managed entity. Besides the generic delete procedure defined in clause 10.1.5, the procedure in table 10.2.8.11-1 shall be used when management is performed using external management technologies. If the management is performed by service layer entities, the procedure is the same as generic delete procedure defined in clause 10.1.5. In this case, local APIs (drivers) on the managed entity is required to monitor the change of the <mgmtObj> resource and reflect the change to the managed entity.

Table 10.2.8.11-1: *<mgmtObj>* DELETE

***<mgmtObj>* DELETE**	
Information in Request message	***From:*** Identifier of the AE, or the CSE that initiates the Request ***To:*** The address of the <*mgmtObj*> resource
Processing at Originator before sending Request	The Originator shall be an AE or CSE: • The Originator is a CSE: In this case, the CSE issues the request to the Hosting CSE to hide the corresponding management function from being exposed by the <*mgmtObj*> resource • The Originator is an AE: In this case, the AE requests the Hosting CSE to delete the <*mgmtObj*> resource from the Hosting CSE and to remove the corresponding technology specific data model object from the managed entity (See notes 1 and 2)
Processing at Receiver	For the DELETE operation, besides the common create operation defined in clause 10.1.5, the Receiver shall: • If the Originator is an AE, identify the corresponding technology specific data model object on the managed entity according to the mapping relationship the CSE maintains. Check if there is an existing management session between the management server and the managed entity. If not, request the management server to establish a management session towards the managed entity. The CSE sends technology specific request to remove the corresponding technology specific data model object from the managed entity based on technology specific protocol • Respond to the Originator with the appropriate generic responses based on the technology specific response
Information in Response message	Error code if the technology specific data model object cannot be deleted
Processing at Originator after receiving Response	None
Exceptions	• Corresponding technology specific data model object cannot be deleted from managed entity (e.g. not reachable, technology specific data model object not found)
NOTE 1: The Hosting IN-CSE can delete the <*mgmtObj*> resource locally by itself. This internal procedure is out of scope. NOTE 2: The <*mgmtObj*> resource can be deleted in the Hosting CSE by offline provisioning means which are out of scope.	

10.2.8.12 Execute <*mgmtObj*>

This procedure shall be used to execute a technology specific requests on a managed entity through an existing <*mgmtObj*> resource on the Hosting CSE.

Table 10.2.8.12-1: <*mgmtObj*> EXECUTE

<*mgmtObj*> EXECUTE	
Information in Request message	***From:*** Identifier of the AE, or the CSE that initiates the Request ***To:*** The address of the <*mgmtObj*> resource
Processing at Originator before sending Request	The Originator shall be an AE or a CSE. The Originator shall request to execute a management command which is represented by a <*mgmtObj*> resource or its attribute by using an UPDATE operation The request shall address the executable <*mgmtObj*> resource. For an execute operation on an attribute(s), the ***Content*** parameter shall be included with the name of such attribute(s) with predefined value(s) to trigger the respective action After the execution request, the Originator shall request to retrieve the execution result or status from the executable <*mgmtObj*> resource or its attribute/child resource by using a RETRIEVE operation as specified in clause 10.2.8.9
Processing at Receiver	For the EXECUTE operation, the Receiver shall: • Check if the Originator has the WRITE privilege on the addressed <*mgmtObj*> resource or its attribute • Check if there is an existing management session between the management server and the managed entity. If not, request the management server to establish a management session towards the managed entity. Send the technology specific request to execute the corresponding management command (e.g. "Exec" in OMA DM [i.3]) on the managed entity based on technology specific protocol • Respond to the Originator with the appropriate response based on the technology specific response. If available, the technology specific response shall contain execution results • Retrieve the execution result or status from the executable <*mgmtObj*> resource or its attribute, perform the procedures as described in clause 10.2.8.9 • Upon receiving a management notification (e.g. OMA-DM [i.3] "Generic Alert" message or BBF TR-069 [i.2] "Inform" message) from a managed entity regarding the execution result or status, the Receiver shall send the technology specific request to retrieve the execution result or status of the technology specific data model object information received from the managed entity and update the corresponding <*mgmtObj*> resource or its attribute
Information in Response message	Error code if the technology specific request cannot be executed
Processing at Originator after receiving Response	None
Exceptions	• Corresponding technology specific request cannot be executed in managed entity (e.g. not reachable, technology specific data model object not found)

10.2.8.13 Device management using <*mgmtCmd*> and <*execInstance*>

This clause describes how RESTful management operations may be performed using <*mgmtCmd*> resources over the Mca and Mcc reference points. The <*mgmtCmd*> resource, together with its attributes or sub-resources, may be used in the process of translating between RESTful operations and management commands and procedures from existing management technologies (e.g. BBF TR-069 [i.2]). These procedures can then be performed on the managed entity, using the Management Adapter and the procedures described in the following clauses.

10.2.8.14 Create *<mgmtCmd>*

A CREATE request shall be used by an Originator to create a specific *<mgmtCmd>* resource in a Hosting CSE.

The created <mgmtCmd> resource will be mapping a RESTful method to management commands and/or procedures which may be translated from existing management protocols (e.g. BBF TR-069 [i.2]). At run-time the Hosting CSE can expose the translated commands, over the Mcc reference point, to the managed entities (i.e. ASN/MN-CSE).

The Originator may be:

- An AE registered to the IN-CSE.
- The CSE on the managed entity: In this case, the CSE transforms supported management command into the *<mgmtCmd>* resource representation, then requests the Hosting CSE to create the corresponding *<mgmtCmd>* resource.

NOTE 1: The Hosting IN-CSE in the network domain may also create the *<mgmtCmd>* resource locally by itself. The details are out of scope. Then an AE can discover the created *<mgmtCmd>* and manipulate it.

NOTE 2: The *<mgmtCmd>* resource could also be created in the Hosting CSE by other offline provisioning means which are out of scope.

The Receiver shall be an IN-CSE.

Table 10.2.8.14-1: *<mgmtCmd>* CREATE

***<mgmtCmd>* CREATE**	
Information in Request message	The attributes of the *<mgmtCmd>* resource. The mandatory and/or optional attributes defined in clause 9.6.16, as needed
Processing at Originator before sending Request	According to clause 10.1.2 with the following: • The CSE on the originating node shall first collect local management command
Processing at the Receiver	According to clause 10.1.2 with the following: • The Receiver CSE shall maintain the mapping between the created *<mgmtCmd>* resource and the corresponding nonRESTful commands represented by the *cmdType* attribute of *<mgmtCmd>* resource
Information in Response message	According to clause 10.1.2 with the following specific information: • ***Content:*** Address of created *<mgmtCmd>* resource
Processing at Originator after receiving Response	According to clause 10.1.2
Exceptions	According to clause 10.1.2

10.2.8.15 Retrieve *<mgmtCmd>*

This procedure shall be used for retrieving all or part information from a previously created *<mgmtCmd>* resource on a target CSE.

The Originator may be:

- An AE.
- A CSE.

The Receiver shall be an IN-CSE.

Table 10.2.8.15-1: *<mgmtCmd>* RETRIEVE

***<mgmtCmd>* RETRIEVE**	
Information in Request message	According to clause 10.1.3, with the mandatory and/or optional attributes defined in clause 9.6.16, as needed
Processing at Originator before sending Request	According to clause 10.1.3
Processing at Receiver	According to clause 10.1.3
Information in Response message	According to clause 10.1.3
Processing at Originator after receiving Response	According to clause 10.1.3
Exceptions	According to clause 10.1.3

10.2.8.16 Update <*mgmtCmd*>

This procedure shall be used for updating some of the attributes (other than *execEnable*) of an existing <*mgmtCmd*> resource with new attribute values. An UPDATE method applied to the *execEnable* attribute is used to trigger the execution of the management procedure represented by <*mgmtCmd*>, as described in clause 10.2.8.18.

The Originator may be:

- An AE.
- A CSE.

The Receiver shall be an IN-CSE.

Table 10.2.8.16-1: *<mgmtCmd>* UPDATE

***<mgmtCmd>* UPDATE**	
Information in Request message	According to clause 10.1.4, including mandatory and/or optional attributes defined in clause 9.6.16, as needed
Processing at Originator before sending Request	According to clause 10.1.4
Processing at Receiver	According to clause 10.1.4
Information in Response message	According to clause 10.1.4
Processing at the Originator after receiving Response	According to clause 10.1.4
Exceptions	According to clause 10.1.4

10.2.8.17 Delete <*mgmtCmd*>

This procedure shall be used for deletion of an existing <*mgmtCmd*> resource on a Hosting CSE. An AE may also use this procedure to cancel any initiated <*execInstance*> of an <*mgmtCmd*> if applicable.

The Originator may be:

- The CSE on the manageable entity: In this case, the CSE issues the request to the Hosting CSE to hide the corresponding management command from being exposed by the <*mgmtCmd*> resource.
- An AE: In this case, the AE requests the Hosting CSE to delete the <*mgmtCmd*> resource from the Hosting CSE and cancel all initiated <*execInstance*> of an <*mgmtCmd*> if applicable.

NOTE 1: The Hosting CSE in the network domain could also delete an <*mgmtCmd*> resource locally by itself. This internal procedure is out of scope.

NOTE 2: The <*mgmtCmd*> resource could also be deleted in the Hosting CSE by other offline provisioning means which are out of scope.

If the Originator is an AE and there is any initiated <*execInstance*> under the <*mgmtCmd*> that can be cancelled by a corresponding management command. The Hosting CSE shall also issue the management command to the managed entity to cancel those initiated <*execInstance*> based on existing management protocol (i.e. BBF TR-069 [i.2]). Then the CSE shall respond to the Originator with the appropriate generic responses.

The Receiver shall be an IN-CSE.

Table 10.2.8.17-1: *<mgmtCmd>* DELETE by ASN-CSE or MN-CSE

***<mgmtCmd>* DELETE by ASN-CSE or MN-CSE**	
Information in Request message	According to clause 10.1.5
Processing at Originator before sending Request	According to clause 10.1.5 with the following: • Before issuing a DELETE request to the IN-CSE, the originating CSE may perform cancelling of the corresponding management command locally
Processing at Receiver	According to clause 10.1.5 with the following: • The Receiver IN-CSE shall verify if there are any initiated <*execInstance*> commands under the <*mgmtCmd*> which are cancellable by using a corresponding management command. If there are, the Receiver IN-CSE shall issue the management command to the managed entity to cancel those initiated <*execInstance*> based on existing management protocol (i.e. BBF TR-069 [i.2]) • The <*mgmtCmd*> resource shall be deleted from the repository of the Receiver IN-CSE • Then the Receiver IN-CSE shall respond to the Originator ASN-CSE or MN-CSE with the appropriate responses
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	According to clause 10.1.5
Exceptions	According to clause 10.1.5 with the following: • If the deletion is not allowed or the specific <*mgmtCmd*> resource does not exist, there is no local processing in the Receiver IN-CSE and a proper error code shall be returned to the Originator ASN-CSE or MN-CSE • If the corresponding initiated commands cannot be deleted from the managed entity due to some reason (e.g. not found) a response with the proper indication shall be returned to the Originator ASN-CSE or MN-CSE

Table 10.2.8.17-2: *<mgmtCmd>* DELETE by an AE

***<mgmtCmd>* DELETE by an AE**	
Information in Request message	According to clause 10.1.5
Processing at the Originator before sending Request	According to clause 10.1.5
Processing at Receiver	According to clause 10.1.5 with the following: • If there is any initiated *<execInstance>* under *<mgmtCmd>* and it is cancellable, the Receiver IN-CSE shall cancel those initiated *<execInstance>* from the managed entity using corresponding management procedures in existing management protocol (i.e. CancelTransfer RPC in BBF TR-069 [i.2]) The *<mgmtCmd>* resource shall be deleted from the repository of the Receiver IN-CSE
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	According to clause 10.1.5
Exceptions	According to clause 10.1.5 with the following: • If the deletion is not allowed or the specific *<mgmtCmd>* resource does not exist, there is no local processing in the Receiver IN-CSE and a proper error code shall be returned to the Originator AE • If the corresponding initiated commands cannot be deleted from managed entity due to some reason (e.g. not found) a response with the proper indication shall be returned to the Originator AE

10.2.8.18 Execute *<mgmtCmd>*

The Execute procedure shall be used by an Originator in order to trigger execution of a specific management command on a managed entity, by employing an UPDATE method to the *execEnable* attribute of an existing *<mgmtCmd>* resource on the Hosting CSE.

The Originator shall be an AE.

The Receiver shall be an IN-CSE.

Table 10.2.8.18-1: *<mgmtCmd>* EXECUTE

***<mgmtCmd>* EXECUTE**	
Information in Request message	According to clause 10.1.4, with the following (see attributes defined in clause 9.6.16): • The UPDATE request shall address the *execEnable* attribute with a predefined value to trigger the EXECUTE action
Processing at the Originator before sending Request	According to clause 10.1.4, with the following: After issuing the execution request, the Originator may request to retrieve the execution result or status from <*execInstance*> sub-resources of the <*mgmtCmd*>by using a RETRIEVE method as described in clause 10.2.8.20
Processing at the Receiver	According to clause 10.1.4 with the following: • The Receiver shall check if the Originator has the UPDATE privilege on the addressed <*mgmtCmd*> resource. Upon successful validation, the Hosting CSE shall perform command conversion and mapping, and send the converted management command to execute with the provided arguments on the remote entity based on existing device management protocol (i.e. BBF TR 069 [i.2]) • Then the Hosting CSE shall create for each target a corresponding <*execInstance*> resource under <*mgmtCmd*> and shall respond to the Originator with the appropriate generic responses. It shall also provide in the response the URL of the created <*execInstance*> resource • If the *execTarget* attribute of the addressed <*mgmtCmd*> addresses a group, the Hosting CSE shall create corresponding <*execInstance*> resources for each target in the group and provide the corresponding URLs in the response Upon receiving from any remote entity, a management notification (i.e. BBF TR-069 [i.2] "Inform" message) regarding the execution result or status, the Hosting CSE may update the corresponding <*execInstance*> sub-resource locally
Information in Response message	According to clause 10.1.4
Processing at Originator after receiving Response	According to clause 10.1.4, with additional processing which is dependent on the type of the command and execution status. The following actions may occur in any order after the command execution is finished: • The managed entity may send responses including execution results to the Receiver CSE, who will store the execution results in corresponding <*execInstance*> resource • The Originator AE may use normal RETRIEVE procedure to retrieve the execution results or status of an <*execInstance*>*.* After receiving a RETRIEVE request from the Originator AE, the Receiver CSE can retrieve the execution status or results on the managed entity using existing management protocol • A response shall be returned to the Originator AE
Exceptions	• If the execution is not allowed or the specified <*mgmtCmd*> resource does not exist, no further processing is required on the Receiver CSE, and a proper error code shall be returned to the Originator AE in the message response • If the corresponding management command cannot be executed on the managed entity, an error code shall be returned with the response to Originator AE

10.2.8.19 Cancel <*execInstance*>

The Cancel procedure shall be used by an originating AE to disable/stop/cancel an initiated management command execution on the remote entity, through an UPDATE method to the *execDisable* attribute of an existing <*execInstance*> resource on the Hosting CSE.

The Originator shall be an AE.

The Receiver shall be an IN-CSE.

Table 10.2.8.19-1: *<execInstance>* CANCEL

***<execInstance>* CANCEL**	
Information in Request message	According to clause 10.1.4, with the following (see attributes defined in clause 9.6.17): The UPDATE request shall address the *execDisable* attribute with a predefined value in order to trigger the CANCEL action
Processing at the Originator before sending Request	Originator needs to disable/stop/cancel an initiated management command execution on the managed entity using an *<execInstance>* sub-resource at the Receiver, by using an UPDATE operation See also clause 10.1.4
Processing at Receiver	The Receiver shall check if the Originator has the UPDATE privilege on the addressed *<execInstance>* resource Then, the Receiver shall check if the management operation is initiated and cancellable. Upon successful validation, the Receiver IN-CSE shall perform command conversion and mapping, then use existing management protocol (i.e. BBF TR-069 [i.2]) to cancel the corresponding management command execution initiated on the managed entity The Receiver IN-CSE shall respond to the Originator with the appropriate responses
Information in Response message	According to clause 10.1.4
Processing at Originator after receiving ResponsePost-	According to clause 10.1.4
Exceptions	• If the *<execInstance>* has not been initiated, is already complete or it is not cancellable, or the specified *<execInstance>* resource does not exist in the Receiver IN-CSE, the post processing on Receiver CSE shall be skipped and a proper error code shall be returned to Originator in the Response message

10.2.8.20 Retrieve *<execInstance>*

This procedure shall be used for retrieving all or part information from an *<execInstance>* resource on a target CSE.

The Originator shall be an AE.

The Receiver shall be an IN-CSE.

Table 10.2.8.20-1: *<execInstance>* RETRIEVE

***<execInstance>* RETRIEVE**	
Information in Request message	According to clause 10.1.3, with the mandatory and/or optional attributes defined in clause 9.6.17, as needed
Processing at the Originator before sending Request	Originator needs to create a resource
Processing at Receiver	According to clause 10.1.3, with the following: • If the retrieval is allowed, the Receiver IN-CSE can retrieve the execution status or results on the managed entity using existing management protocol (i.e. BBF TR-069 [i.2]) • If the retrieval is allowed, the addressed attributes of the *<execInstance>* resource shall be retrieved from the repository of the Receiver IN-CSE
Information in Response message	According to clause 10.1.3
Processing at Originator after receiving Response	According to clause 10.1.3
Exceptions	If the retrieval is not allowed or the specific *<execInstance>* resource does not exist in the Receiver IN-CSE, there is no local processing on the Receiver CSE and a proper error code shall be returned to Originator AE in the Response Message

10.2.8.21 Delete <*execInstance*>

The DELETE request procedure shall be used by an originating AE to delete an existing <*execInstance*> resource on a Receiver IN-CSE.

The Originator shall be an AE.

NOTE 1: The Receiver IN-CSE in the network domain could also delete an <*execInstance*> resource locally by itself. This internal procedure is out of scope.

NOTE 2: The <*execInstance*> resource could also be deleted in the Receiver IN-CSE by other offline provisioning means which are out of scope.

Receiver: The Receiver shall check if the Originator has the DELETE permission on the addressed <*execInstance*> resource. Upon successful validation, the Hosting CSE shall remove the resource from its repository. If a corresponding management command has been initiated and is pending finished on the managed entity and the management command is cancellable, the Hosting CSE shall use existing management protocols (i.e. BBF TR-069 [i.2] CancelTransfer RPC) to cancel the corresponding management currently initiated at the managed entity. Then the CSE shall respond to the Originator with the appropriate generic responses.

The Hosting CSE shall be an IN-CSE.

Table 10.2.8.21-1: <*execInstance*> DELETE

<*execInstance*> DELETE	
Information in Request message	According to clause 10.1.5
Processing at the Originator before sending Request	According to clause 10.1.5
Processing at Receiver	According to clause 10.1.5 with the following: • If the <*execInstance*> has not been initiated, is already complete or it is not cancellable, the <*execInstance*> resource shall be deleted from the repository of the IN-CSE • If the <*execInstance*> is pending and it is cancellable, the Receiver IN-CSE shall first cancel the <*execInstance*> from the managed entity using corresponding management procedures in existing management protocol (i.e. CancelTransfer RPC in BBF TR-069 [i.2]). Afterwards, the <*execInstance*> resource shall be deleted from the repository of the Receiver IN-CSE If the corresponding initiated commands cannot be successfully cancelled on the managed entity for some reason, the <*execInstance*> resource shall be still deleted Then the Receiver IN-CSE shall respond to the Originator with the appropriate generic responses
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	According to clause 10.1.5
Exceptions	If the deletion is not allowed or the specific <*execInstance*> resource does not exist on the Receiver IN-CSE, there is no processing at the Receiver and a proper error code shall be returned to the Originator

10.2.9 Location management

10.2.9.1 Introduction

This clause introduces the procedures for obtaining and managing a target M2M Node's location information, which are associated with the <*locationPolicy*> resource that contains the method for obtaining and managing location information. Since the actual location information of a target M2M Node shall be stored in the <*contentInstance*> resource as per the configuration described in the associated <*locationPolicy*> resource, this clause introduces the procedures related to the <*contentInstance*> and <*container*> resource.

10.2.9.2 Create <locationPolicy>

This procedure shall be used for creating a <*locationPolicy*> resource.

Table 10.2.9.2-1: *<locationPolicy>* CREATE

***<locationPolicy>* CREATE**	
Information in Request message	***From*:** Identifier of the AE or the CSE that initiates the Request ***To*:** the address of the <*CSEBase*> resource ***Content*:** The representation of the <*locationPolicy*> resource described in clause 9.6.10
Processing at Originator before sending Request	According to clause 10.1.2
Processing at Receiver	• Check whether the Originator is authorized to request the procedure • Check whether the provided attributes of the <*locationPolicy*> resource represent a valid Request • Upon successful validation of the above procedures, the Hosting CSE creates a <*container*> resource and a <*locationPolicy*> resource. Both of these resources shall be hosted locally on the Hosting CSE. The Hosting CSE shall maintain cross-references between both resources: *locationContainerID* attribute for <*locationPolicy*> resource and *locationID* attribute for <*container*> resource. • Check the defined *locationSource* attribute to determine which method is used. The *locationSource* attribute shall be set based on the capabilities of a target M2M Node, the required location accuracy of the Originator and the Underlying Network in which a target M2M Node resides: - For the Network-based case, if the *locationServer* is absent in the Originator's request the Hosting CSE shall either derive the *locationServer* value from the *locationTargetID* or be pre-provisioned with the identity of a locationServer. The Hosting CSE shall transform the request, from the Originator into a Location Server request that includes the following attributes *locationTargetID*, *locationServer,* authID that are defined in the <*locationPolicy*> resource. Additionally, the Hosting CSE shall also provide default values for other parameters (e.g. required quality of position) in the Location Server request [i.5] according to local policies. If the request which requests the location information of the target device towards the Location Server crosses over the Mcn reference point, then the Location Server in the Underlying Network verifies whether the external entity is authorized to request the location information, and only if the AE is permitted, the Location server performs positioning procedures, and returns the successful results over the Mcn reference point. - The specific mechanism used to communicate with the network Location Server depends on the capabilities of the Underlying Network and other factors. For example, it could be either the OMA Mobile Location Protocol [i.5] or OMA RESTful NetAPI for Terminal Location [i.31]. Check the assigned *locationInformationType* attribute and if the value of this attribute is *Geo-fence event*, following the steps below: The Hosting CSE shall check the target Node's capability (Positionable, Non-Positionable or both) by retrieving the stored <node> resource or <mgmtCmd> procedure(e.g. checking the Node's capability through RPC-based procedure) and the Hosting CSE shall create <mgmtCmd> resource type with appropriate configuration based on the node capability and attributes stored in the created <locationPolicy> resource (e.g. *locationUpdatePeriod* attribute of <locationPolicy> to *execFrequency* attribute of <mgmtCmd>) to obtain the Geo-fence relevant information (e.g. measurement or position fix) from the target Node. The node shall respond the information and the Hosting CSE shall create <execInstance> resource type as a placeholder for the information. The Hosting CSE shall forward this information to Geo-Fence Server (refer to locationServer attribute) and returns the results (e.g. event type) over the *Mcn* reference point. The result shall be stored in the created <container> resource as explained in clause 10.2.9.6. (See note).

***<locationPolicy>* CREATE**	
	- For the Device-based case, this case is applicable if the Originator is ASN-AE and the ASN has location determination capabilities (e.g. GPS). The Hosting CSE is capable of performing positioning procedure using the module or technologies. For example, if the ASN has a GPS module itself, the ASN-CSE obtains the location information of Node from the GPS module through internal interfaces (e.g. System call or JNI [i.18]). The detail procedure is out-of-scope. - For the Sharing-based case, this case shall be applicable if the Originator is an ADN-AE and the Hosting CSE is MN CSE and the ADN is a resource constrained node, no location determination capabilities (e.g. GPS) and Network-based positioning capabilities. Also according to the required location accuracy of the AE, the Originator may choose this case. When the Hosting CSE receives the CREATE request and if the Hosting CSE can find the closest Node that is registered with the Hosting CSE and has location information from the Originator in the M2M Area Network, the location information of the closest Node shall be stored as the location information of the Originator, or if the Hosting CSE cannot find any closest Node or has no topology information, the location information of the Node of the Hosting CSE (MN) shall be stored as the location information of the Originator. The closest Node can be determined by the minimum hop based on the topology information stored in the *<node>* resource.
Information in Response message	The representation of the created *<locationPolicy>* resource
Processing at Originator after receiving Response	According to clause 10.1.2
Exceptions	No change from the generic procedure
NOTE: The details of the mechanisms are addressed in the ETSI TS 118 104 [3].	

10.2.9.3 Retrieve <locationPolicy>

This procedure shall be used for retrieving an existing <*locationPolicy*> resource.

Originator: The Originator shall request to obtain <*locationPolicy*> resource information by using RETRIEVE operation. The Originator is either an AE or a CSE.

Receiver: The Receiver shall check if the Originator has RETRIEVE permission on the <*locationPolicy*> resource. Upon successful validation, the Hosting CSE shall respond to the Originator with the appropriate responses.

Table 10.2.9.3-1: *<locationPolicy>* RETRIEVE

***<locationPolicy>* RETRIEVE**	
Information in Request message	***From*:** Identifier of the AE or the CSE that initiates the Request ***To*:** The address of the target *<locationPolicy>* resource
Processing at Originator before sending Request	None
Processing at Receiver	According to clause 10.1.3
Information in Response message	According to clause 10.1.3
Processing at Originator after receiving Response	None
Exceptions	According to clause 10.1.3

10.2.9.4 Update <locationPolicy>

This procedure shall be used for updating an existing <*locationPolicy*> resource.

Originator: The Originator shall request to update attributes of an existing <*locationPolicy*> resource by using an UPDATE operation. The request shall address the specific <*locationPolicy*> resource of a CSE. The Originator may be either an AE or a CSE.

Receiver: The Receiver of an UPDATE request shall check whether the Originator is authorized to request the operation. The receiver shall further check whether the provided attributes of the <*locationPolicy*> resource represent a valid request for updating <*locationPolicy*> resource.

Table 10.2.9.4-1: *<locationPolicy>* UPDATE

***<locationPolicy>* UPDATE**	
Information in Request message	***From*:** Identifier of the AE or the CSE that initiates the Request ***To*:** The address of the target *<locationPolicy>* resource ***Content*:** The attributes which are to be updated
Processing at Originator before sending Request	None
Processing at Receiver	According to clause 10.1.4 with the following: • If the value of *locationUpdatePeriod* attribute is updated to 0 or NULL, the Hosting CSE shall stop periodical positioning procedure and perform the procedure when Originator retrieves the <latest> resource of the linked <container> resource. See clause 10.2.9.6 and clause 10.2.9.7 for more detail • If the value of *locationUpdatePeriod* attribute is updated to bigger than 0 (e.g. 1 hour) from 0 or NULL, the Hosting CSE shall start periodical positioning procedure
Information in Response message	According to clause 10.1.4
Processing at Originator after receiving Response	None
Exceptions	According to clause 10.1.4

10.2.9.5 Delete <*locationPolicy*>

This procedure shall be used for deleting an existing <*locationPolicy*> resource.

Originator: The Originator shall request to delete an existing <*locationPolicy*> resource by using the DELETE operation. The Originator may be either an AE or a CSE. This request can be occurred when the *locationSource* attribute of the created <*locationPolicy*> resource is "sharing-based" and the Originator is an AE that disconnects from the registered MN-CSE.

Receiver: The Receiver shall check if the Originator has DELETE permission on the <*locationPolicy*> resource. Upon successful validation, the CSE shall remove the resource from its repository and shall respond to the Originator with appropriate responses.

Table 10.2.9.5-1: *<locationPolicy>* DELETE

***<locationPolicy>* DELETE**	
Information in Request message	***From*:** Identifier of the AE or the CSE that initiates the Request ***To*:** the address of the target *<locationPolicy>* resource
Processing at Originator before Sending Request	None
Processing at Receiver	According to clause 10.1.5
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	Once the *<locationPolicy>* resource is deleted, the Receiver shall delete the associated resources (i.e. *<container>, <contentInstance>* resources). If the *locationSource* attribute and the *locationUpdatePeriod* attribute of the *<locationPolicy>* resource has been set with appropriate value, the Receiver shall tear down the session. The specific mechanism used to tear down the session depends on the support of the Underlying Network and other factors
Exceptions	According to clause 10.1.5

10.2.9.6 Procedure for <container> resource that stores the location information

This procedure is mainly triggered by the creation of <*locationPolicy*> resource. Based on the defined attributes related to the <*container*> resource such as '*locationContainerID*' and '*locationContainerName*', the Hosting CSE shall create <*container*> resource to store the location information in its child resource, <*contentInstance*> resource after the CSE obtains the actual location information of a target M2M Node. If the Originator provides the *'locationContainerName'* and the given *'locationContainerName'* does not exist in the Hosting CSE, the Hosting CSE shall set the *'resourceName'* of the created <*container*> resource to the *'locationContainerName'* provided by the Originator. If the given *'locationContainerName'* already exists in the Hosting CSE, the Hosting CSE shall respond with an error following the general exceptions written in clause 10.1.2. If the Originator does not provide the *'locationContainerName'* the Hosting CSE shall provide *'resourceName'* for the created <*container*> resource. After the creation of the <*container*> resource, the *resourceID* attribute of the resource shall be stored in the '*locationContainerID*'.

10.2.9.7 Procedure for *<contentInstance>* resource that stores location information

After the <*container*> resource that stores the location information is created, each instance of location information shall be stored in the different <*contentInstance*> resources. In order to store the location information in the <*contentInstance*> resource, the Hosting CSE firstly checks the defined *locationUpdatePeriod* attribute. If a valid period value is set for this attribute, the Hosting CSE shall perform the positioning procedures as defined by *locationUpdatePeriod* in the associated <*locationPolicy*> resource and stores the results (e.g. position fix and uncertainty) in the <*contentInstance*> resource under the created <*container*> resource. However, if no value (e.g. null or zero) is set and locationUpdateEventCriteria is absent, the positioning procedure shall be performed when an Originator requests to retrieve the <latest> resource of the <container> resource and the result shall be stored as a <*contentInstance*> resource under the <container> resource.

10.2.10 Subscription and notification

10.2.10.1 Introduction

An Originator may create a <*subscription*> resource as a child resource of a subscribed-to resource on a Hosting CSE in order to instruct the Hosting CSE to send notifications to the Subscriber(s) of the subscribed-to resource when the subscribed-to resource is modified. After successful <*subscription*> resource creation, the Hosting CSE shall notify the Subscriber(s) of a modification of the subscribed-to resource that meets conditions configured in the <*subscription*> resource.

A subscription shall be represented by a <*subscription*> resource (see clause 9.6.8). This allows manipulation of the subscription in a resource oriented manner, e.g. the conditions of a subscription may be modified by modifying a <*subscription*> resource, or a resource subscriber may unsubscribe by deleting the <*subscription*> resource.

The following clauses describe procedures for Creation, Retrieval, Update and Deletion of a <*subscription*> resource.

The following clauses also describe procedures for Creation, Retrieval, Update, and Deletion of a <*crossResourceSubscription*> resource, and the procedure for generating cross-resource notification.

10.2.10.2 Create *<subscription>*

This procedure shall be used to request the creation of a new <*subscription*> resource to instruct the Hosting CSE to send notifications to configured Subscriber(s) for modifications of a subscribed-to resource. The generic create procedure is described in clause 10.1.2.

Table 10.2.10.2-1: *<subscription>* CREATE

***<subscription>* CREATE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content*:** The resource content shall provide the information as defined in clause 9.6.8
Processing at Originator before sending Request	According to clause 10.1.2 with the following additions: The Request shall address a subscribable resource The Request shall include a <*subscription*> resource representation with the attribute *notificationURI* If the *notificationURI* attribute includes Notification Target(s) which is/are not targeting the Originator, the Originator should send the request as non-blocking request (see clauses 8.2.2 and 9.6.12)
Processing at Receiver	According to clause 10.1.2 with the following Which is also the Hosting CSE shall validate the followings: • Check if the subscribed-to resource, addressed in the ***To*** parameter in the Request, is a subscribable resource • Check if the Originator has privileges for retrieving the subscribed-to resource • In case a <*subscription*> resource representation is provided with a *notificationEventType* tag equal to "Update to attributes of the subscribed-to resource with blocking of the triggering UPDATE operation" in the *eventNotificationCriteria* attribute, check that no other subscriptions with this setting exist for the resource in the ***To*** parameter, check that only one entity is targeted by the *notificationURI* attribute and check that this entity has privileges for updating the subscribed-to resource • If an entity listed in the notificationURI is not the Originator, the Hosting CSE may send a Notify request to that entity to verify this <*subscription*> creation request. If the Hosting CSE initiates the verification, it shall check if the verification result in the Notify response is successful or not. If any of the entities listed in the *notificationURI* attribute fails verification then the <*subscription*> create process fails If any of the checks above fails, the Hosting CSE shall send an unsuccessful response to the Originator with corresponding error information. Otherwise, the Hosting CSE shall create the <*subscription*> resource and send a successful response to the Originator. Upon successful creation of a <*subscription*> resource, the Hosing CSE shall evaluate subsequent operations on the subscribed-to resource and trigger notifications in line with the notification policies provisioned in the created <*subscription*> resource
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content*:** address of the created <*subscription*> resource, according to clause 10.1.2
Processing at Originator after receiving Response	According to clause 10.1.2
Exceptions	According to clause 10.1.2

10.2.10.3 Retrieve <*subscription*>

This procedure shall be used to retrieve attributes and child resource information of a <subscription> resource. The generic retrieve procedure is described in clause 10.1.3.

Table 10.2.10.3-1: *<subscription>* RETRIEVE

***<subscription>* RETRIEVE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content*:** void
Processing at Originator before sending Request	According to clause 10.1.3
Processing at Receiver	According to clause 10.1.3
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content*:** attributes of the <*subscription*> resource as defined in clause 9.6.8
Processing at Originator after receiving Response	According to clause 10.1.3
Exceptions	According to clause 10.1.3

10.2.10.4 Update <*subscription*>

This procedure shall be used to update an existing subscription, e.g. extension of its lifetime or the modification of the list of Notification Targets provided in the *notificationURI* attribute. The generic update procedure is described in clause 10.1.4.

Table 10.2.10.4-1: <*subscription*> UPDATE

<*subscription*> UPDATE	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content:*** attributes of the <*subscription*> resource as defined in clause 9.6.8 which need be updated
Processing at Originator before sending Request	According to clause 10.1.4
Processing at Receiver	According to clause 10.1.4 • If the *notificationURI* attribute contains Notification Target(s) that is/are not the Originator, see applicable processing in table 10.2.10.2-1 in clause 10.2.10.2 • If the *latestNotify* attribute is set during this UPDATE operation, the Hosting CSE shall assign ***Event Category*** parameter of value 'latest' of the notifications generated pertaining to the subscription created and remove all buffered pending notifications for this subscription except for the latest one Upon successful updating of a <*subscription*> resource, the Hosing CSE shall evaluate subsequent operations on the subscribed-to resource and trigger notifications in line with the new notification policies provisioned in the created <*subscription*> resource
Information in Response message	According to clause 10.1.4
Processing at Originator after receiving Response	According to clause 10.1.4
Exceptions	According to clause 10.1.4

10.2.10.5 Delete <*subscription*>

This procedure shall be used to unsubscribe an existing subscription. The generic delete procedure is described in clause 10.1.5.

Table 10.2.10.5-1: <*subscription*> DELETE

<*subscription*> DELETE	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.5
Processing at Receiver	According to clause 10.1.5
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	According to clause 10.1.5
Exceptions	According to clause 10.1.5

10.2.10.6 Notification procedures

This procedure shall be used to notify Notification Targets of modifications of a resource for an associated <subscription> resource and notify about a <subscription> resource deletion. Also, this procedure shall be used to request resource subscription verification to Notification Target(s) which is/are not the Originator.

When the notification is forwarded or aggregated by transit CSEs, the Hosting CSE or an transit CSE shall check whether there is a *latestNotify* notification policy to enforce between subscription resource Hosting CSE and the notification target. In that case, the transit CSE as well as the Hosting CSE shall process notification(s) by using the corresponding policy and send processed notification(s) to the next CSE with notification policies related to the enforcement so that the transit CSE is able to enforce the policy defined by the Originator. The notification policies related to the enforcement at this time is verified by using the subscription reference in the Notify request message. If any transit CSE does not recognize the attribute, then it should ignore it.

A notifier can request verification of a Notification Target by including the Originator ID of the subscription creator in the notify request that it generates towards the Notification Target for that purpose. In this case, the Notification Target shall check if both the Notify Originator and the corresponding <subscription> creation Originator have NOTIFY privilege.

- If either of the two checks are not successful, the Receiver shall return an unsuccessful response to the Originator with subscription verification failure information.
- Otherwise, the Receiver shall send successful response to the Originator.

If the Notification Target wants to remove itself from the Notification Target list (i.e. *notificationURI* attribute of the corresponding <subscription> resource), it shall follow one of the procedures below:

- The Notification Target shall set in a Notify response the 'targetRemoval' indicator to TRUE after receiving a Notify request.

NOTE: In this case the Notification Target will not know the outcome of its removal request immediately.

- The Notification Target shall send a Delete Request to the <*notificationTargetSelfReference*> virtual resource subordinated as a child resource to the corresponding <*subscription*> resource.

For either of the above procedures, the Notifier shall handle that according to the *action* attribute defined in the corresponding <*notificationTargetPolicy*> resource for the Notification Target.

10.2.10.7 Notification message handling procedure

When a Hosting CSE receives a <*subscription*> creation request which requires verification (see clause 10.2.10.2), the Hosting CSE may send a notification to perform subscription verification. In this case, the notification shall include the ID of the Originator of the <subscription> resource creation.

When there is an event for a <subscription> resource that triggers a notification, the <subscription> Hosting CSE shall include in the notification the *creator* if the <subscription> resource has *creator* attribute.

When a subscription shall be established that sends notifications upon update of attributes of the subscribed-to resource while blocking the triggering UPDATE operation until the result of the notification is received, the value of the *eventNotificationType* tag in the *notificationEventCriteria* attribute shall be set to "Update to attributes of the subscribed-to resource with blocking of the triggering UPDATE operation", see clause 9.6.8. For this *eventNotificationType* value setting, only one single Notification Target shall be present in the *notificationURI* attribute - see *notificationURI* attribute definition in clause 9.6.8. A subset of attributes of the subscribed-to resource that are triggering a notification when modified can be specified in the *attribute* tag of the *notificationEventCriteria* attribute. If the *attribute* tag is not present, all attributes of the subscribed-to resource will trigger a notification when modified. Upon occurrence of a triggering UPDATE operation that has been validated and results in an authorized UPDATE operation for any of the triggering attributes of the subscribed-to resource, the triggering UPDATE operation shall be blocked before modifying the targeted attributes by the Hosting CSE until a notification request was sent out and a corresponding response message was received or a timeout happens. While such an UPDATE request is pending, no other UPDATE or DELETE requests to the same resource instance shall be processed, i.e. if they occur while the UPDATE operation that triggered this type of subscription is blocked, they need to be delayed until the blocked UPDATE has been completed. When the response status code of the notification response message indicates a successful notification reception by the Notification Target in combination with a successful notification action taken by the Notification Target, the blocked UPDATE operation shall be completed with a successful update of the targeted attribute(s). If the notification response message indicates an unsuccessful notification request reception or a successful notification request reception with unsuccessful notification action by the Notification Target or when the reception of a response message times out, the blocked UPDATE operation shall be completed with no success and no change of the targeted attribute(s).

There shall exist a maximum of only one subscription with this setting of *notificationEventType* for a given resource. All other notification policies shall not be allowed when this setting of *notificationEventType* is used.

Further details of Hosting CSE related notification policies follow:

The *expirationCounter* shall be decreased by one when the Hosting CSE successfully sends the notification request to Receiver(s). If the counter reaches zero, the corresponding subscription resource shall be deleted.

In the case an Originator wants to create batches of notifications rather than have the Hosting CSE send notifications one by one, it may set the *batchNotify* attribute to express its notification policy. The *batchNotify* attribute (notification policy) is based on two values, the number of notifications to be batched for delivery, and/or a duration. When the Hosting CSE generates a notification event it checks the *batchNotify* policy, if a duration value is specified then a timer is started which expires after the duration value. If a number of notifications is specified then notification events are accumulated until the accumulated notification events reaches the specified number. If only the duration is specified, then the accumulated notifications are sent as a batch when the timer expires. If both values are set then accumulated notifications are sent as a batch when either the timer expires or the number is reached whichever happens first. If neither the number nor the duration is specified (i.e. the *batchNotify* attribute is present and empty), then the Hosting CSE shall batch notifications using the default duration value as given by the M2M Service Provider. Note that Hosting CSE shall not batch notifications when the *batchNotify* is not present in the <subscription> resource. When the first notification event is generated then a timer shall be started and keep batching notifications for the duration. After the duration, batched notification shall be sent and a timer shall be set again at the next notification event. For example, a *batchNotify* policy having a duration of 10 minutes and a number of 20 notifications will accumulate notifications which is sent when the first of these two conditions are satisfied. The sending order is First-In First-Out (FIFO). The batch timer shall be reset once the batched notifications are being sent. *notificationEventCat* is checked at the time of batch transmission and applied to each notification individually in the batch. Stored notification events may be dropped according to the *notificationStoragePriority* and the *notificationCongestionPolicy* (see clause 9.6.3). When the *batchNotify* and *latestNotify* attributes (notification policies) are used together, they enable two ways of sampling notification events for notification generation. If the number of notification is set high then the duration value will drive the policy, and the *latestNotify* policy will cause a single event notification every duration period, e.g. send the latest event notification every hour. If the duration value is set high then the number of notifications will drive the policy, and the *latestNotify* policy will cause a single notification for every specified number of notifications, e.g. send the latest event notification for every 500 events notifications generated. The scope of the *batchNotify* policy is the Hosting CSE for the one subscription it is set in, and does not extend to transit CSEs.

In the case when an Originator wants to limits the rate at which notifications are sent, it may set the *rateLimit* attribute (notification policy) to express its notification policy. The *rateLimit* policy is based on two values, a maximum specified number of events (e.g. 10 000) that may be sent within some specified *rateLimit* window duration (e.g. 60 seconds), and the *rateLimit* window duration. When the Hosting CSE generates a notification event it checks the *rateLimit* policy and whether the current total number of events sent is less than the maximum number of events within the current *rateLimit* window duration. If the current total is less than the maximum number then the notification may be sent. If it is equal or more then the notification is temporarily stored until the end of the current window duration, when the sending of notification events restarts in the next window duration. The sending of notification events continues as long as the maximum number of notification events is not exceeded within the window duration. The *rateLimit* windows are sequential (not rolling). The *rateLimit* policy may be used simultaneously with *batchNotify* and *notificationStoragePriority* policies. The scope of the *rateLimit* policy is the Hosting CSE for the one subscription it is set in, and does not extend to transit CSEs.

The *pendingNotification* attribute (notification policy) indicates the notification procedure to be followed following a connectionless period (due to lack of notification schedule or reachability schedule). When the Hosting CSE generates a notification with the *pendingNotification*, it shall check the notification schedule of the subscription and the reachability schedule associated with the Notification Target. If there is no restriction then the notification is immediately sent, otherwise the notification may be cached according to the *pendingNotification*. If caching of retained notifications is supported on the Hosting CSE and contains the subscribed events then pending notification (those that occurred during the connectionless period) will be sent to Notification Target per the *pendingNotification* policy. If it is set to the "sendLatest", most recent notification should be sent and it shall have the ***Event Category*** set to "latest". Figure 10.2.10.7-1 illustrates an example for this case. If it is set to "sendAllPending", all the missed cached notifications should be sent in the order they occurred. Figure 10.2.10.7-2 illustrates an example of this case. The Hosting CSE may use the *pendingNotification* policy to determine whether and how many interim notifications to retain in its cache. The *pendingNotification* policy may be used simultaneously with any other notification policy, which would impact what would be sent during the connection period. The scope of the *pendingNotification* is the Hosting CSE for the one subscription it is set in, and does not extend to transit CSEs.

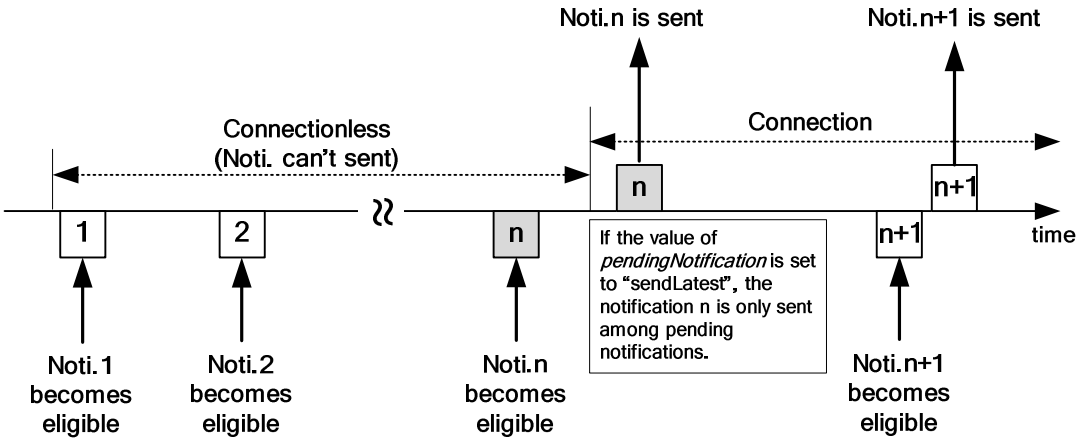


Figure 10.2.10.7-1: Notification Mechanism when *pendingNotification* (sendLatest) is used

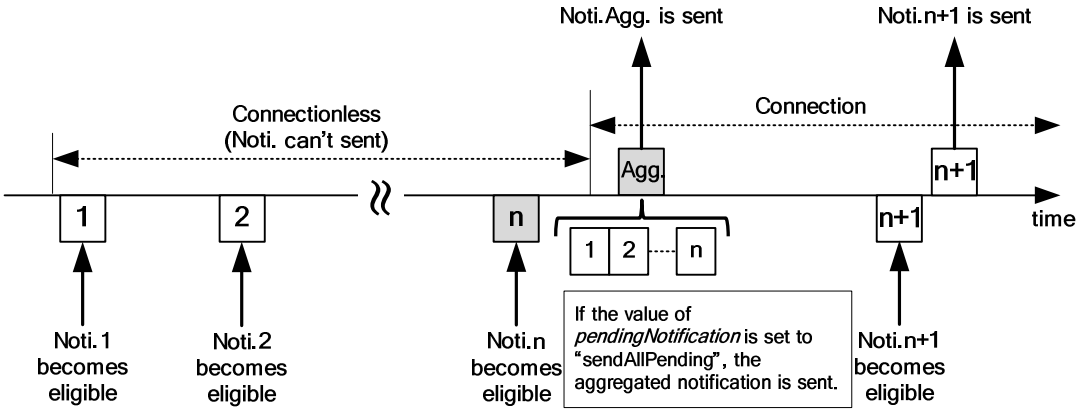


Figure 10.2.10.7-2: Notification Mechanism when *pendingNotification* (sendAllPending) is used

In the case an Originator wants (for example in the case where notification events occur on an irregular basis) that notifications are be sent for events generated prior to the creation of this subscription, it may set the *preSubscriptionNotify* attribute (notification policy) to express its notification policy. The *preSubscriptionNotify* policy is based upon a number of prior notifications that the Originator wants to be sent. When creating a subscription the Hosting CSE checks the *preSubscriptionNotify* policy. If caching of retained notifications is supported on the Hosting CSE and contains the subscribed events then prior notification events shall be sent to Receiver(s) up to the number requested by the *preSubscriptionNotify* policy. If caching of retained notifications is supported for the subscribed events but the available number of prior notification events is less than the number requested then the Hosting CSE shall send those notifications. If caching of retained notifications is not supported, then the response to the subscription creation request shall include a warning. The *preSubscriptionNotify* policy may be used simultaneously with any other notification policy. The scope of the *preSubscriptionNotify* policy is the Hosting CSE for the one subscription it is set in, and does not extend to transit CSEs.

The *latestNotify* attribute (notification policy) indicates if the Originator is only interested in the latest state of the subscribed-to resource. If the *latestNotify* attribute is set, the Hosting CSE shall assign ***Event Category*** parameter of value 'latest' to the latest notifications generated pertaining to the subscription created. In the case the Receiver is a transit CSE which forwards or aggregates the notifications before sending them to the Originator or the other transit CSEs, upon receiving the notification with the ***Event Category*** set to 'latest', the transit CSE shall identify the latest notification with the same subscription reference while storing the notifications locally. When the Receiver as a transit CSE needs to send the pending notifications, it shall send the latest notification only for that subscription. The scope of the *latestNotify* policy is the Hosting CSE as well as transit CSEs.

The *notificationContentType* attribute (notification policy) indicates the notification content type that shall be contained in notifications. The *notificationContentType values* shall be "modified attributes" (i.e. send the modified attribute(s) only), or "all attributes" (i.e. send all attributes of the subscribed-to resource), or "ID" of the resource indicated in the *notificationEventType* condition tag or the value "Trigger Payload". If it is not given by the Originator at the creation procedure, the default is "all attributes". The scope of the *notificationContentType* policy is the Hosting CSE for all Originator's subscriptions, and does not extend to transit CSEs. The value "Trigger Payload" for this attribute is only valid when at least one "*notificationEventType"* tag in the *eventNotificationCriteria* attribute is set to "Trigger Received targeting the MN/ASN-AE associated with the <*AE*> parent resource".

The *notificationEventCat* attribute (notification policy) indicates an event category of the subscription that shall be included in the notification request to be able for the Notification Target to correctly handle the notification. When the *notificationEventCat* policy is not configured by the Originator, it shall be determined as a default value by the CMDH policy. The scope of the *notificationEventCat* policy is the Hosting CSE for all Originator's subscriptions, and does not extend to transit CSEs.

When the Hosting CSE receives unsuccessful Notify response with subscription verification failure information, the Hosting CSE shall send unsuccessful result to the Originator of the corresponding <subscription> creation procedure if it has not created the <*subscription*> resource, otherwise the Hosting CSE may delete the corresponding <*subscription*> resource.

Table 10.2.10.7-1: Notification Procedure

Description	
Information in Request message	According to clause 10.1.6 with the following additions: ***Content*:** • notification data that represents the content of subscribed-to resource may be included. The content is decided by *notificationContentType* attribute • subscription reference (i.e. address of the corresponding <*subscription*> resource) that generates this notification shall be included • notification event type shall be included • monitored operation and its Originator information shall be included when *operationMonitor* condition in the *eventNotificationCriteria* attribute is configured • *notificationForwardingURI* in case the subscriber intends the group to aggregate the notifications
Processing at Originator before sending Request	Notification is triggered regarding subscription information in a <subscription> resource
Processing at Receiver	According to clause 10.1.6
Information in Response message	According to clause 10.1.6
Processing at Originator after receiving Response	If the response includes 'targetRemoval' indicator which is set to TRUE, then the Notifier(i.e. the Originator of the Notify request) shall perform the procedure in clause 10.2.10.8 (Notification target removal handling procedure)
Exceptions	According to clause 10.1.6

10.2.10.8 Notification Target removal procedure

Notification Target removal involves <notificationTargetSelfReference>, <notificationTargetMgmtPolicyRef>, <notificationTargetPolicy> and <policyDeletionRules> resources. When a Notification Target sends a Delete request to a <notificationTargetSelfReference> virtual resource, the <subscription> Hosting CSE gets deletion policies to handle the removal request. Depending on the *action* of the <notificationTargetPolicy> resource which is associated to the Notification Target, the <subscription> Hosting CSE can accept or reject the request by itself and get the authorization from the <subscription> resource creator. When the Hosting CSE gets a successful response from the creator, it deletes the target in the notificationURI and sends back the successful response to the removal Originator. If the action is to inform the subscription Originator, then the Hosting CSE just sends a notification to inform of this removal request. The subscription Originator is supposed to determine and perform the target removal. Figure 10.2.10.8-1 briefly illustrates the procedure mentioned above depending on the different action settings.

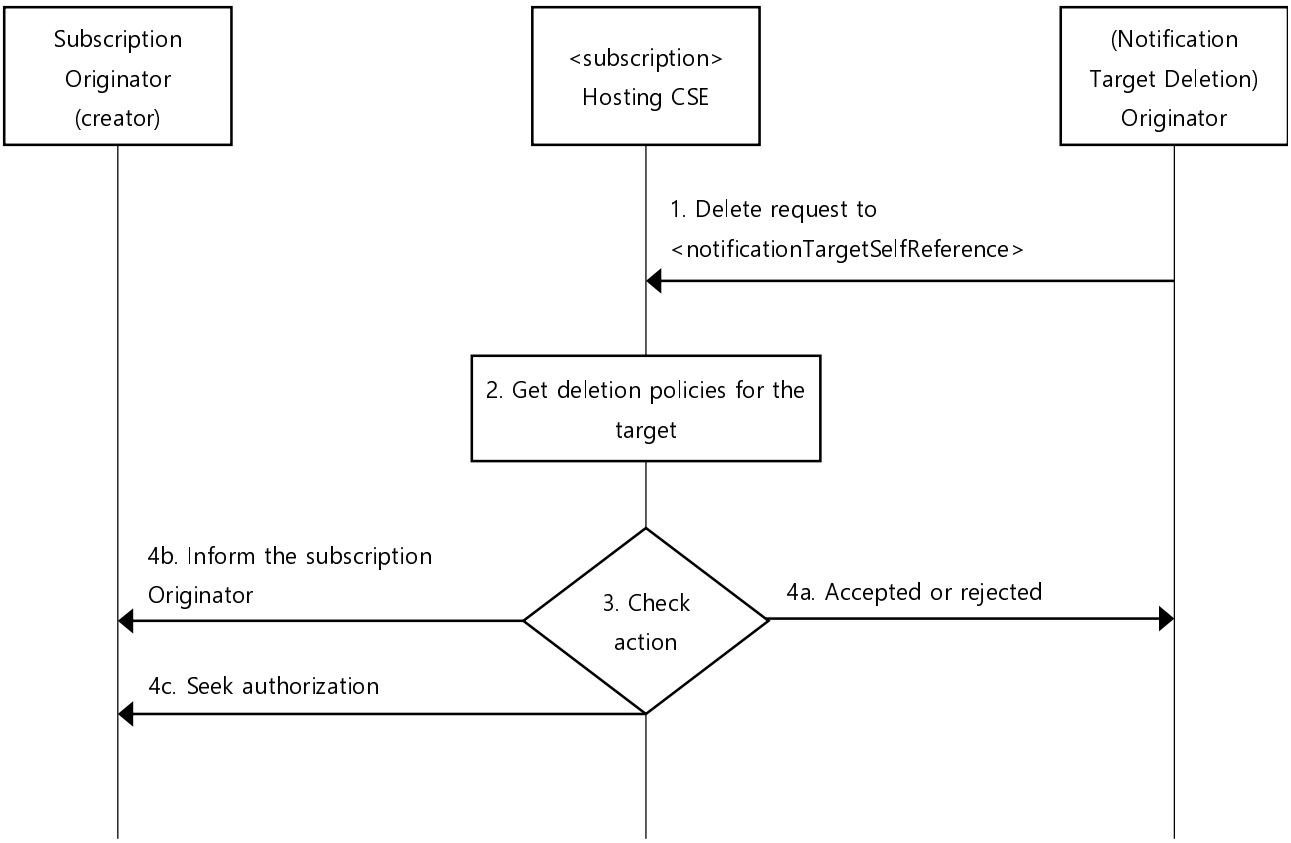


Figure 10.2.10.8-1: Notification target removal procedure

The Notifier(i.e. the Originator of the Notify request) shall handle the notification target removal based on a <*notificationTargetPolicy*> resource. Selecting the applicable<*notificationTargetPolicy*> resource shall be performed as follows:

- Check if there is a <*notificationTargetMgmtPolicyRef*> resource as a child of the <*subscription*> resource which includes the Notification Target in the *notificationTargetURI* attribute. If one is located , the Notifier shall apply the <*notificationTargetPolicy*> resource specified in the *notificationPolicyID* attribute in the matching <*notificationTargetMgmtPolicyRef*> resource.

- Otherwise, the Notifier shall check if there is a <*notificationTargetMgmtPolicyRef*> resource which has the *creator* attribute set in the corresponding <*subscription*> resource and there is a <*notificationTargetPolicy*> resource which has the *policyLabel* attribute set as "default" and the *creator* attribute is equal to the *creator* of the <*subscription*> resource.

- Otherwise, the Notifier shall fetch the <*notificationTargetPolicy*> resource which has the *policyLabel* attribute set as "default".

With the selected <*notificationTargetPolicy*> resource, the Notifier shall handle the target removal as specified in the *action* attribute of the <*notificationTargetPolicy*> resource. If there is <*policyDeletionRules*> resource(s) then the action shall be applied when the rule(s) is satisfied.

The action shall be performed as follows:

- If the action is "accept", then the Notifier shall remove the address which is corresponding to the Notification Target and returns a successful response if applicable.

- If the action is "reject" and if the target removal was requested with Delete request (clause 10.2.10.9), then the Notifier shall return an unsuccessful response if applicable.

- If the action is "seek authorization from the subscription creator", then the Notifier shall return a successful response to the Notification Target if applicable and shall send a Notify request including the ID of the <subscription> resource, the Notification Target, and the 'removalAuthorization' indicator which is set as TRUE, to the subscription creator. When the Notifier gets successful response from the creator, then the Notifier shall remove the address which is corresponding to the Notification Target.

- If the action is "inform the subscription creator", then the Notifier shall return a successful response to the Notification Target if applicable and shall send a Notify request including the ID of the <subscription> resource, the Notification Target, and the 'targetRemoval' indicator which is set as TRUE to the subscription creator.

10.2.10.9 Delete <*notificationTargetSelfReference*>

Only Delete operations shall be allowed for the <*notificationTargetRemove*> resource.

This procedure shall apply to the <*subscription*> resource . Whenever a Delete Request is received at the <*notificationTargetSelfReference*> virtual resource from a Notification Target, the Notifier shall handle the request according to the *action* attribute defined in the <*notificationTargetPolicy*> resource which is linked from the <*notificationTargetDisposition*> resource. Detailed handling procedure is specified in the clause 10.2.10.8 (Notification Target removal handling procedure).

10.2.10.10 Create <*notificationTargetMgmtPolicyRef*>

This procedure shall be used for creating a <*notificationTargetMgmtPolicyRef*> resource.

Table 10.2.10.10-1: *<notificationTargetMgmtPolicyRef>* CREATE

***<notificationTargetMgmtPolicyRef>* CREATE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content:*** The resource content shall provide the information as defined in clause 9.6.6
Processing at Originator before sending Request	According to clause 10.1.2
Processing at Receiver	According to clause 10.1.2
Information in Response message	According to clause 10.1.2
Processing at Originator after receiving Response	According to clause 10.1.2
Exceptions	According to clause 10.1.2

10.2.10.11 Retrieve <*notificationTargetMgmtPolicyRef*>

This procedure shall be used for retrieving the attributes of a <*notificationTargetMgmtPolicyRef*> resource.

Table 10.2.10.11-1: *<notificationTargetMgmtPolicyRef>* RETRIEVE

***<notificationTargetMgmtPolicyRef>* RETRIEVE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.3
Processing at Receiver	According to clause 10.1.3
Information in Response message	All parameters defined in table 8.1.3-1 apply
Processing at Originator after receiving Response	According to clause 10.1.3
Exceptions	According to clause 10.1.3

10.2.10.12 Update <*notificationTargetMgmtPolicyRef*>

This procedure shall be used for updating attributes of a <*notificationTargetMgtPolicyRef*> resource.

Table 10.2.10.12-1: <*notificationTargetMgmtPolicyRef*> UPDATE

<*notificationTargetMgmtPolicyRef*> UPDATE	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.4
Processing at Receiver	According to clause 10.1.4
Information in Response message	According to clause 10.1.4
Processing at Originator after receiving Response	According to clause 10.1.4
Exceptions	According to clause 10.1.4

10.2.10.13 Delete <*notificationTargetMgmtPolicyRef*>

This procedure shall be used for deleting a <*notificationTargetMgmtPolicyRef*> resource.

Table 10.2.10.13-1: <*notificationTargetMgmtPolicyRef*> DELETE

<*notificationTargetMgmtPolicyRef*> DELETE	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.5
Processing at Receiver	According to clause 10.1.5
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	According to clause 10.1.5
Exceptions	According to clause 10.1.5

10.2.10.14 Create <*notificationTargetPolicy*>

This procedure shall be used for creating a <*notificationTargetPolicy*> resource.

Table 10.2.10.14-1: <*notificationTargetPolicy*> CREATE

<*notificationTargetPolicy*> CREATE	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content:*** The resource content shall provide the information as defined in clause 9.6.6
Processing at Originator before sending Request	According to clause 10.1.2
Processing at Receiver	According to clause 10.1.2
Information in Response message	According to clause 10.1.2
Processing at Originator after receiving Response	According to clause 10.1.2
Exceptions	According to clause 10.1.2

10.2.10.15 Retrieve <*notificationTargetPolicy*>

This procedure shall be used for retrieving the attributes of a <*notificationTargetPolicy*> resource.

Table 10.2.10.15-1: <*notificationTargetPolicy*> RETRIEVE

<*notificationTargetPolicy*> RETRIEVE	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.3
Processing at Receiver	According to clause 10.1.3
Information in Response message	All parameters defined in table 8.1.3-1 apply
Processing at Originator after receiving Response	According to clause 10.1.3
Exceptions	According to clause 10.1.3

10.2.10.16 Update <*notificationTargetPolicy*>

This procedure shall be used for updating attributes of a <*notificationTargetPolicy*> resource.

Table 10.2.10.16-1: <*notificationTargetPolicy*> UPDATE

<*notificationTargetPolicy*> UPDATE	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.4
Processing at Receiver	According to clause 10.1.4
Information in Response message	According to clause 10.1.4
Processing at Originator after receiving Response	According to clause 10.1.4
Exceptions	According to clause 10.1.4

10.2.10.17 Delete <*notificationTargetPolicy*>

This procedure shall be used for deleting a <*notificationTargetPolicy*> resource.

Table 10.2.10.17-1: <*notificationTargetPolicy*> DELETE

<*notificationTargetPolicy*> DELETE	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.5
Processing at Receiver	According to clause 10.1.5
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	According to clause 10.1.5
Exceptions	According to clause 10.1.5

10.2.10.18 Create <*policyDeletionRules*>

This procedure shall be used for creating a <*policyDeletionRules*> resource.

Table 10.2.10.18-1: <*policyDeletionRules*> CREATE

<*policyDeletionRules*> CREATE	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content**:* The resource content shall provide the information as defined in clause 9.6.6
Processing at Originator before sending Request	According to clause 10.1.2
Processing at Receiver	According to clause 10.1.2
Information in Response message	According to clause 10.1.2
Processing at Originator after receiving Response	According to clause 10.1.2
Exceptions	According to clause 10.1.2

10.2.10.19 Retrieve <*policyDeletionRules*>

This procedure shall be used for retrieving the attributes of a <*policyDeletionRules*> resource.

Table 10.2.10.19-1: <*policyDeletionRules*> RETRIEVE

<*policyDeletionRules*> RETRIEVE	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.3
Processing at Receiver	According to clause 10.1.3
Information in Response message	All parameters defined in table 8.1.3-1 apply
Processing at Originator after receiving Response	According to clause 10.1.3
Exceptions	According to clause 10.1.3

10.2.10.20 Update <*policyDeletionRules*>

This procedure shall be used for updating attributes of a <*policyDeletionRules*> resource.

Table 10.2.10.20-1: <*policyDeletionRules*> UPDATE

<*policyDeletionRules*> UPDATE	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.4
Processing at Receiver	According to clause 10.1.4
Information in Response message	According to clause 10.1.4
Processing at Originator after receiving Response	According to clause 10.1.4
Exceptions	According to clause 10.1.4

10.2.10.21 Delete <*policyDeletionRules*>

This procedure shall be used for deleting a <*policyDeletionRules*> resource.

Table 10.2.10.21-1: <*policyDeletionRules*> DELETE

<*policyDeletionRules*> DELETE	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.5
Processing at Receiver	According to clause 10.1.5
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	According to clause 10.1.5
Exceptions	According to clause 10.1.5

10.2.10.22 Create <*crossResourceSubscription*>

This procedure shall be used to request the creation of a new <*crossResourceSubscription*> resource to be notified for the modifications of multiple subscribed-to target resources. The generic create procedure is described in clause 10.1.2.

Table 10.2.10.22-1: <*crossResourceSubscription*> CREATE

<*crossResourceSubscription*> CREATE	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content*:** The resource content shall provide the information as defined in clause 9.6.58
Processing at Originator before sending Request	According to clause 10.1.2 with the following additions: The Request shall include at least one of attributes: *regularResourcesAsTarget*, *subscriptionResourcesAsTarget*. The Request shall include *timeWindowType* and *timeWindowSize*. The Request shall include *notificationURI(s)*. The Request shall include *notificationContentType*. The Request shall include *eventNotificationCriteriaSet* if *regularResourcesAsTarget* is included in the Request. If the request includes *notificationURI(s)* which is not the Originator, the Originator should send the request as non-blocking request (see clauses 8.2.2 and 9.6.12)
Processing at Receiver	According to clause 10.1.2 with the following Which is also the Hosting CSE shall validate the followings: • Check if the Originator has privileges for creating a child resource in the ***To*** parameter in the Request. • Check if each target resource in *regularResourcesAsTarget* is a subscribable resource. • Check if the Originator has privileges for retrieving the subscribed-to resource • If a notificationURI is not the Originator, the Hosting CSE may send a Notify request to the *notificationURI* to verify this <*crossResourceSubscription*> creation request. If the Hosting CSE initiates the verification, it shall check if the verification result in the Notify response is successful or not. If any *notificationURI* contained in a list fails verification then the <*crossResourceSubscription*> create process fails. If any of the checks above fails, the Hosting CSE shall send an unsuccessful response to the Originator with corresponding error information. Otherwise, the Hosting CSE shall use the following procedure to create the <*crossResourceSubscription*> resource and send a successful or an unsuccessful response to the Originator.

***<crossResourceSubscription>* CREATE**	
	• If *regularResourcesAsTarget* is included, the Hosting CSE shall send a CREATE request message to each target resource host to create a *<subscription>* child resource under each target resource indicated by *regularResourcesAsTarget* using corresponding event notification criteria as included in *eventNotificationCriteriaSet*; the *notificationURI* for the *<subscription>* to be created shall be of this *<crossResourceSubscription>* resource. The *associatedCrossResourceSub* attribute having the address of *<crossResourceSubscription>* resource shall be included in the *<subscription>* resource. In the CREATE request, the Hosting CSE shall include the identifier of the Originator, which shall be leveraged by the target resource host to verify if the Originator has the privilege to create a *<subscription>* resource; if the Originator has no privilege to create this *<subscription>* resource, this step shall be regarded as a failure. If any *<subscription>* for a target resource cannot be successfully created, the Hosting CSE shall send an unsuccessful response to the Originator and shall delete already created *<subscription>* resources. • If *subscriptionResourcesAsTarget* is included, the Hosting CSE shall add the resource identifier of this *<crossResourceSubscription>* resource to the *associatedCrossResourceSub* attribute of each *<subscription>* resource as indicated in *subscriptionResourcesAsTarget* by issuing an UPDATE request to the *<subscription>* resource host. In the UPDATE request, the Hosting CSE shall include the identifier of the Originator, which shall be leveraged by the *<subscription>* resource host to verify if the Originator has the privilege to retrieve this *<subscription>* resource and update the *subscriptionResourcesAsTarget* attribute; if the Originator has no privilege to retrieve this *<subscription>* resource and update the *subscriptionResourcesAsTarget* attribute, this step shall be regarded as a failure. If this step is not successfully performed, the Hosting CSE shall send an unsuccessful response to the Originator and shall also remove it from already successful associated *<subscription>* resources. • Otherwise, the Hosting CSE shall send a successful response to the Originator. Once the *<crossResourceSubscription>* resource is created, the Hosting CSE shall start the time window.
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content*:** address of the created *<crossResourceSubscription>* resource, according to clause 10.1.2
Processing at Originator after receiving Response	According to clause 10.1.2
Exceptions	According to clause 10.1.2

10.2.10.23 Retrieve *<crossResourceSubscription>*

This procedure shall be used to retrieve attributes and child resource information of a *<crossResourceSubscription>* resource. The generic retrieve procedure is described in clause 10.1.3.

Table 10.2.10.23-1: *<crossResourceSubscription>* RETRIEVE

***<crossResourceSubscription>* RETRIEVE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content*:** void
Processing at Originator before sending Request	According to clause 10.1.3
Processing at Receiver	According to clause 10.1.3
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content*:** attributes of the *<crossResourceSubscription>* resource as defined in clause 9.6.58
Processing at Originator after receiving Response	According to clause 10.1.3
Exceptions	According to clause 10.1.3

10.2.10.24 Update <*crossResourceSubscription*>

This procedure shall be used to update an existing cross-resource subscription (i.e. represented as a <*crossResourceSubscription*> resource), e.g. the modification of the list of *notificationURI(s)*. The generic update procedure is described in clause 10.1.4.

Table 10.2.10.24-1: <*crossResourceSubscription*> UPDATE

<*crossResourceSubscription*> UPDATE	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content:*** attributes of the <*crossResourceSubscription*> resource as defined in clause 9.6.58 which need be updated
Processing at Originator before sending Request	According to clause 10.1.4
Processing at Receiver	According to clause 10.1.4 • If a *notificationURI* is not the Originator, see table 10.2.10.2-1 in clause 10.2.10.2 • If *regularResourcesAsTarget* is updated, the Hosting CSE shall: – First, delete the <*subscription*> child resource if the target resource is deleted in the new *regularResourcesAsTarget* attribute value. – Second, issue a CREATE request to create a <*subscription*> child resource under each target resource indicated by the new value of *regularResourcesAsTarget* using corresponding event notification criteria as included in *eventNotificationCriteriaSet*; the *notificationURI* for the <*subscription*> to be created shall be the Hosting CSE itself. The *associatedCrossResourceSub* attribute having the address of <*crossResourceSubscription*> resource shall be included in the <*subscription*> resource. In the CREATE request, the Hosting CSE shall include the identifier of the Originator, which shall be leveraged by the target resource host to verify if the Originator has the privilege to create a <*subscription*> resource; if the Originator has no privilege to create this <*subscription*> resource, this step shall be regarded as a failure. If any <*subscription*> for a target resource cannot be successfully created, the Hosting CSE shall send an unsuccessful response to the Originator and shall delete already created <*subscription*> resources. • If *subscriptionResourcesAsTarget* is updated, the Hosting CSE shall: – First, remove itself from the *associatedCrossResourceSub* of each <*subscription*> resource as deleted in the new value of *subscriptionResourcesAsTarget*. – Second, issue an UPDATE request to add the resource identifier of this <*crossResourceSubscription*> resource to the *associatedCrossResourceSub o*f each <*subscription*> resource as indicated in the new value of *subscriptionResourcesAsTarget*. In the UPDATE request, the Hosting CSE shall include the identifier of the Originator, which shall be leveraged by the <*subscription*> resource host to verify if the Originator has the privilege to retrieve this <*subscription*> resource; if the Originator has no privilege to retrieve this <*subscription*> resource, this step shall be regarded as a failure. If this step is not successfully performed, the Hosting CSE shall send an unsuccessful response to the Originator. • If *eventNotificationCriteriaSet* is updated, the Hosting CSE shall use each new event notification criteria to update the *eventNotificationCriteria* of the corresponding <*subscription*> child resource which has been created previously using the clause 10.2.10.22 for each target resource as included in the *regularResourcesAsTarget* attribute.
Information in Response message	According to clause 10.1.4
Processing at Originator after receiving Response	According to clause 10.1.4
Exceptions	According to clause 10.1.4

10.2.10.25 Delete <*crossResourceSubscription*>

This procedure shall be used to unsubscribe an existing cross-resource subscription (i.e. represented as a <*crossResourceSubscription*> resource). The generic delete procedure is described in clause 10.1.5.

Table 10.2.10.25-1: <*crossResourceSubscription*> DELETE

<*crossResourceSubscription*> DELETE	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.5
Processing at Receiver	According to clause 10.1.5 The Hosting CSE shall check whether the Originator has the privilege to delete the <*crossResourceSubscription*> resource. If this check fails, the Hosting CSE shall send an unsuccessful response to the Originator with corresponding error information; otherwise, the Hosting CSE shall perform the following operations: • If *regularResourcesAsTarget* is included in the <*crossResourceSubscription*> to be deleted, the Hosting CSE shall issue a DELETE request to delete the <*subscription*> child resource of the corresponding target resource as contained in the *regularResourcesAsTarget* attribute which has been created previously using the clause 10.2.10.22. The identifier of the <*crossResourceSubscription*> resource creator (i.e. its *creator* attribute) shall be included in this DELETE request and shall be leveraged by the target resource host to check the privilege to delete the <*subscription*> resource. • If *subscriptionResourcesAsTarget* is included, the Hosting CSE shall remove the resource identifier of this <*crossResourceSubscription*> from the *associatedCrossResourceSub* of each <*subscription*> resource as indicated in the *subscriptionResourcesAsTarget*.
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	According to clause 10.1.5
Exceptions	According to clause 10.1.5

10.2.10.26 Cross-Resource Notification Procedure

For each <*crossResourceSubscription*> resource, the Hosting CSE shall use the following procedures to determine whether a cross-resource notification shall be generated and sent to notification targets as indicated in the *notificationURI* attribute.

- The Hosting CSE shall wait for notifications from target resources as indicated by *regularResourcesAsTarget* and *subscriptionResourcesAsTarget* to a <*crossResourceSubscription*> resource.

- The Hosting CSE shall use the designated time window mechanism as indicated by *timeWindowType* to determine if a cross-resource notification shall be issued each time when receiving a notification from a target resource as indicated by *regularResourcesAsTarget* and *subscriptionResourcesAsTarget*. Only when expected changes on all target resources occur within the required time window (as indicated by *timeWindowSize*), the Hosting CSE shall issue a notification (i.e. cross-resource notification); otherwise, the Hosting CSE shall discard the received notification from any target resource.

10.2.11 Service Charging and Accounting Procedures

10.2.11.1 Service event-based statistics collection for applications

This clause is informative and provides a use case example to explain how the Infrastructure Node provides statistics for AEs using the <*statsConfig*> and <*statsCollect*> resources as defined in clauses 9.6.23, 9.6.24 and 9.6.25.

Figure 10.2.11.1-1 shows an example of service layer event-based charging based on the Infrastructure Node:

Step 1-2: A statistics collection resource called <*statsConfigSCA1*> was created at the IN-CSE by a billing application. Note that the <*statsConfig*> can also be provisioned. In this use case, the <*statsConfigSCA1*> has the <*eventConfigSCA1*> sub-resource. For this specific use case, the <*eventConfigSCA1*> can be set as following: The *eventID* attribute is set with a unique ID to differentiate from other chargeable events. The *eventType* attribute defines what event will trigger the generation of service statistics collection record and is set to "Data Operation" for this case. *eventStart* and *eventEnd* attributes apply to timer based event so they will not be included in this event. *operationType* attribute will be "RETRIEVE". *dataSize* attribute does not apply so it is not included.

Step 3-5: In this example, AE1 already registered to IN-CSE. IN-CSE can make the statistics collection configuration accessible by AE. Based on the <*statsConfigSCA1*>, AE1 creates a statistics collection trigger for itself, stored in <*statsCollectAE1*>. AE1 will fill in the information for the collection rule. For example, it fills the *collectingEntityID* attribute with the AE-ID of AE1, and the *collectedEntityID* attribute empty, which means to collect for any entities. *status* attribute is set to "Active". The *statModel* is *event-based*. The *eventID* is set with the same ID value as the *eventID* in the <*eventConfigSCA1*>. This event collection trigger can be stored in the <*eventConfigSCA1*> resource at the IN-CSE and IN-CSE will assign a unique ID in attribute *statsCollectID*.

Step 6-8: When the configured event happens, i.e. when AE2 performed a RETRIEVE operation to the data stored by AE1 at IN-CSE, the event is recorded by IN-CSE. IN-CSE generates a service statistics collection record and sends it to AE1. AE1 can choose to use such information for statistics or billing. Transfer of the statistics is out of scope of the present document.

Step 9: The AE of billing application can update or retrieve the charging policies and collection scenarios that it has the access control privilege.

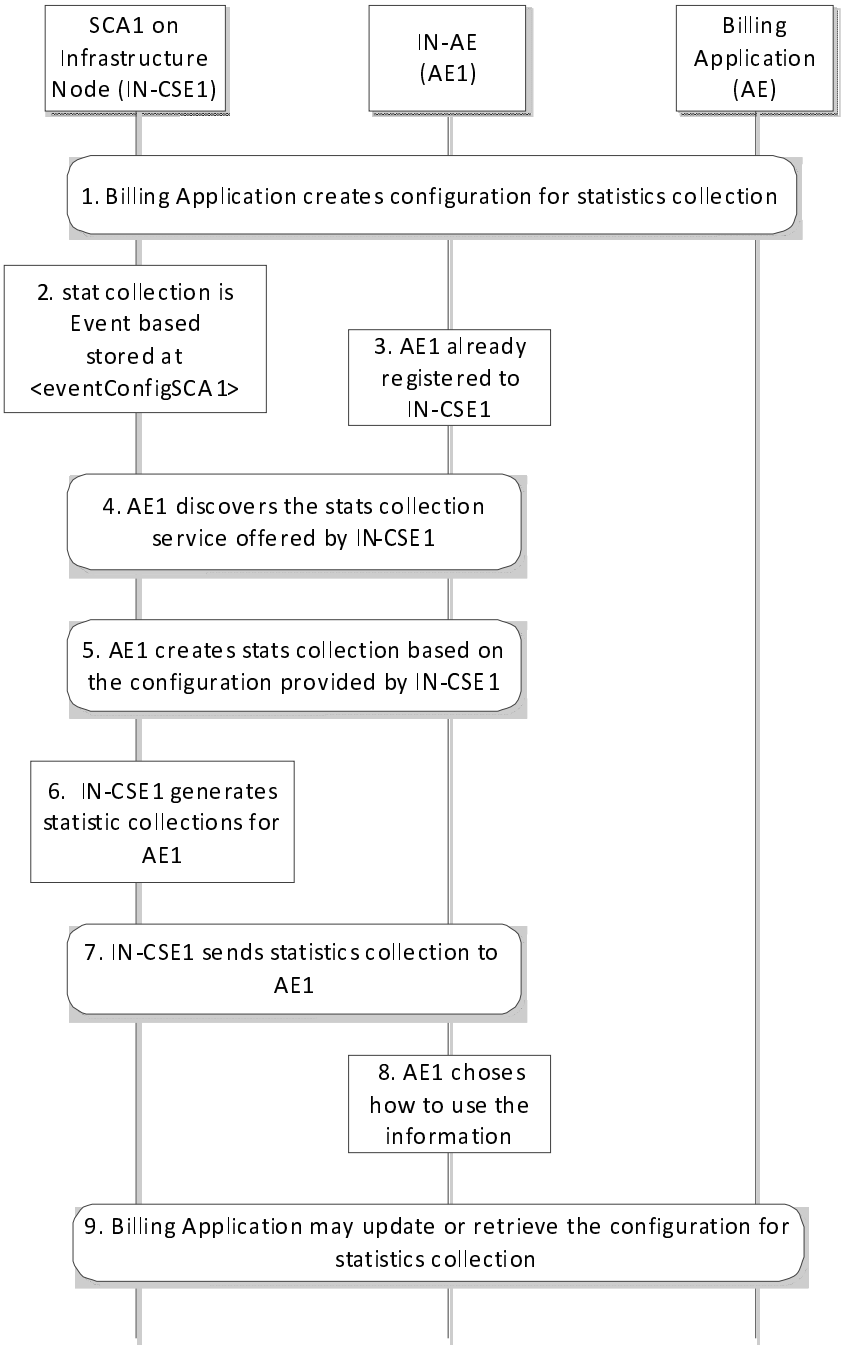


Figure 10.2.11.1-1: Event-based Statistics Collection for Applications

10.2.11.2 Create <*statsConfig*>

This procedure shall be used for the Originator to establish a set of configurations for statistics collection at the Receiver.

The configurations shall be stored at the <*statsConfig*> resource and each instance of the <*statsConfig*> resource shall represent a specific configuration.

The Originator shall be an AE that wants to set up the statistics collection configurations.

The Receiver shall be an IN-CSE.

Table 10.2.11.2-1: <*statsConfig*> CREATE

<*statsConfig*> CREATE	
Information in Request message	***From:*** Identifier of the AE that initiates the Request ***To:*** The address of the <*CSEBase*> where the <*statsConfig*> resource is intended to be Created ***Content:*** The representation of the <*statsConfig*> resource for which the attributes are described in clause 9.6.23 Other information in the Request message is defined according to clause 10.1.2
Processing at Originator before sending Request	The Originator shall request to Create a new <*statsConfig*> resource by addressing to the <*CSEBase*> resource of a Hosting CSE. The Originator shall be an AE
Processing at Receiver	According to clause 10.1.2
Information in Response message	According to clause 10.1.2
Processing at Originator after receiving Response	None
Exceptions	According to clause 10.1.2

10.2.11.3 Retrieve <*statsConfig*>

The RETRIEVE procedure shall be used for the Originator to retrieve the existing <*statsConfig*> resource from the Receiver.

The Originator shall be an AE that is allowed to retrieve configuration information available for AEs within an IN-CSE.

The Receiver shall be the IN- CSE containing the <*statsConfig*> resource.

Table 10.2.11.3-1: <*statsConfig*> RETRIEVE

<*statsConfig*> RETRIEVE	
Information in Request message	***From*:** ID of the Originator ***To*:** Address of the <*statsConfig*> resource or its attribute to be retrieved
Processing at Originator before sending Request	The Originator shall request to obtain <*statsConfig*> resource information by using the RETRIEVE operation. The request shall address the specific <*statsConfig*> resource or its attributes of a Hosting CSE. The Originator shall be an AE
Processing at Receiver	According to clause 10.1.3
Information in Response message	According to clause 10.1.3
Processing at Originator after receiving Response	According to clause 10.1.3
Exceptions	According to clause 10.1.3

10.2.11.4 Update <*statsConfig*>

This procedure shall be used for updating <*statsConfig*> resource.

An UPDATE procedure on the <*statsConfig*> resource is used for the Originator to update charging related policies at the Receiver.

The Originator shall be the AE that created the <*statsConfig*> resource. The same AE shall be able to update the resource.

The Receiver shall be a CSE containing the <*statsConfig*> resource.

Table 10.2.11.4-1: <*statsConfig*> UPDATE

<*statsConfig*> UPDATE	
Information in Request message	***From*:** ID of the Originator ***To*:** Address of the <*statsConfig*> resource to be updated ***Content*:** the Originator provides the attributes of <*statsConfig*> to be updated
Processing at Originator before sending Request	According to clause 10.1.4
Processing at Receiver	According to clause 10.1.4
Information in Response message	According to clause 10.1.4
Processing at Originator after receiving Response	None
Exceptions	According to clause 10.1.4

10.2.11.5 Delete <*statsConfig*>

This procedure shall be used for deleting <*statsConfig*> resource.

The Originator shall be the AE that created the <*statsConfig*> resource.

The Receiver shall be a CSE containing the <*statsConfig*> resource.

Table 10.2.11.5-1: <*statsConfig*> DELETE

<*statsConfig*> DELETE	
Information in Request message	***From*:** ID of the Originator ***To*:** Address of the <*statsConfig*> resource to be deleted
Processing at Originator before sending Request	According to clause 10.1.5
Processing at Receiver	According to clause 10.1.5
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	None
Exceptions	According to clause 10.1.5

10.2.11.6 Create <*eventConfig*>

This procedure shall be used to create <*eventConfig*> resource.

Table 10.2.11.6-1: <*eventConfig*> CREATE

<*eventConfig*> CREATE	
Information in Request message	***From:*** Identifier of the AE that initiates the Request ***To:*** The address of the <*statsConfig*> resource where the <*eventConfig*> sub-resource is intended to be created ***Content*:** The representation of the <*eventConfig*> resource for which the attributes are described in clause 9.6.24 Other information in the Request message is defined according to clause 10.1.2
Processing at Originator before sending Request	The Originator shall be an AE. The Originator shall request to Create a new <*eventConfig*> resource by addressing to the <*statsConfig*> resource of a Hosting CSE
Processing at Receiver	The Receiver shall be an IN-CSE: • The Receiver shall check if the *eventID* is unique, and if not, provides a new value • The Receiver shall verify that the *eventEnd* time is greater than the *eventStart* time if these two attributes are present • The Receiver shall verify that the *dataSize* attribute is present and contains a value greater than zero if the *eventType* is set to "Storage based"
Information in Response message	According to clause 10.1.2
Processing at Originator after receiving Response	None
Exceptions	According to clause 10.1.2

10.2.11.7 Retrieve <*eventConfig*>

The RETRIEVE procedure shall be used for the Originator to retrieve the existing <*eventConfig*> resource from the Receiver.

The Originator shall be an AE that is allowed to retrieve configuration information available for AEs within an IN-CSE.

The Receiver shall be the IN-CSE containing the <*eventConfig*> resource.

Table 10.2.11.7-1: <*eventConfig*> RETRIEVE

<*eventConfig*> RETRIEVE	
Information in Request message	***From*:** ID of the Originator ***To*:** Address of the <*eventConfig*> resource or its attributes to be retrieved.
Processing at Originator before sending Request	The Originator shall request to obtain <*eventConfig*> resource information by using the RETRIEVE operation. The request shall address the specific <*eventConfig*> resource or its attributes of a Hosting CSE. The Originator shall be an AE
Processing at Receiver	According to clause 10.1.3
Information in Response message	According to clause 10.1.3
Processing at Originator after receiving Response	According to clause 10.1.3
Exceptions	According to clause 10.1.3

10.2.11.8 Update <*eventConfig*>

This procedure shall be used for updating an existing <*eventConfig*> resource.

The Originator shall be the AE that created the <*eventConfig*> resource. The same AE shall be able to update the resource.

The Receiver shall be the IN-CSE containing the <*eventConfig*> resource.

Table 10.2.11.8-1: *<eventConfig>* UPDATE

***<eventConfig>* UPDATE**	
Information in Request message	***From*:** ID of the Originator ***To*:** Address of the *<eventConfig>* resource to be updated ***Content*:** The Originator provides the attributes of *<eventConfig>* to be updated The Originator can update attributes under *<eventConfig>* to update event-based configuration for statistics collection
Processing at Originator before sending Request	According to clause 10.1.4
Processing at Receiver	According to clause 10.1.4
Information in Response message	According to clause 10.1.4
Processing at Originator after receiving Response	None
Exceptions	According to clause 10.1.4

10.2.11.9 Delete *<eventConfig>*

This procedure shall be used for deleting *<eventConfig>* resource.

The Originator shall be the AE that created the *<eventConfig>* resource.

The Receiver shall be the IN-CSE containing the *<eventConfig>* resource.

Table 10.2.11.9-1: *<eventConfig>* DELETE

***<eventConfig>* DELETE**	
Information in Request message	***From*:** ID of the Originator ***To*:** Address of the *<eventConfig>* resource to be deleted
Processing at Originator before sending Request	According to clause 10.1.5
Processing at Receiver	According to clause 10.1.5
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	None
Exceptions	According to clause 10.1.5

10.2.11.10 Create *<statsCollect>*

This procedure shall be used for the Originator to establish collection scenarios at the Receiver.

The collection scenarios are stored at the *<statsCollect>* resource. Multiple collection scenarios can be created based on one instance of *<statsConfig>*.

The Receiver shall be an IN-CSE. The Receiver shall validate whether the Originator has proper permissions for creating a *<statsCollect>* resource. Upon successful validation, create a new *<statsCollect>* resource with the provided attributes. The IN-CSE shall also create a unique *statsCollectID*.

Table 10.2.11.10-1: *<statsCollect>* CREATE

***<statsCollect>* CREATE**	
Information in Request message	***From*:** Identifier of the AE that initiates the Request ***To*:** The Address of the *<CSEBase>* where the *<statsCollect>* resource is intended to be Created ***Content*:** Contain the resource representation of *<statsCollect>* Other information in the Request message is defined according to clause 10.1.2
Processing at Originator before sending Request	The Originator shall be an AE that wants to set up the collection scenarios to an IN-CSE. The Originator shall request to Create a new *<statsCollect>* resource by addressing to the *<CSEBase>* resource of a Hosting CSE The Originator shall populate the attributes for the *<statsCollect>* resource as defined in clause 9.6.25, except for *statsCollectID*
Processing at Receiver	In addition to procedures defined in clause 10.1.2, the Receiver shall perform the following specific operations: • Create *statsCollectID* which shall be unique in the same service provider domain • Once a *<statsCollect>* resource instance is created and the *status* is "ACTIVE", the IN-CSE shall generate service statistics collection records when the conditions defined by the *<statsCollect>* are met
Information in Response message	According to clause 10.1.2
Processing at Originator after receiving Response	None
Exceptions	According to clause 10.1.2

10.2.11.11 Retrieve *<statsCollect>*

The RETRIEVE procedure shall be used for the Originator to retrieve the existing *<statsCollect>* resource from the Receiver.

The Originator shall be an AE that is allowed to retrieve the collection scenario information from the IN-CSE.

The Receiver shall be the IN- CSE containing the *<statsCollect>* resource.

Table 10.2.11.11-1: *<statsCollect>* RETRIEVE

***<statsCollect>* RETRIEVE**	
Information in Request message	***From*:** ID of the Originator ***To*:** Address of the *<statsCollect>* resource or its attribute to be retrieved
Processing at Originator before sending Request	The Originator shall request to obtain *<statsCollect>* resource information by using the RETRIEVE operation. The request shall address the specific *<statsCollect>* resource or its attributes of a Hosting CSE. The Originator shall be an AE
Processing at Receiver	According to clause 10.1.3
Information in Response message	According to clause 10.1.3
Processing at Originator after receiving Response	According to clause 10.1.3
Exceptions	According to clause 10.1.3

10.2.11.12 Update *<statsCollect>*

An UPDATE procedure on the *<statsCollect>* resource shall be used for the Originator to update chargeable scenarios at the Receiver.

The Originator shall be the AE that created the *<statsCollect>* resource. The same AE shall be able to update the resource.

The Receiver shall be the IN-CSE containing the *<statsCollect>* resource.

Table 10.2.11.12-1: *<statsCollect>* UPDATE

***<statsCollect>* UPDATE**	
Information in Request message	***From*:** ID of the Originator ***To*:** Address of the *<statsCollect>* resource to be updated ***Content*:** the Originator provides the attributes of *<statsCollect>* to be updated
Processing at Originator before sending Request	According to clause 10.1.4
Processing at Receiver	According to clause 10.1.4
Information in Response message	According to clause 10.1.4
Processing at Originator after receiving Response	None
Exceptions	According to clause 10.1.4

10.2.11.13 Delete *<statsCollect>*

This procedure shall be used for deleting *<statsCollect>* resource.

The Originator shall be the AE that created the *<statsCollect>* resource.

The Receiver shall be a CSE containing the *<statsCollect>* resource.

Table 10.2.11.13-1: *<statsCollect>* DELETE

<statsCollect> DELETE	
Information in Request message	***From*:** ID of the Originator ***To*:** Address of the *<statsCollect>* resource to be deleted
Processing at Originator before sending Request	According to clause 10.1.5
Processing at Receiver	According to clause 10.1.5
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	None
Exceptions	According to clause 10.1.5

10.2.11.14 Service Statistics Collection Record

When the Service Statistics Collection is supported, the Information Elements shall be generated according to table 10.2.11.14-1.

The contents of each Service statistics collection record are decided by the specific collection scenario that triggered the information recording.

Transfer of the Statistics Collection Records over the Mch reference point is not defined in the present document.

Table 10.2.11.14-1: Information Elements for Service Statistics Collection Record

Information Element	Mandatory/ optional	Description
statsCollectID	M	It is the unique ID that identifies a specific statistics collection scenario, which triggers information recording for a specific event.
collectingEntityID	M	This is the unique ID of the entity that collects the statistics. It can be an AE-ID or CSE-ID.
collectedEntityID	M	This is the unique ID of the entity whose service layer operation statistics are being collected. It can be an AE-ID or CSE-ID.
event	O	This indicates a specific event type in each record, such as timer based, data operation, storage triggering. It is only present if the *statModel* is "event based".
eventStart	O	The start time for the recording the M2M event record.
eventEnd	O	The end time for the recording the M2M event record.
transactionType	O	Specifies the detailed type of a transaction, such as CREATE, RETRIEVE, etc.
dataSize	O	Storage Memory in Kbytes, where applicable, to store data associated events with container related operations.
Vendor Specific Information	O	Defines Vendor specific information.

10.2.12 M2M service subscription management

10.2.12.1 Introduction

This clause describes the procedures for creation, retrieval, update and deletion of the M2M Service Subscription related resources (i.e. <*m2mServiceSubscriptionProfile*>, <*serviceSubscribedNode*> and <*serviceSubscribedAppRule*>). These resources are used to store M2M Service Subscription related information regarding an established contract between a M2M Service Provider and M2M Application Service Providers. The relationship between these three resource types is specified in clause 9.6.19.

10.2.12.2 Create *<m2mServiceSubscriptionProfile>*

This procedure shall be used for creating a <*m2mServiceSubscriptionProfile*> resource.

Table 10.2.12.2-1: *<m2mServiceSubscriptionProfile>* CREATE

***<m2mServiceSubscriptionProfile>* CREATE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***To:*** The Receiver or Hosting CSE shall be an IN-CSE ***Content:*** The resource content shall provide the information as defined in clause 9.6.19
Processing at Originator before sending Request	According to clause 10.1.2
Processing at Receiver	According to clause 10.1.2
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content:*** Address of the created <*m2mServiceSubscriptionProfile*> resource, according to clause 10.1.2
Processing at Originator after receiving Response	According to clause 10.1.2
Exceptions	According to clause 10.1.2

10.2.12.3 Retrieve <*m2mServiceSubscriptionProfile*>

This procedure shall be used for retrieving the attributes of a <*m2mServiceSubscriptionProfile*> resource.

Table 10.2.12.3-1: <*m2mServiceSubscriptionProfile*> RETRIEVE

<*m2mServiceSubscriptionProfile*> RETRIEVE	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***To:*** The Receiver or Hosting CSE shall be an IN-CSE ***Content:*** Void
Processing at Originator before sending Request	According to clause 10.1.3
Processing at Receiver	According to clause 10.1.3
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content*:** Attributes of the <*m2mServiceSubscriptionProfile*> resource as defined in clause 9.6.19
Processing at Originator after receiving Response	According to clause 10.1.3
Exceptions	According to clause 10.1.3

10.2.12.4 Update <*m2mServiceSubscriptionProfile*>

This procedure shall be used for updating the attributes of a <*m2mServiceSubscriptionProfile*> resource.

Table 10.2.12.4-1: <*m2mServiceSubscriptionProfile*> UPDATE

<*m2mServiceSubscriptionProfile*> UPDATE	
Information in Request message	All parameters defined in table 8.1.2-3 are applicable as indicate in the table with the specific details for: ***To*:** The Receiver or Hosting CSE shall be an IN-CSE ***Content*:** Attributes of the <*m2mServiceSubscriptionProfile*> resource as defined in clause 9.6.19 which need be updated, with the exception of the following that cannot be modified: • "*lastModifiedTime*"
Processing at Originator before sending Request	According to clause 10.1.4
Processing at Receiver	According to clause 10.1.4
Information in Response message	According to clause 10.1.4
Processing at Originator after receiving Response	According to clause 10.1.4
Exceptions	According to clause 10.1.4

10.2.12.5 Delete <*m2mServiceSubscriptionProfile*>

This procedure shall be used for deleting a <*m2mServiceSubscriptionProfile*> resource residing under a <m2mServiceSubscriptionProfile> resource.

Table 10.2.12.5-1: *<m2mServiceSubscriptionProfile>* DELETE

***<m2mServiceSubscriptionProfile>* DELETE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with specific details for: ***To*:** The Receiver or Hosting CSE shall be an IN-CSE
Processing at Originator before sending Request	According to clause 10.1.5
Processing at Receiver	According to clause 10.1.5
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	According to clause 10.1.5
Exceptions	According to clause 10.1.5

10.2.12.6 Create *<serviceSubscribedNode>*

This procedure shall be used for creating a *<serviceSubscribedNode>* resource which is sub-resource of *<m2mServiceSubscriptionProfile>* resource.

Table 10.2.12.6-1: *<serviceSubscribedNode>* CREATE

***<serviceSubscribedNode>* CREATE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***To:*** The Receiver or Hosting CSE shall be an IN-CSE ***Content:*** The resource content shall provide the information as defined in clause 9.6.20
Processing at Originator before sending Request	According to clause 10.1.2
Processing at Receiver	According to clause 10.1.2
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content:*** Address of the created *<serviceSubscribedNode>* resource, according to clause 10.1.2
Processing at Originator after receiving Response	According to clause 10.1.2
Exceptions	According to clause 10.1.2

10.2.12.7 Retrieve *<serviceSubscribedNode>*

This procedure shall be used for retrieving the attributes of a *<serviceSubscribedNode>* resource which is sub-resource of *<m2mServiceSubscriptionProfile>* resource.

Table 10.2.12.7-1: *<serviceSubscribedNode>* RETRIEVE

***<serviceSubscribedNode>* RETRIEVE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***To:*** The Receiver or Hosting CSE shall be an IN-CSE **Content:** Void
Processing at Originator before sending Request	According to clause 10.1.3
Processing at Receiver	According to clause 10.1.3
Information in Response message	All parameters defined in table 8.1.3-1 apply
Processing at Originator after receiving Response	According to clause 10.1.3
Exceptions	According to clause 10.1.3

10.2.12.8 Update <*serviceSubscribedNode*>

This procedure shall be used for updating the attributes of a <*serviceSubscribedNode*> resource which is sub-resource of <*m2mServiceSubscriptionProfile*> resource.

Table 10.2.12.8-1: <*serviceSubscribedNode*> UPDATE

<*serviceSubscribedNode*> UPDATE	
Information in Request message	All parameters defined in table 8.1.2-3 are applicable as indicate in the table with the specific details for: ***To:*** The Receiver or Hosting CSE shall be an IN-CSE ***Content:*** Attributes of the <*serviceSubscribedNode*> resource as defined in clause 9.6.16 which need be updated, with the exception of the following that cannot be modified: "*lastModifiedTime*"
Processing at Originator before sending Request	According to clause 10.1.4
Processing at Receiver	According to clause 10.1.4
Information in Response message	According to clause 10.1.4
Processing at Originator after receiving Response	According to clause 10.1.4
Exceptions	According to clause 10.1.4

10.2.12.9 Delete <*serviceSubscribedNode*>

This procedure shall be used for deleting a <*serviceSubscribedNode*> resource residing under a <*m2mServiceSubscriptionProfile*> resource.

Table 10.2.12.9-1: <*serviceSubscribedNode*> DELETE

<*serviceSubscribedNode*> DELETE	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***To*:** The Receiver or Hosting CSE shall be an IN-CSE
Processing at Originator before sending Request	According to clause 10.1.5
Processing at Receiver	According to clause 10.1.5
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	According to clause 10.1.5
Exceptions	According to clause 10.1.5

10.2.12.10 Create <*serviceSubscribedAppRule*>

This procedure shall be used for creating an <*serviceSubscribedAppRule*> resource. The information represented in the attributes of a <*serviceSubscribedAppRule*> resource impacts the Application Entity Registration procedure as outlined in clause 10.2.2.2. Instances of <*serviceSubscribedAppRule*> resources are associated with specific CSEs by linking to them via the *ruleLinks* attribute of a <*serviceSubscribedNode*> resource that contains the respective CSE-ID in its CSE-ID attribute.

Table 10.2.12.10-1: *<serviceSubscribedAppRule>* CREATE

***<serviceSubscribedAppRule>* CREATE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***To:*** The Hosting CSE shall be an IN-CSE. ***Content:*** The resource content shall provide the information as defined in clause 9.6.29.
Processing at Originator before sending Request	According to clause 10.1.2.
Processing at Receiver	According to clause 10.1.2.
Information in Response message	All parameters defined in table 8.1.3-1 apply.
Processing at Originator after receiving Response	According to clause 10.1.2.
Exceptions	According to clause 10.1.2.

10.2.12.11 Retrieve *<serviceSubscribedAppRule>*

This procedure shall be used for retrieving the representation of the *<serviceSubscribedAppRule>* resource.

Table 10.2.12.11-1: *<serviceSubscribedAppRule>* RETRIEVE

***<serviceSubscribedAppRule>* RETRIEVE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***To:*** The Hosting CSE shall be an IN-CSE. ***Content:*** void.
Processing at Originator before sending Request	According to clause 10.1.3.
Processing at Receiver	According to clause 10.1.3.
Information in Response message	All parameters defined in table 8.1.3-1 apply.
Processing at Originator after receiving Response	According to clause 10.1.3.
Exceptions	According to clause 10.1.3.

10.2.12.12 Update *<serviceSubscribedAppRule>*

This procedure shall be used for updating the attributes of the *<serviceSubscribedAppRule>* resource.

Table 10.2.12.12-1: *<serviceSubscribedAppRule>* UPDATE

***<serviceSubscribedAppRule>* UPDATE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***To:*** The Hosting CSE shall be an IN-CSE. ***Content:*** Attributes of the *<serviceSubscribedAppRule>* resource as defined in clause 9.6.29.
Processing at Originator before sending Request	According to clause 10.1.4.
Processing at Receiver	According to clause 10.1.4.
Information in Response message	All parameters defined in table 8.1.3-1 apply.
Processing at Originator after receiving Response	According to clause 10.1.4.
Exceptions	According to clause 10.1.4.

10.2.12.13 Delete *<serviceSubscribedAppRule>*

This procedure shall be used for deleting the *<serviceSubscribedAppRule>* resource with all related information.

Table 10.2.12.13-1: *<serviceSubscribedAppRule>* DELETE

***<serviceSubscribedAppRule>* DELETE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***To:*** The Hosting CSE shall be an IN-CSE.
Processing at Originator before sending Request	According to clause 10.1.5.
Processing at Receiver	According to clause 10.1.5.
Information in Response message	All parameters defined in table 8.1.3-1 apply.
Processing at Originator after receiving Response	According to clause 10.1.5.
Exceptions	According to clause 10.1.5.

10.2.13 Resource announcement

10.2.13.1 Introduction

This clause describes the procedures for announcing the original resource and de-announcing the announced resource. A resource may be announced from its Hosting CSE to one or more announcement target CSEs to inform the announcement target CSE(s) of the existence of the original resource. The announced resource also may be de-announced from the announcement target CSE(s). A limited set of attributes of original resource may be announced or de-announced in the resource announcement or de-announcement procedure.

10.2.13.2 Procedure for AE and CSE to initiate Creation of an Announced Resource

This clause describes the procedure for an AE or a CSE to initiate the creation of an announced resource.

Figure 10.2.13.2-1 depicts how creation of an announced resource is initiated (clause 10.2.13.2) and the announced resource is created on an announcement target CSE (clause 10.2.13.5).

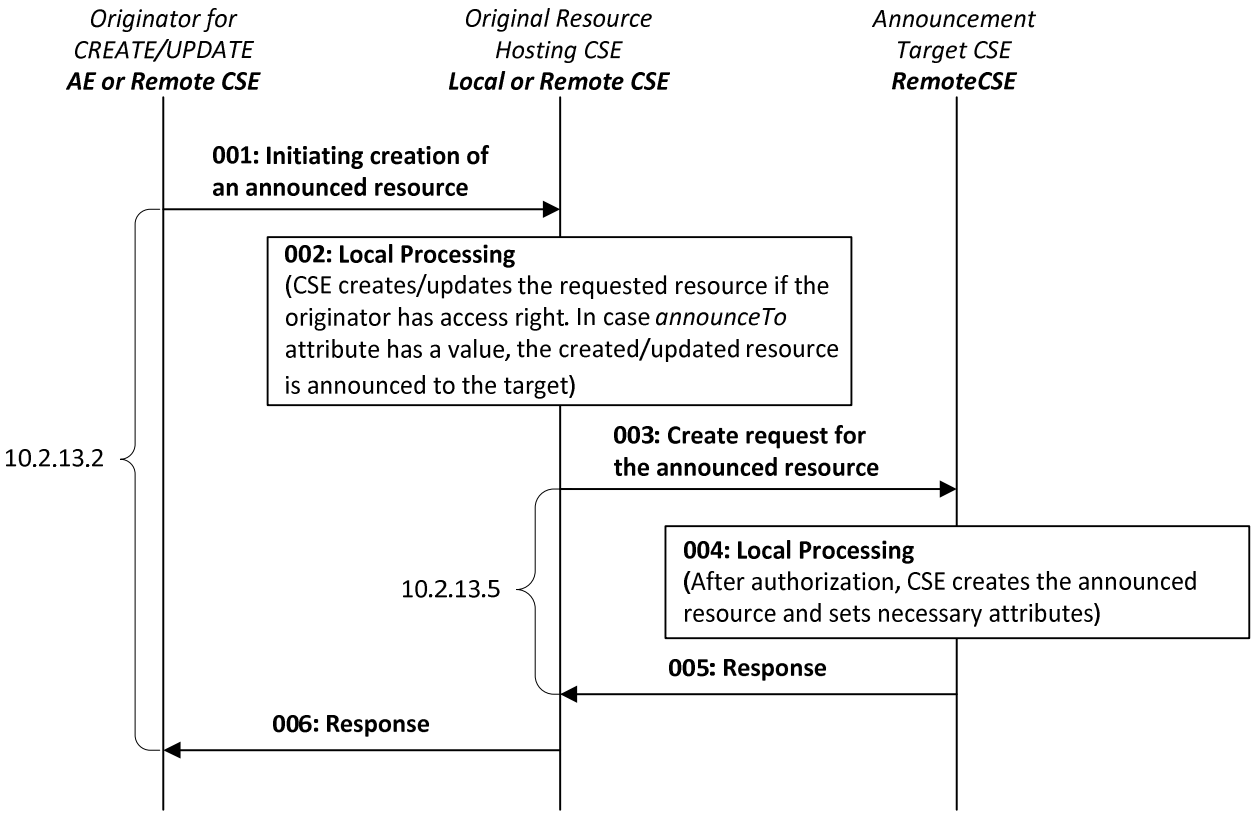


Figure 10.2.13.2-1: Announced resource CREATE procedures

The Originator of a Request for initiating resource announcement can be either an AE or a CSE. Two methods are supported for initiating the creation of an announced resource:

- CREATE: The Originator can initiate the creation of an announced resource during the creation of the original resource by providing *announceTo* attribute in the CREATE Request.
- UPDATE: The Originator can initiate the creation of an announced resource by using the UPDATE Request to update the *announceTo* attribute at the original resource.

Table 10.2.13.2-1: Initiate Resource Announcement: UPDATE or CREATE

***Initiate Resource Announcement:* CREATE or UPDATE**	
Information in Request message	All parameters defined in table 8.1.2-3 are applicable as indicated in that table. In addition, for the case of the CREATE procedure for a specific resource is described in clause 10.2. The Originator suggests the address(es) or the CSE-ID(s) to which the resource will be announced in the ***Content*** parameter.
Processing at the Originator before sending Request	***Content*****:** contains address where the resource needs to be announced (within *announceTo* attribute): • The Originator provides either the address(es) for the announced resource or the list of CSE-IDs of the remote CSEs where the original resource needs to be announced by including such information within the *announceTo* attribute of the UPDATE or CREATE Request.
Processing at the Receiver	Once the Originator has been successfully authorized, the Receiver (which shall be the original resource Hosting CSE) shall grant the Request after successful validation of the Request: • If the Request provides address(es) for the announced resource that are not already stored in the *announceTo* attribute or for a newly created *announceTo* attribute, the Receiver shall announce the resource to the provided address(es). • If the Request provides a list of CSE-IDs of the remote CSEs that are not already stored in the announceTo attribute or for the newly created announceTo attribute, the announced resource target location shall be the announced parent resource or the <CSEBaseAnnc> resource representing the Receiver at the announcement target CSE. In order to determine the target location, the Receiver shall: 1) Check if the parent resource is announced by checking the announceTo attribute of the parent resource and if so, create the announced resource as a child of the announced parent resource. 2) If the parent resource is not announced, the Receiver shall check if <CSEBase> is announced to the announcement target CSE by checking the *announceTo* attribute of <CSEBase>. If <CSEBase> is not announced, the Receiver shall create a <CSEBaseAnnc> to the announcement target CSE. The Receiver shall then create the announced resource as a child of the <CSEBaseAnnc> resource.
Information in Response message	On successful completion of resource announcement as in clause 10.2.3.5, the Receiver shall provide all parameters defined in table 8.1.3-1 that are applicable as indicated in that table in the Response message: • The Receiver shall provide the address(es) of the announced resource to the Originator which replace the given CSE-ID or URI in the content of the *announceTo* attribute in the original resource and by providing it in the UPDATE or CREATE Response message depending on the type of the Request.
Processing at Originator after receiving Response	According to clause 10.1.2 in case of CREATE Request. According to clause 10.1.4 in case of UPDATE Request.
Exceptions	All exceptions described in the basic procedures (clause 10.1.2) are applicable. If the parent resource of a contentInstance, or a timeSeriesInstance or a flexContainerInstance is not announced then the announcement shall fail.

10.2.13.3 Procedure at AE or CSE to Retrieve information from an Announced Resource

This clause describes the procedures that shall be use for an AE or a CSE to retrieve information about an announced resource or the corresponding original resource.

Figure 10.2.13.3-1 depicts how the announced resource is retrieved from an announcement target CSE.

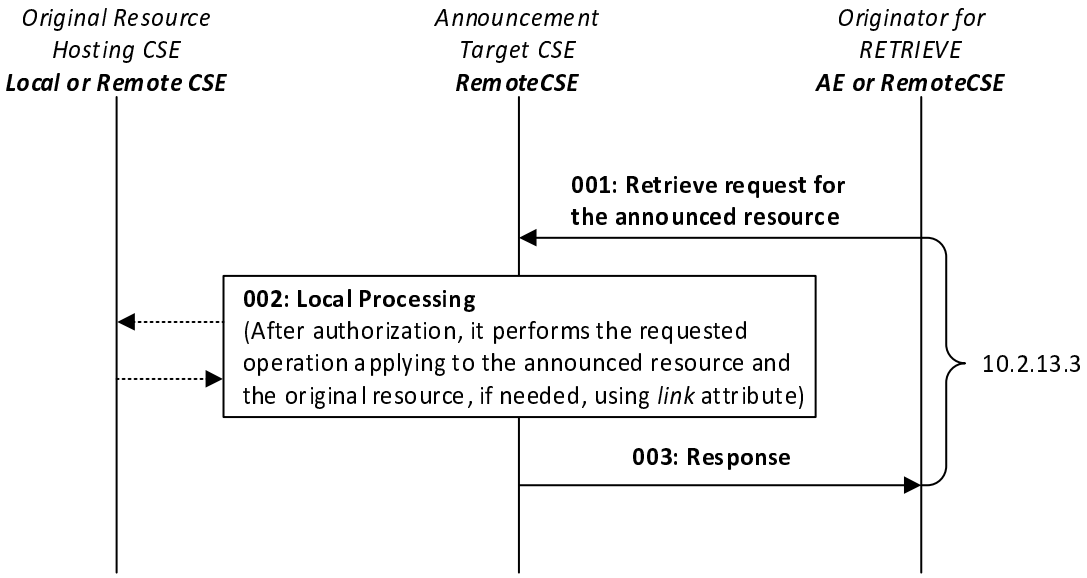


Figure 10.2.13.3-1: Announced resource RETRIEVE procedures

The Originator of a Request for initiating retrieval of information about a resource can be either an AE or a CSE. The Originator initiates this procedure by using RETRIEVE Request.

Table 10.2.13.3-1: Announced Resource Information Retrieval: RETRIEVE

***Resource Retrieval:* RETRIEVE**	
Information in Request message	Clause 8.1.2 specifies the information to be included in the Request message. Table 8.1.2-3 also describes the parameters that are applicable in the Request message: • Specifically, the ***To*** parameter is set to the address of the announced resource to be retrieved. • If a specific attribute is to be retrieved, the address of such attribute is included in the ***To*** parameter. • The Originator can specify one of the values for the optional ***Result Content*** parameter. • The Originator can request retrieval of the original resource by targeting the announced resource at the Hosting CSE by setting the ***Result Content*** parameter to the "original-resource".
Processing at the Originator before sending Request	The Originator can request retrieval of information from an announced resource at the Hosting CSE. Optionally, the Originator can request retrieval of the original resource by targeting the announced resource at the Hosting CSE by setting the ***Result Content*** parameter to the "original-resource".
Processing at the Receiver	Once the Originator has been successfully authorized, the Receiver (Hosting CSE) shall grant the Request after successful validation of the Request: • Information from the identified announced resource (at Hosting CSE) shall be returned to Originator via RETRIEVE Response, as described in clause 8.1.2. • If ***Result Content*** request message parameter set to "original-resource" is included in the Request message, the Receiver shall provide the representation of the original resource indicated by the *link* attribute in the announced resource. The Receiver shall retrieve the original resource to return the representation of the original resource to the Originator.
Information in Response message	Information from the identified announced resource (at Hosting CSE), or the original resource shall be returned to Originator via RETRIEVE Response, as described in clause 8.1.3.
Exceptions	All exceptions described in the basic procedure (clause 10.1.3) are applicable.

10.2.13.4 Procedure for AE and CSE to initiate Deletion of an Announced Resource

This clause describes the procedure that shall be used for an AE or a CSE (not the original resource Hosting CSE) to initiate the deletion of an announced resource.

Figure 10.2.13.4-1 depicts how deletion of an announced resource is initiated (clause 10.2.13.4) and the announced resource is deleted on an announcement target CSE (clause 10.2.13.6).

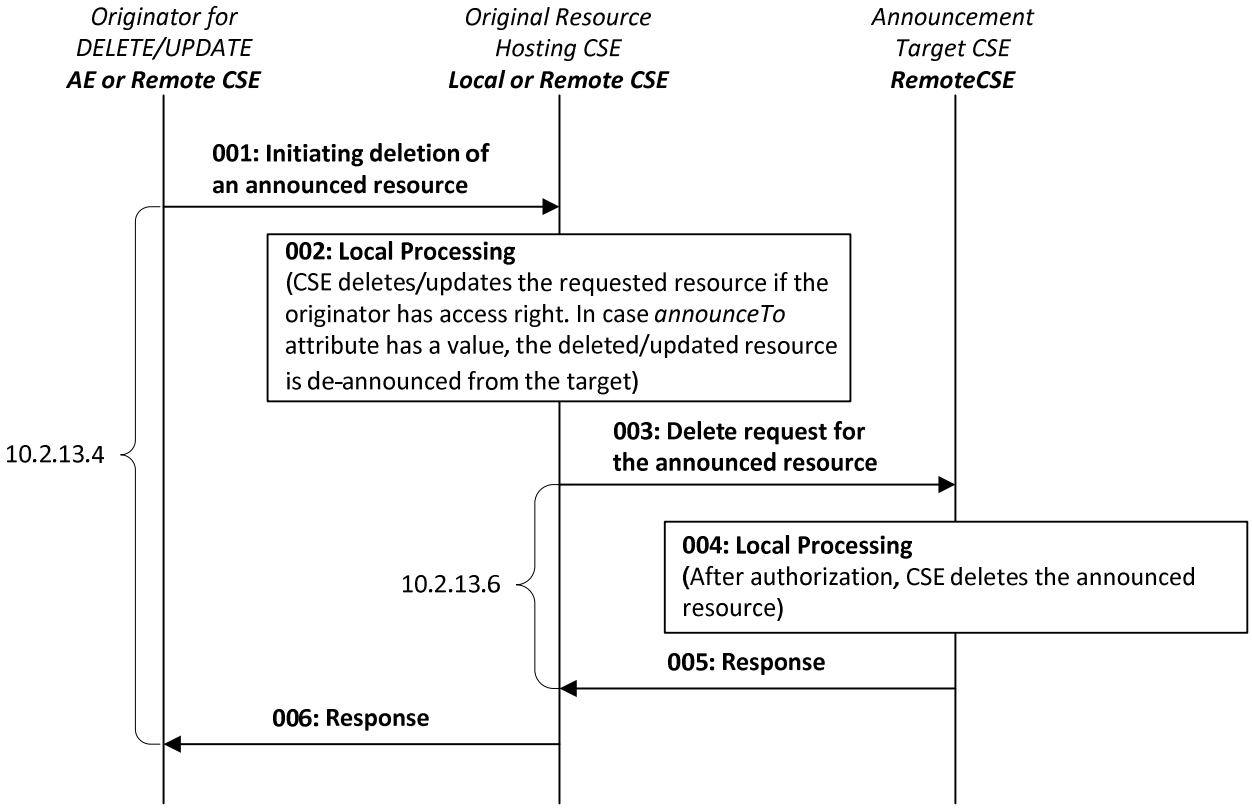


Figure 10.2.13.4-1: Announced resource DELETE procedures

The Originator of a Request for initiating resource de-announcement can be either an AE or a CSE. Two methods are supported for initiating resource de-announcement:

- UPDATE: The Originator can request to initiate the deletion of an announced resource by using UPDATE Request to the *announceTo* attribute at the original resource Hosting CSE.
- DELETE: Resource de-announcement (deletion) shall also be performed when the Originator deletes the original resource at the original resource Hosting CSE by using DELETE Request.

Table 10.2.13.4-1: Initiate Resource De-Announcement: UPDATE and DELETE

***Initiate Resource De-Announcement:* UPDATE or DELETE**	
Information in Request message	All parameters defined in table 8.1.2-3 are applicable as indicated in that table.
Processing at the Originator before sending Request	The Originator shall perform one of the following for the deletion of an announced resource: • The Originator shall request to update the *announceTo* attribute at the original resource Hosting CSE by providing new content of the *announceTo* attribute which does not include the CSE-IDs of the announcement target CSEs where the announced resource needs to be de-announced (deleted) by the UPDATE operation. • The Originator shall request to delete the *announceTo* attribute at the original resource Hosting CSE by sending UPDATE Request that sets the value of the *announceTo* attribute to NULL for the deletion of all announced resources. • For DELETE operation, the Originator shall include the resource address of the original resource Hosting CSE that needs to be deleted, in the DELETE Request. • ***Content*:** Void.
Processing at the Receiver	Once the Originator has been successfully authorized, the Receiver (which shall be the original resource Hosting CSE) shall grant the Request after successful validation of the Request. The Receiver shall be the resource Hosting CSE. On receiving the UPDATE or DELETE Request, the Receiver shall perform as follows: • For UPDATE Request, the Receiver shall request to delete the announced resource(s) whose address(es) is/are not included in the *announceTo* attribute of the request as per procedures in clause 10.2.13.6. • For DELETE Request, the Receiver shall request to delete all announced resources in the *announceTo* attribute as per procedures in clause 10.2.13.6.
Information in Response message	On successful completion of resource de-announcement procedure in clause 10.2.13.6, the Receiver knows that the announced resource has been deleted: • The Receiver shall provide confirmation of resource de-announcement to the Originator. • The content of the updated *announceTo* attribute shall be provided to the Originator to indicate the successfully deleted announced resource, if the *announceTo* attribute is not deleted by the Originator in the Request message.
Exceptions	All exceptions described in the basic procedure (clause 10.1.4) are applicable for UPDATE operation. All exceptions described in the basic procedure (clause 10.1.5) are applicable for DELETE operation.

10.2.13.5 Procedure for original resource Hosting CSE to Create an Announced Resource

This clause explains the resource announcement procedure that shall be used by the original resource Hosting CSE to announce the original resource to the remote CSE(s).

See figure 10.2.13.2-1 for the graphical explanation.

The Originator of this Request shall be the original resource Hosting CSE. The Originator shall request to create the announced resource by using CREATE Request.

Table 10.2.13.5-1: Resource Hosting CSE to Announce Resource: CREATE

***Resource Announcement:* CREATE**	
Information in Request message	All parameters defined in table 8.1.2-3 are applicable as indicated in that table. ***Content*:** contains MA attributes and OA attributes that are included in *announcedAttribute* attribute.
Processing at the Originator before sending Request	Other details for the information in the Request message shall be as follows: • Attributes marked with MA and attributes marked with OA that are included in the *announcedAttribute* attribute at the original resource shall be provided in the CREATE Request. Such attributes shall have the same value as for the original resource. • *resourceType* which shall be set to the appropriate tag that identifies the *<Annc>* resource. • *expirationTime* provided by the Originator equal to the one for the original resource. • The *link* attribute of the announced resource shall have the address of the original resource in SP-relative Resource-ID format or Absolute Resource-ID format. • The *labels* attribute of the announced resource shall have the same value as for the original resource. • The *accessControlPolicyIDs* attribute shall always be provided in the CREATE Request even if it is not present in the original resource. In this case the original resource shall include *accessControlPolicyIDs* from its parent resource or from the local policy at the original resource, as needed. • *accessControlPolicyIDs* and *labels* attributes, if present at the original resource, shall be provided by the original resource Hosting CSE in the CREATE Request. Such attributes shall have the same value at the original resource and at the announced resource(s).
Processing at the Receiver	Once the Originator has been successfully authorized, the Receiver shall grant the Request after successful validation of the Request. The Receiver shall perform as follows: • The basic procedure (clause 10.1.2) for the Receiver of the CREATE Request apply. • The created announced resource shall include the common attributes specified in clause 9.6.26.1. The created announced resource shall contain the additional attributes that are provided by the Originator; i.e. attributes marked with MA and the attributes marked with OA that are included in the *announcedAttribute* attribute. • The created announced resource shall set the *accessControlPolicyIDs* attribute to the value received in the Request message, and shall set the *labels* attribute (if present) and the *link* attribute to the value received in the Request message. • Respond to the Originator with the CREATE Response. In this Response, the address of the successfully announced resource shall be provided.
Information in Response message	All parameters defined in table 8.1.3-1 are applicable as indicated in that table with the specific details for: ***Content:*** address where the announced resource is created according to clause 10.1.2.
Processing at Originator after receiving Response	The Originator after receiving the Response from the Receiver shall perform the following steps: • If the announced resource has been successfully created, the *announceTo* attribute of the original resource shall be updated to include the address for the successfully announced resource at the Receiver. The *announcedAttribute* attribute shall be updated as well to represent the successfully announced attributes as received in the Response. • For the attributes marked as MA and for the attributes marked as OA that are included in the *announcedAttribute* attribute, the Originator shall further take the responsibility to keep their values synchronized at the announced resource by using UPDATE operation (clause 10.1.4).
Exceptions	All exceptions described in the basic procedures (clause 10.1.2) are applicable.

10.2.13.6 Procedure for original resource Hosting CSE to Delete an Announced Resource

This clause explains the procedure that shall be used for deleting an announced resource (i.e. the resource de-announcement). This procedure shall be used by the original resource Hosting CSE for deleting the announced resource that resides at the remote CSE.

The Originator of this Request shall be the original resource Hosting CSE.

Table 10.2.13.6-1: Resource Hosting CSE to De-Announce Resource: DELETE

Resource De-Announcement: **DELETE**	
Information in Request message	All parameters defined in table 8.1.2-3 are applicable as indicate in that table. ***From*****:** Identifier of the CSE that initiates the Request. ***To*****:** The address where announced resource needs to be deleted.
Processing at the Originator before sending Request	The Originator shall request to delete an announced resource by using the DELETE Request. ***To:*** Parameter provides an address that identifies the announced resource to be deleted.
Processing at the Receiver	If the value of the *From* parameter in Request message is identical with the CSE-ID included in the *link* attribute in the announced resource, the Receiver shall grant the Request after successful validation of the Request: • Delete the announced resource identified by the ***To*** parameter in the Request, as per basic procedure in clause 10.1.5. • Respond to the Originator with the appropriate DELETE Response, as per basic procedure in clause 10.1.5.
Information in Response message	No change from the basic procedure (clause 10.1.5).
Processing at Originator after receiving Response	The Originator after receiving the Response from the Receiver shall: • If the announced resource is successfully deleted, the *announceTo* attribute in the original resource shall be updated to delete the address for the deleted announced resource.
Exceptions	All exceptions described in the basic procedures (clause 10.1.5) are applicable.

10.2.13.7 Procedure for AE and CSE to initiate the Creation of an Announced Attribute

This clause describes the procedure that shall be used for an AE and CSE (not the original resource Hosting CSE) to initiate the creation of an announced attribute (attribute announcement).

The Originator of a Request, for initiating attribute announcement, can be either AE or CSE (not the original resource Hosting CSE).

Table 10.2.13.7-1: Initiate Creation of Announced Attributes

***Initiate Attribute Announcement:* UPDATE**	
Information in Request message	Parameters defined in table 8.1.2-3 that are applicable for UPDATE. ***Content*** parameter includes the names of the attributes to be announced.
Processing at the Originator before sending Request	The Originator shall request attribute announcement by updating the *announcedAttribute* attribute at the original resource: • The Originator shall update the *announcedAttribute* attribute at the original resource by adding the attribute name for the attribute that needs to be announced by using the UPDATE Request. Only the attributes marked with OA can be announced to remote announced resources.
Processing at the Receiver	Once the Originator has been successfully authorized, the Receiver, which shall be the original resource Hosting CSE, shall grant the Request after successful validation of the Request. • The attributes received in the Request, which are not marked as OA, are invalid. • The attributes received in the Request, which are not present in the original resource structure, are invalid. • If some attributes received in the Request do not already exist in the *announcedAttribute* attribute, the Receiver shall announce such attributes to all announced resources listed in the *announceTo* attribute as per procedures in clause 10.2.13.9. On successful announcement of attributes as per procedures in clause 10.2.13.9, the Receiver shall perform the following: • The Receiver shall respond to the Originator (requesting AE/CSE) with UPDATE Response as specified in clause 10.1.4. The content of the announced attributes can be provided in such Response.
Information in Response message	Parameters defined in table 8.1.3-1 that are applicable.
Exceptions	All exceptions described in the basic procedures (clause 10.1.4) are applicable.

10.2.13.8 Procedure for AE and CSE to initiate the Deletion of an Announced Attribute

This clause describes the procedure that shall be used for an AE and CSE (not the original resource Hosting CSE) to initiate the deletion of announced attributes (attribute de-announcement).

The Originator of a Request, for initiating attribute de-announcement, can be either AE or CSE (not the original resource Hosting CSE).

Table 10.2.13.8-1: Initiate Deletion of Announced Attributes

***Initiate Attribute De-Announcement:* UPDATE**	
Information in Request message	Parameters defined in table 8.1.2-3 that are applicable for UPDATE. ***Content*** parameter does not include the names of the attributes to be de-announced.
Processing at the Originator before sending Request	The Originator shall request attribute de-announcement by updating the *announcedAttribute* attribute at the original resource as follows: • The Originator shall update the *announcedAttribute* attribute at the original resource by deleting the attribute name for the attribute that needs to be de-announced by using the UPDATE Request. Only the attributes marked with OA can be de-announced to remote announced resources.
Processing at the Receiver	Once the Originator has been successfully authorized, the Receiver, which shall be the original resource Hosting CSE, shall grant the Request after successful validation of the Request: • The attributes received in the Request, which are not marked as OA, are invalid. • If some attributes that exist in the *announcedAttribute* attribute are not received in the Request (i.e. attributes that need to be deleted by the UPDATE Request), the Receiver shall de-announce such attributes to all announced resources listed in the *announceTo* attributes as per procedure in clause 10.2.3.10. On successful de-announcement of all attributes as per procedures in clause 10.2.3.10, the Receiver shall perform the following: • The Receiver shall respond to the Originator (requesting AE/CSE) with UPDATE Response as specified in clause 10.1.4. The names of the de-announced attributes can be provided in such Response.
Information in Response message	Parameters defined in table 8.1.3-1 that are applicable.
Exceptions	All exceptions described in the basic procedures (clause 10.1.4) are applicable.

10.2.13.9 Procedure for original resource Hosting CSE for Announcing Attributes

This clause describes procedure that shall be used by the original resource Hosting CSE to create announced attributes at the remote announced resources (i.e. the attribute announcement).

The Originator of this Request shall be the original resource Hosting CSE.

Table 10.2.13.9-1: Original Resource Hosting CSE to Announce Attribute: UPDATE

***Attribute Announcement:* UPDATE**	
Information in Request message	Information described for the Originator of the UPDATE Request as in clause 10.1.4. ***Content:*** Parameter includes the names of the attributes to be announced and their values.
Processing at the Originator before sending Request	The Originator shall request to create attributes at the announced resources by using the UPDATE Request as specified in clause 10.1.4. Only parameters marked with OA can be announced.
Processing at the Receiver	Once the Originator has been successfully authorized, the Receiver (CSE hosting announced resource) shall grant the Request after successful validation of the Request. The Receiver shall perform as follows: • Create announced attributes at the announced resource as per procedures in clause 10.1.4. The initial value for the announced attributes shall use the same value as with the original resource. • Respond to the Originator with UPDATE Response as in clause 10.1.4.
Information in Response message	Parameters defined in table 8.1.3-1 that are applicable.
Processing at Originator after receiving Response	Originator after receiving the Response from the Receiver shall perform the following steps: • If the announced attributes have been successfully created, the *announcedAttribute* attribute shall be updated to include the attribute names for the successfully announced attributes. • For the newly announced attributes in the *announcedAttribute* attribute, the Originator shall take the responsibility to keep their values synchronized at the announced resources by using UPDATE operation as in clause 10.1.4.
Exceptions	All exceptions described in the basic procedures (clause 10.1.4) are applicable.

10.2.13.10 Procedure for original resource Hosting CSE for De-Announcing Attributes

This clause describes procedure that shall be used by the original resource Hosting CSE to remove announced attributes at remote announced resources (i.e. the attribute de-announcement).

The Originator of this Request shall be the original resource Hosting CSE.

Table 10.2.13.10-1: Original Resource Hosting CSE to De-Announce Attribute: UPDATE

***Attribute De-Announcement:* UPDATE**	
Information in Request message	Information described for the Originator of the UPDATE Request as in clause 10.1.4. ***Content:*** Parameter includes the names of the attributes to be deleted (de-announced) with their values set to NULL.
Processing at the Originator before sending Request	The Originator shall request to delete the announced attributes by using the UPDATE Request as specified in clause 10.1.4. Only attributes marked as OA can be de-announced: ***Content:*** Parameter in the UPDATE Request shall provide the names of the attributes to be de-announced by setting their values set to NULL.
Processing at the Receiver	If the value of the *From* parameter in Request message is identical with the CSE-ID included in the *link* attribute in the announced resource, the Receiver (CSE hosting announced resource) shall grant the Request after successful validation of the Request. The Receiver shall perform as follows: • Delete the de-announced attributes identified by the ***Content*** parameter in the UPDATE Request as per procedures in clause 10.1.4. • Respond to the Originator with the appropriate UPDATE Response as in clause 10.1.4.
Information in Response message	Parameters defined in table 8.1.3-1 that are applicable.
Processing at Originator after receiving Response	The Originator after receiving the Response from the Receiver shall perform the following steps: • If the attributes have been successfully removed, the *announcedAttribute* attribute shall be updated so as to remove the attribute names for the successfully de-announced attributes.
Exceptions	All exceptions described in the basic procedures (clause 10.1.4) are applicable.

10.2.13.11 Procedure for original resource Hosting CSE for Updating Attributes

This clause describes procedure that shall be used by the original resource Hosting CSE to update announced attributes at the remote announced resources. The Originator of this Request shall be the original resource Hosting CSE.

Table 10.2.13.11-1: Original Resource Hosting CSE to Update Attribute: UPDATE

***Attribute Update:* UPDATE**	
Information in Request message	Information described for the Originator of the UPDATE Request as in clause 10.1.4. ***Content:*** Parameter includes the names of the attributes to be updated with their target values.
Processing at the Originator before sending Request	The Originator shall request to update the announced attributes by using the UPDATE Request as specified in clause 10.1.4. Attributes marked as MA or OA can be updated: ***Content:*** Parameter in the UPDATE Request shall provide the names of the attributes to be updated by setting their target values.
Processing at the Receiver	If the value of the *From* parameter in Request message is identical with the CSE-ID included in the *link* attribute in the announced resource, the Receiver (CSE hosting announced resource) shall grant the Request after successful validation of the Request. The Receiver shall perform as follows: • Update the target attributes identified by the ***Content*** parameter in the UPDATE Request as per procedures in clause 10.1.4. • Respond to the Originator with the appropriate UPDATE Response as in clause 10.1.4.
Information in Response message	Parameters defined in table 8.1.3-1 that are applicable.
Exceptions	All exceptions described in the basic procedures (clause 10.1.4) are applicable.

10.2.13.12 Notification Procedure targeting an AE Announced Resource

This clause describes handling of notifications received at an <AEAnnc> resource Hosting CSE.

Table 10.2.13.12-1: Notification Procedure for AE Announced Resource

Notification Procedure for AE Announced Resource	
Information in Request message	Notification message made according to clause 10.2.10
Processing at the Originator before sending Request	According to clause 10.1.6
Processing at the Receiver	<*AEAnnc*> hosting CSE shall forward received notification message to original resource Hosting CSE targeting original <*AE*> resource when <*AE*> resource is available
Information in Response message	According to clause 10.1.6
Processing at Originator after receiving Response	According to clause 10.1.6
Exceptions	According to clause 10.1.6

10.2.14 Semantics management

Semantics management is performed for the purpose of leveraging CRUD operations on semantic-related resources to enable semantic functionalities in service layer (e.g. enhancing the meaning of resources and data in the system).

Table 10.2.14-1 summarizes the specialized resource types defined for the purpose of providing semantic enablement, providing references to the resource type definition clause. The table also provides references to the corresponding CRUD procedures.

Table 10.2.14-1: Specialized resource types for semantic management

Resource type	Description	Resource Type Reference	CRUD procedures
<*semanticDescriptor*>	Resource type used for annotating resources with semantic descriptions, providing the means for resource discovery in a semantically- aware fashion and for semantic queries	9.6.30	[14] clause 6.1
<*semanticFanOutPoint*>	Virtual resource type used to form an overall graph based on the content of the semantic descriptors associated with the members of the group, for the purpose of performing semantic resource discovery and semantic query	9.6.14a	[14] clause 6.2
<*semanticMashupJobProfile*>	Resource type describing the profile and necessary information (e.g. input parameters, member resources, mashup function, output parameters) required for a specific mashup service	9.6.53	[14] clause 6.3
<*semanticMashupInstance*>	Resource type describing a mashup instance based on mashup request and implementing the semantic mashup function. Each instance corresponds to a semantic mashup job profile	9.6.54	[14] clause 6.4
<*mashup*>	Virtual resource type used for triggering a calculation and generation of the mashup result based on its parent resource.	9.6.55	[14] clause 6.5
<*semanticMashupResult*>	Resource type storing the result generated when it executes a semantic mashup operation	9.6.56	[14] clause 6.6
<*ontology*>	Resource type storing the representation of an ontology	9.6.51	[14] clause 6.8
<*ontologyRepository*>	Resource type for storage of all ontology representations	9.6.50	[14] clause 6.7
<*semanticValidation*>	Virtua resource type used to trigger validation of semantic content	9.6.52	[14] clause 6.9

Table 10.2.14-2 summarizes the specialized procedures defined for the purpose of providing semantic enablement, providing references to the oneM2M TS-0034 [14] clauses where the detail procedural descriptions are provided.

Table 10.2.14-2: Specialized procedures and functions for semantic management

Procedure	Description	Reference
Access Control for Semantic Content	Functionality enabling the use of access control information applicable to resources for accessing RDF triple content when executing semantic operations.	[14] clause 7.2
Semantics Annotation	Functionality for providing semantic description for resources and content.	[14] clause 7.3
Semantic Filtering and Discovery	Procedures for the discovery of resources and semantic information, respectively, based on the semantic annotation.	[14] clause 7.4
Semantics Mash-up	Procedures enabling the creation, execution and result retrieval of functions based on semantic mashup.	[14] clause 7.7
Semantic Query	Procedures for directly retrieving both explicitly and implicitly derived information based on syntactic, semantic and structural information contained in semantic content data (such as RDF triples). The result of a semantic query is the semantic information/knowledge for answering/matching the query.	[14] clause 7.5
Semantic Validation	Procedures enabling the validation of semantic content.	[14] clause 7.10
Ontology Management	Procedures enabling the use and management of ontologies.	[14] clause 7.9

10.2.15 3GPP network interworking

10.2.15.1 Introduction

This clause introduces the functionality supporting 3GPP interworking and 3GPP CIoT features. The oneM2M system leverages the IoT related features and services that 3GPP added in releases 10 through 14. At the field Node level, features and services may be accessed by an ADN-AE, MN-CSE, or an ASN-CSE that is hosted on a UE. IN-CSEs are also able to access services that are exposed by a mobile network operator. Detailed analysis of interworking features and specification for the related functionality are provided in ETSI TS 118 126 [15].

Table 10.2.15.1-1 summarizes the specialized procedures defined for the purpose of providing 3GPP Interworking, providing references to other clauses where the detail procedural descriptions are provided.

Table 10.2.15.1-1: Specialized procedures and functions for 3GPP Interworking

Procedure	Description	References
Device Triggering	Functionality enabling nodes in the infrastructure domain to initiate sending of information to a node in the field domain	See [15] clause 7.5
Group Message Delivery	Functionality enabling communication to a group of nodes in the field domain	See [15] clause 7.7
Configuration of Traffic Patterns	Procedures enabling the oneM2M System to provide service layer information about the communication patterns of oneM2M devices to the Underlying Network	See [15] clause 7.6
Background Data Transfer	Procedures enabling nodes in the infrastructure domain to negotiate with the Underlying Network a background data transfer for a set of field nodes	See [15] clause 7.10
Monitoring events	Procedures enabling nodes in the infrastructure domain to request monitoring of Underlying Network evenest	See [15] clause 7.4
Change the chargeable party	Procedures enabling the infrastructure nodes to request a change of chargeable party for the traffic flows	See [15] clause 7.11

10.2.15.2 Create <*triggerRequest*>

This procedure shall be used for creating a <*triggerRequest*> resource.

Table 10.2.15.2-1: <*triggerRequest*> CREATE

<*triggerRequest*> CREATE	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content:*** The resource content shall provide the information as defined in clause 9.6.49.
Processing at Originator before sending Request	According to clause 10.1.2
Processing at Receiver	According to clause 10.1.2 with the following modifications: The CSE shall: • The trigger payload sent in the Trigger request shall be serialized based on the *contentSerialization* attribute of the <AE> or <remoteCSE> resource of the targeted entity. • Determine which NSE to send the trigger request to. The CSE may determine which NSE based on locally provisioned information or based on a DNS lookup of the M2M-Ext-ID. If an NSE cannot be determined, the IN-CSE sets the *triggerStatus* attribute to ERROR-NSE-NOT-FOUND. Otherwise, the CSE sets the *triggerStatus* attribute to PROCESSING. • The CSE shall submit a trigger request to the appropriate NSE using the appropriate Mcn protocol. The message shall contain information needed by the NSE to generate a trigger request for the corresponding underlying network. For example, for a 3GPP trigger request the required information needed within the trigger request message is captured in ETSI TS 118 126 [15]. • Upon receipt of trigger response(s) from the NSE, the CSE shall set the *triggerStatus* attribute of the <*triggerRequest*> resource. If the CSE receives a confirmation from the NSE that the trigger was accepted, the CSE shall set the *triggerStatus* attribute to TRIGGER-SUBMITTED. If the CSE receives an indication that the trigger request was successfully delivered, the CSE shall set the *triggerStatus* attribute to TRIGGER-DELIVERED. If the CSE receives an indication that the trigger request was not accepted or the delivery was not successful, the CSE shall set the *triggerStatus* attribute to TRIGGER-FAILED.
Information in Response message	According to clause 10.1.2
Processing at Originator after receiving Response	According to clause 10.1.2
Exceptions	According to clause 10.1.2

10.2.15.3 Retrieve <*triggerRequest*>

This procedure shall be used for retrieving the attributes of a <*triggerRequest*> resource.

Table 10.2.15.3-1: <*triggerRequest*> RETRIEVE

<*triggerRequest*> RETRIEVE	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.3
Processing at Receiver	According to clause 10.1.3
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content:*** The resource content shall provide the information as defined in clause 9.6.49
Processing at Originator after receiving Response	According to clause 10.1.3
Exceptions	According to clause 10.1.3

10.2.15.4 Update <*triggerRequest*>

This procedure shall be used for updating attributes of a <*triggerRequest*> resource.

Table 10.2.15.4-1: <*triggerRequest*> UPDATE

<*triggerRequest*> UPDATE	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content:*** The resource content shall provide the information as defined in clause 9.6.49.
Processing at Originator before sending Request	According to clause 10.1.4 The Originator determines that a trigger that is still being processed needs to be updated (i.e. replaced). The Originator initiates a device trigger replace by updating the <triggerRequest> resource.
Processing at Receiver	According to clause 10.1.4 with the following modifications: The CSE shall: • Check whether the trigger request can be updated or not by checking the *triggerStatus*. If the *triggerStatus* is PROCESSING, the CSE shall continue to process the UPDATE request. Otherwise, the CSE shall return an error response to the Originator and shall not update the *triggerStatus* attribute. • Determine which NSE to send the trigger update request to. The CSE may determine which NSE based on locally provisioned information or based on a DNS lookup of the M2M-Ext-ID. If an NSE cannot be determined, the IN-CSE sets the *triggerStatus* attribute to ERROR-NSE-NOT-FOUND. • The CSE shall submit a trigger update request to the appropriate NSE using the appropriate Mcn protocol. The message shall contain information needed by the NSE to update the trigger request for the corresponding underlying network. For example, for a 3GPP trigger update request the required information needed within the trigger request message is captured in ETSI TS 118 126 [15]. • Upon receipt of trigger update response(s) from the NSE, the CSE shall determine whether to set the *triggerStatus* attribute of the <*triggerRequest*> resource. If the CSE receives a confirmation from the NSE that the trigger update was accepted, the CSE shall update the applicable <*triggerRequest*> attributes included in the request and set the *triggerStatus* attribute to TRIGGER-SUBMITTED. If the CSE receives an indication that the trigger update request was not accepted, the CSE shall return an error response to the Originator and shall not update the <*triggerRequest*> resource.
Information in Response message	According to clause 10.1.4
Processing at Originator after receiving Response	According to clause 10.1.4
Exceptions	According to clause 10.1.4

10.2.15.5 Delete <*triggerRequest*>

This procedure shall be used for deleting a <*triggerRequest*> resource.

Table 10.2.15.5-1: <*triggerRequest*> DELETE

<*triggerRequest*> DELETE	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.5 The Originator determines that a trigger, that is still being processed needs to be deleted (i.e. recalled). The Originator initiates a device trigger recall by deleting the <triggerRequest> resource.
Processing at Receiver	According to clause 10.1.5 with the following modifications: The CSE shall: • Check whether the trigger request can be recalled or not by checking the *triggerStatus*. If the *triggerStatus* is PROCESSING, the CSE shall continue to process the DELETE request. Otherwise, the CSE shall return an error response to the Originator and shall not update the *triggerStatus* attribute. • Determine which NSE to send the trigger recall request to. The CSE may determine which NSE based on locally provisioned information or based on a DNS lookup of the M2M-Ext-ID. If an NSE cannot be determined, the IN-CSE sets the *triggerStatus* attribute to ERROR-NSE-NOT-FOUND. • The CSE shall submit a trigger recall request to the appropriate NSE using the appropriate Mcn protocol. The message shall contain information needed by the NSE to recall the trigger request for the corresponding underlying network. For example, for a 3GPP trigger recall request the required information needed within the trigger request message is captured in ETSI TS 118 126 [15]. • Upon receipt of trigger recall response(s) from the NSE, the CSE shall determine whether to set the *triggerStatus* attribute of the <*triggerRequest*> resource. If the CSE receives a confirmation from the NSE that the trigger recall was accepted, the CSE shall delete the applicable <*triggerRequest*> resource and return a successful response to the Originator. If the CSE receives an indication that the trigger recall request was not accepted, the CSE shall return an error response to the Originator and shall not update the <*triggerRequest*> resource.
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	According to clause 10.1.5
Exceptions	According to clause 10.1.5

10.2.16 Procedure for Managing Change in AE Registration Point

10.2.16.1 Procedure at IN-CSE

The IN-CSE may determine that an AE has changed registration point either by:

- Observing the creation on an <*AEAnnc*> resource with an **AE-ID-Stem** that it had previously assigned for a different Registrar CSE (Case e) of clause 10.2.2.2).

- Receiving a NOTIFY request from a Registrar CSE whose content includes the SP-relative-Resource-ID before and after the change in registration point (Case f) of clause 10.2.2.2).

In both cases, the IN-CSE shall send a NOTIFY request to the CSEs, so that these may update the references to the <AE> resources for the AE that has changed its registration point. If the IN-CSE maintains an <AEContactList> resource, the IN-CSE shall determine which CSEs are effected, and shall send the NOTIFY request only to these. If the IN-CSE does not maintain an <AEContactList> resource, the IN-CSE shall send the NOTIFY request to all CSEs. The ***Content*** parameter of the NOTIFY request shall contain the SP-relative-Resource-ID at the prior registration point and at the new registration point.

10.2.16.2 Procedure at any CSE

Upon receiving a NOTIFY request regarding a change in AE registration point:

- if the receiving CSE hosts references to the SP-Relative-Resource-ID (e.g. in Announce links, Notification targets, group Member IDs, <accessControlPolicy> resource *OriginatorID* lists) tied to the prior AE registration point, the receiving CSE shall update these to refer to the new AE registration point.
- if the receiving CSE hosts the registration of the prior AE registration point, the receiving CSE shall update the status of this registration to "INACTIVE".

10.2.17 Schedule Management

10.2.17.1 Introduction

This clause describes the procedures for creation, retrieval, update and deletion of the <*schedule*> resource.

10.2.17.2 Create <*schedule*>

This procedure shall be used for creating an <*schedule*> resource.

Table 10.2.17.2-1: <*schedule*> CREATE

<*schedule*> CREATE	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content:*** The resource content shall provide the information as defined in clause 9.6.9
Processing at Originator before sending Request	According to clause 10.1.2
Processing at Receiver	According to clause 10.1.2
Information in Response message	According to clause 10.1.2
Processing at Originator after receiving Response	According to clause 10.1.2
Exceptions	According to clause 10.1.2

10.2.17.3 Retrieve <*schedule*>

This procedure shall be used for retrieving the representation of the <*schedule*> resource.

Table 10.2.17.3-1: <*schedule*> RETRIEVE

<*schedule*> RETRIEVE	
Information in Request message	All parameters defined in table 8.1.2-3
Processing at Originator before sending Request	According to clause 10.1.3
Processing at Receiver	According to clause 10.1.3
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content***: attributes of the <*schedule*> resource as defined in clause 9.6.9
Processing at Originator after receiving Response	According to clause 10.1.3
Exceptions	According to clause 10.1.3

10.2.17.4 Update <*schedule*>

This procedure shall be used for updating the attributes and the actual data of an <*schedule*> resource.

Table 10.2.17.4-1: <*schedule*> UPDATE

<*schedule*> UPDATE	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content*:** attributes of the <*schedule*> resource as defined in clause 9.6.9 which need be updated
Processing at Originator before sending Request	According to clause 10.1.4
Processing at Receiver	According to clause 10.1.4
Information in Response message	According to clause 10.1.4
Processing at Originator after receiving Response	According to clause 10.1.4
Exceptions	According to clause 10.1.4

10.2.17.5 Delete <*schedule*>

This procedure shall be used for deleting the <*schedule*> resource with all related information.

Table 10.2.17.5-1: <*schedule*> DELETE

<*schedule*> DELETE	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.5
Processing at Receiver	According to clause 10.1.5
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	According to clause 10.1.5
Exceptions	According to clause 10.1.5

10.2.18 Transaction Management

10.2.18.1 Introduction

This clause describes procedures for managing oneM2M transactions via the <*transactionMgmt*> and <*transaction*> resources.

When an Originator creates a <*transactionMgmt*> resource, the <*transactionMgmt*> Hosting CSE shall create corresponding child <*transaction*> resources for each oneM2M request primitive specified in the *requestPrimitives* attribute. Each <*transaction*> resource shall be created as a child of the resource targeted by its corresponding request primitive. The *transactionState* attribute shall be used by the <*transactionMgmt*> Hosting CSE to reflect the current state of the transaction. The *transactionControl* attribute shall be used by the <*transactionMgmt*> resource creator or Hosting CSE to manage the atomic execution of the transaction via controlling the state of the transaction. The *transactionLockTime, transactionExecuteTime* and *transactionCommitTime* attributes shall be used by the <*transactionMgmt*> Hosting CSE to coordinate the locking of the resources targeted by the individual oneM2M request primitives, executing the oneM2M request primitives and committing of the transaction. Using these attributes, the <*transactionMgmt*> Hosting CSE shall configure the corresponding *transactionLockTime, transactionExecuteTime and transactionCommitTime* attributes for each corresponding <*transaction*> resource. The *transactionExpirationTime* attribute shall be used by the <*transactionMgmt*> Hosting CSE to manage the expiration of the transaction. If the transaction is not successfully committed before this time, the Hosting CSE shall abort the transaction.

Before creating <*transaction*> resources for each oneM2M request primitive specified in the *requestPrimitives* attribute, the <*transactionMgmt*> Hosting CSE shall first perform a consistency check to validate that the ***From*** request parameter of each oneM2M request primitive is the same as the Originator that created the <*transactionMgmt*> resource. This check ensures that the transaction functionality cannot be used by an AE to impersonate other AEs.

The *transactionMode* attribute shall be used by the <*transactionMgmt*> Hosting CSE to detect whether it is responsible for managing the state of the *transactionControl* attribute and in turn the atomic execution of the transaction on behalf of the creator or whether the creator is to manage the state of the *transactionControl* attribute itself. When *transactionMode* is set to "CREATOR_CONTROLLED" then the creator shall be responsible for controlling the state of *transactionControl.* When *transactionMode* is set to "CSE_CONTROLLED" then the <*transactionMgmt*> Hosting CSE shall be responsible for controlling the state of *transactionControl.*

When the *transactionControl* attribute of the <*transactionMgmt*> resource is updated, the <*transactionMgmt*> Hosting CSE shall make corresponding updates to the *transactionControl* attributes of all the <*transaction*> resources affiliated with the transaction. These updates shall trigger the <*transaction*> Hosting CSE(s) to take corresponding action to lock the targeted resource, execute the specified primitive on the locked resource, commit the executed results or abort the transaction. As a result, the atomic execution of the transaction can be explicitly controlled.

The decision to update the *transactionControl* attribute of the <*transactionMgmt*> resource shall be based on the collective *transactionState* of the individual <*transaction*> resources. When consistent state is detected (E.g. all <*transaction*> resources indicate their *transactionState* is "LOCKED"), then transaction state may be updated (e.g. update <*transactionMgmt*> *transactionControl* to "EXECUTE"). When inconsistent state is detected (e.g. one <*transaction*> resource indicates a *transactionState* of "ERROR"), then the transaction may be retried or aborted (e.g. update the <*transactionMgmt*> *transactionControl* to "ABORT"). To facilitate the gathering of this collective *transactionState*, each <*transaction*> Hosting CSE shall share *transactionState* information with the <*transactionMgmt*> Hosting CSE. This *transactionState* shall be shared in the <*transaction*> create and update responses that a <*transaction*> Hosting CSE returns to the <*transactionMgmt*> Hosting CSE. The <*transactionMgmt*> Hosting CSE shall in turn update the *transactionState* attribute of the <*transactionMgmt*> resource to reflect the collective state of the <*transaction*> resources. The creator of <*transactionMgmt*> resource may subscribe to the *transactionState* attribute of the <*transactionMgmt*> resource to receive notifications for any state changes.

Table 10.2.18.1-1 defines the valid *transactionControl* values permitted in an update request to a <*transactionMgmt*> resource based on the current value of the <*transactionMgmt*> resource's *transactionState*. A <*transactionMgmt*> Hosting CSE shall only allow an update to *transactionControl* for the combinations listed.

Table 10.2.18.1-1: Valid <transactionMgmt> transactionControl update values

Current transactionState	Valid *transactionControl* update values
INITIAL	*LOCK*
LOCKED	*EXECUTE, ABORT*
EXECUTED	*COMMIT, ABORT*
ERROR	*ABORT, INITIAL*
COMMITTED	*INITIAL*
ABORTED	*INITIAL*

Figure 10.2.18.1-1 defines the legal state transitions of the *transactionState* attribute of the <*transactionMgmt*> resource as well as the criteria for each state transition. The value of the *transactionMode* attribute shall determine whether the <*transactionMgmt*> Hosting CSE or the creator of the <*transactionMgmt*> resource shall be responsible for adhering to these valid state transitions.

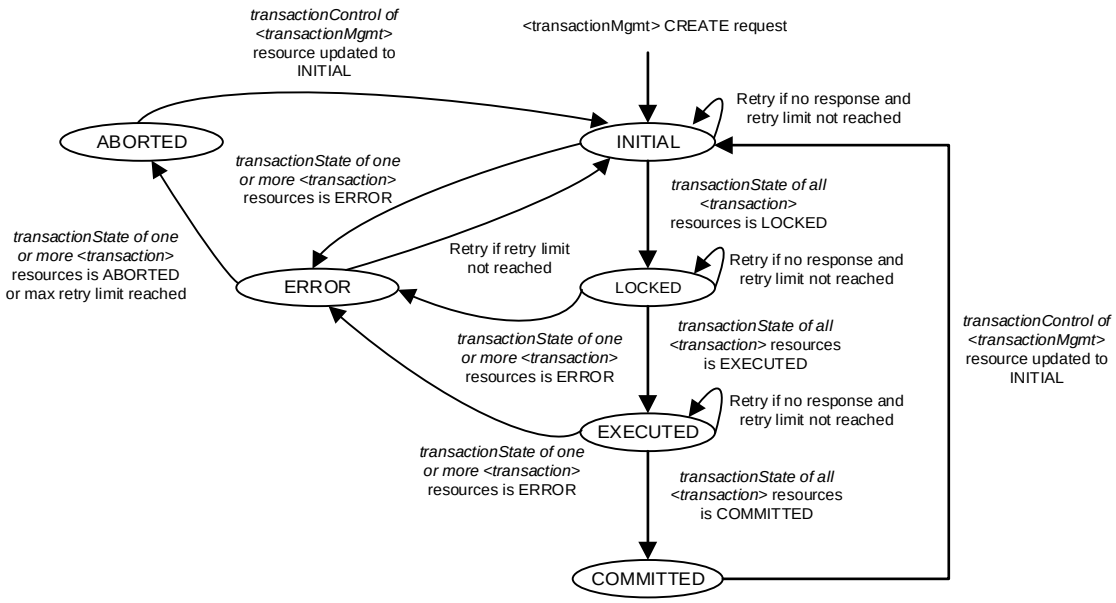


Figure 10.2.18.1-1: *<transactionMgmt> transactionState* State Diagram

If the *transactionMaxRetries* attribute is configured to a non-zero value, the <*transactionMgmt*> Hosting CSE may retry a transaction if it detects a timeout condition or one or more <transaction> resources that have a *transactionState* of "ERROR". To retry a transaction, the <*transactionMgmt*> Hosting CSE may retry sending request(s) that timed out or that resulted in an error. Care should be taken by the <*transactionMgmt*> Hosting CSE when retrying requests. The <*transactionMgmt*> Hosting CSE should evaluate whether a retry makes sense. For example, if a transaction resulted in an error due to the corresponding request primitive not having the proper permissions to access a targeted resource, then the transaction should not be retried since it will have the same results and only waste network bandwidth. An exhaustive list of scenarios for which the Hosting CSE should and should not retry a transaction is not specified in the present document.

If not set or if the max number of retries is exhausted, and the <transactionMgmt> Hosting CSE detects an "ERROR" transactionState from one or more <transaction> Hosting CSEs, then the transactionState of the <transactionMgmt> resource shall be updated to "ABORTED". The <*transactionMgmt*> Hosting CSE supports transaction expiration functionality to detect when the processing of transactions by one or more <*transaction*> Hosting CSEs exceeds the *transactionExpirationTime*. If this occurs, the <*transactionMgmt*> Hosting CSE shall abort the transaction by updating the *transactionState* attribute of each <*transaction*> resource to "ABORT".

When a CSE receives a request to create a new <*transaction*> resource, the CSE shall first check if the targeted resource is already locked. To perform this check, the CSE shall check whether any other <*transaction*> resources exist for the targeted resource. If yes, the CSE shall check the *transactionState* and *transactionLockType* of these other <*transaction*> resource(s). If no other <*transaction*> resources exist or the *transactionState* of the existing <*transaction*> resources is "COMMITTED" or "ABORTED", the CSE shall consider the targeted resource unlocked, otherwise it shall consider it locked. If the targeted resource is locked, the CSE may return an error to the Originator of the <*transaction*> create request. Alternatively, the CSE may buffer requests that target the resource until the resource is unlocked. Note, the details of how a CSE manages this buffering are currently outside the scope of the present document. If a CSE does support buffering for locked resources, then the CSE should also support a timeout mechanism to detect the case where a lock is not released in a timely manner and returning an error to the Originator.

Once the <*transaction*> Hosting CSE obtains a lock for the targeted resource or determines that a lock cannot be obtained, it responds to the Originator of the <*transaction*> create request. In this response, the *transactionState* is included. If the lock is obtained, then the *transactionState* shall have a value of "LOCKED". If the lock cannot be obtained, then the *transactionState* shall have a value of "ERROR".

When a <*transaction*> Hosting CSE receives a request to update the *transactionControl* attribute of an existing <*transaction*> resource, the CSE shall first compare the current value of the *transactionState* and *transactionControl* attributes to ensure they are consistent (e.g. if *transactionControl* has a value of "LOCK" then *transactionState* should have a value of "LOCKED"). This check ensures that the <*transaction*> Hosting CSE is not already in the process of changing the state of the transaction. In addition, the <*transaction*> Hosting CSE shall also compare the value of *transactionControl* specified in the <*transaction*> update request with the current value of *transactionControl* in the <*transaction*> resource to ensure the value specified in the update request is a legal next state (e.g. if transactionState has a value of "LOCKED" then valid transactionControl values are "EXECUTE" or "ABORT"). If any of these checks fail, the <*transaction*> Hosting CSE shall return an error. If these checks pass, the <*transaction*> Hosting CSE shall perform the update to *transactionControl* and in turn perform the corresponding actions associated with transitioning the transaction into this new state.

Table 10.2.18.1-2: Valid <transaction> transactionControl values

Current *transactionState*	Valid *transactionControl* update values
LOCKED	*EXECUTE, ABORT*
EXECUTED	*COMMIT, ABORT*
ERROR	*ABORT, LOCK*
COMMITTED	*LOCK*
ABORTED	*LOCK*

Figure 10.2.18.1-2 defines the legal state transitions of the *transactionState* attribute of the <*transaction*> resource as well as the criteria for each state transition. The creator of the <*transaction*> resource is responsible for adhering to these valid state transitions.

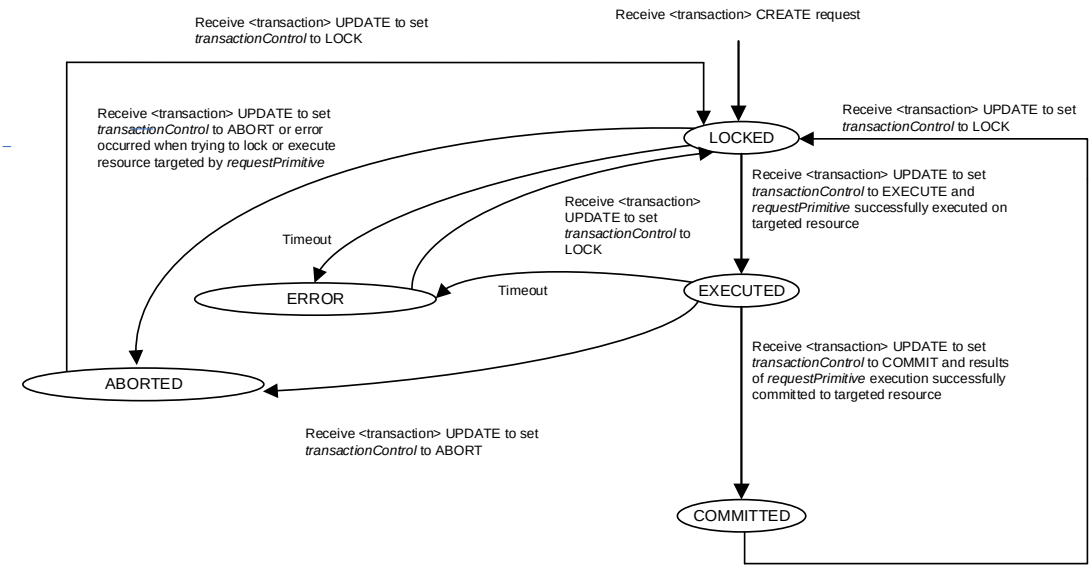


Figure 10.2.18.1-2: <*transaction*> *transactionState* State Diagram

If *transactionControl* is updated to "EXECUTE", then the <*transaction*> Hosting CSE shall perform the request specified in the *requestPrimitive* attribute and update the *responsePrimitive* attribute with the results. If the execution of the primitive is successful the <*transaction*> Hosting CSE shall update the *transactionState* attribute to "EXECUTED". Otherwise, if the execution of the primitive is not successful the <*transaction*> Hosting CSE shall update the *transactionState* attribute to "ERROR".

If *transactionControl* is updated to "COMMIT", then the <*transaction*> Hosting CSE shall make the results of the *requestPrimitive* persistent and visible by updating the *transactionState* attribute to "COMMITTED" which effectively releases the lock on the targeted resource.

If *transactionControl* is updated to "ABORT", the <*transaction*> Hosting CSE shall roll back the transaction by returning the state of the targeted resource to the same state the resource was in before the processing of this transaction began. It shall also update the *transactionState* attribute to "ABORTED" which effectively releases the lock on the targeted resource.

Once the <*transaction*> Hosting CSE completes processing any actions associated with updating the *transactionControl* attribute and changing the state of transaction, it shall return a response back to the Originator of the <*transaction*> update request. In this response, the attributes of the <*transaction*> resource shall be included to allow the Originator to check the *transactionState* and aggregate the *responsePrimitive*.

If a <*transaction*> delete request is received, the <*transaction*> Hosting CSE shall check the transactionState. If the <*transaction*> resource has a transactionState of "INITIAL", "LOCKED", "EXECUTED" or "ERROR", the <*transaction*> Hosting CSE shall roll back the transaction by returning the state of the targeted resource to the same state the resource was in before the processing of this transaction began. It shall also set the *transactionState* attribute to "ABORTED" in the <*transaction*> delete response.

If a <*transaction*> resource is deleted by a delete request that targets the parent or ancestor of the <*transaction*> resource, the <*transaction*> Hosting CSE shall check the *transactionState*. If the <*transaction*> resource has a *transactionState* of "INITIAL", "LOCKED", "EXECUTED" or "ERROR", the <*transaction*> Hosting CSE shall roll back the transaction by returning the state of the targeted resource to the same state the resource was in before the processing of the transaction began.

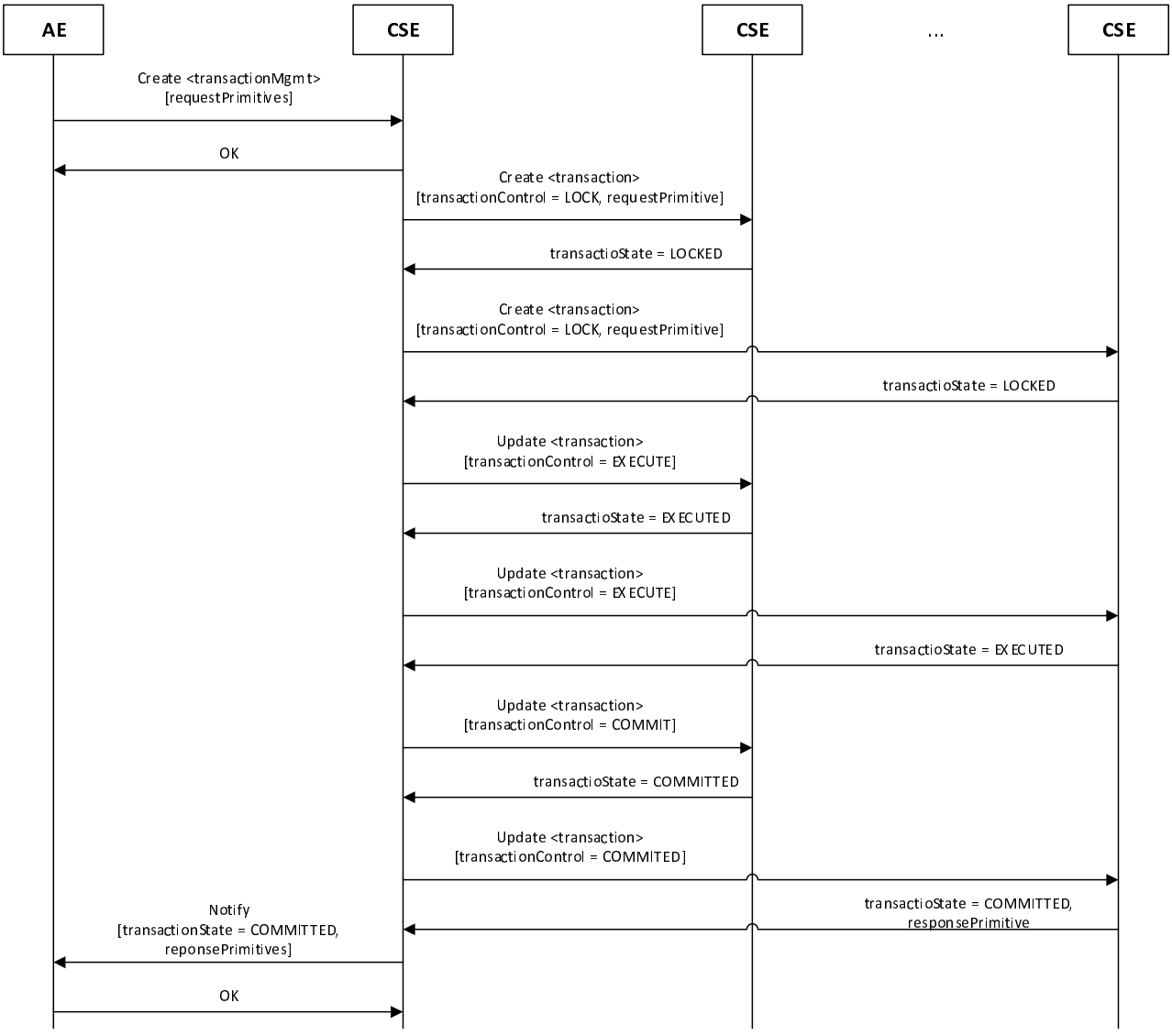


Figure 10.2.18.1-3: Transaction Management using <transactionMgmt> - Successful Case

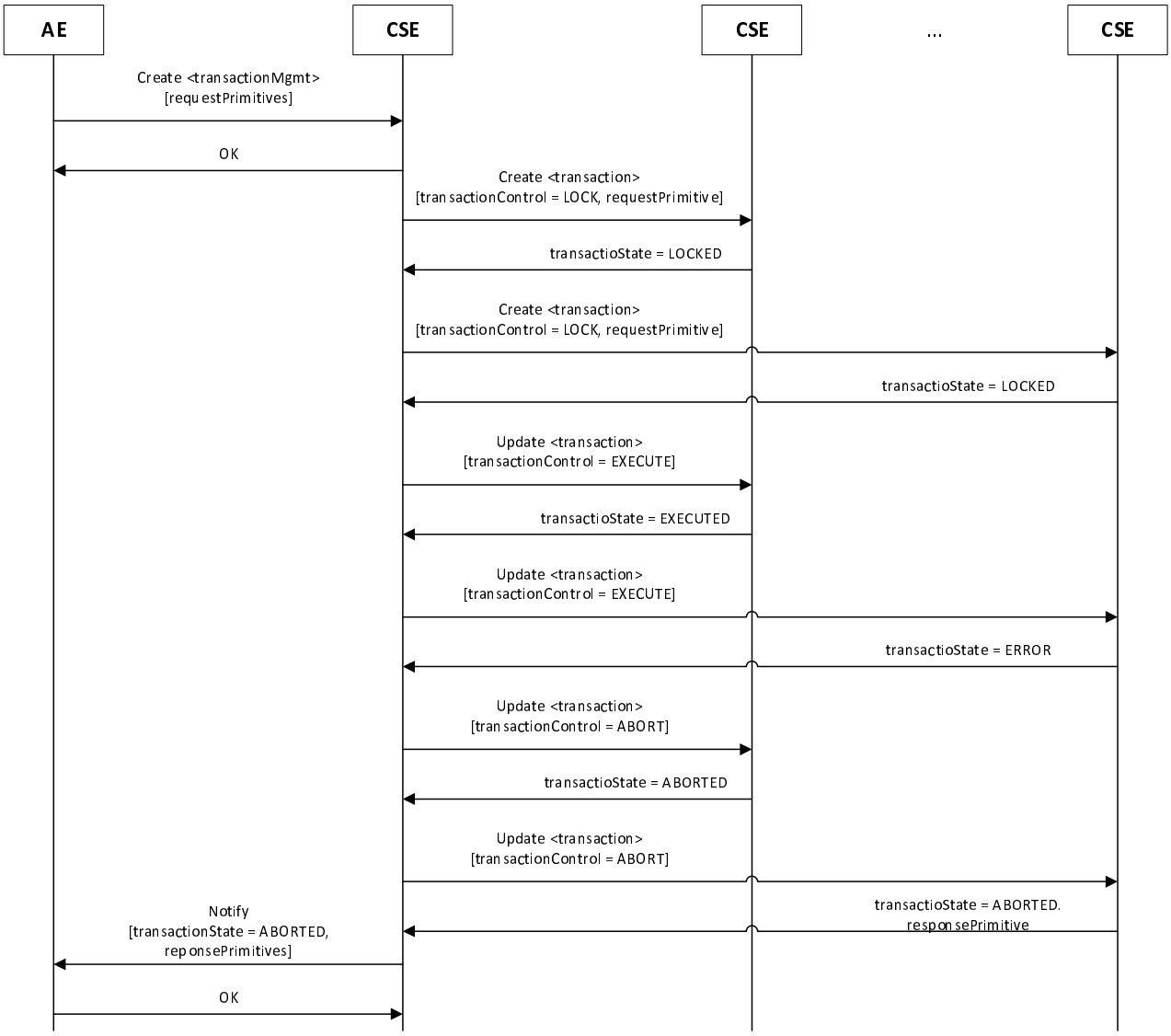


Figure 10.2.18.1-4: Transaction Management using <transactionMgmt> - Failure Case

The <*transaction*> resource may be used independent of the <*transactionMgmt*> resource. In this case, an AE shall be responsible for creating multiple <*transaction*> resources on the corresponding Hosting CSE(s) and controlling the state of each of these transactions. To coordinate the lock, execute and commit times of the individual <*transaction*> resources, an AE may configure the *transactionLockTime*, *transactionExecuteTime* and *transactionCommitTime* attributes to configure the <*transaction*> Hosting CSE to manage the *transactionControl* attribute itself. Alternatively, an AE may perform UPDATEs to the *transactionControl* attribute itself to manage the state of the transaction. Either method may be used by an AE to coordinate the locking, execution and committal of the distributed transaction. Figure 10.2.18.1-5 shows the procedure of an AE managing a transaction.

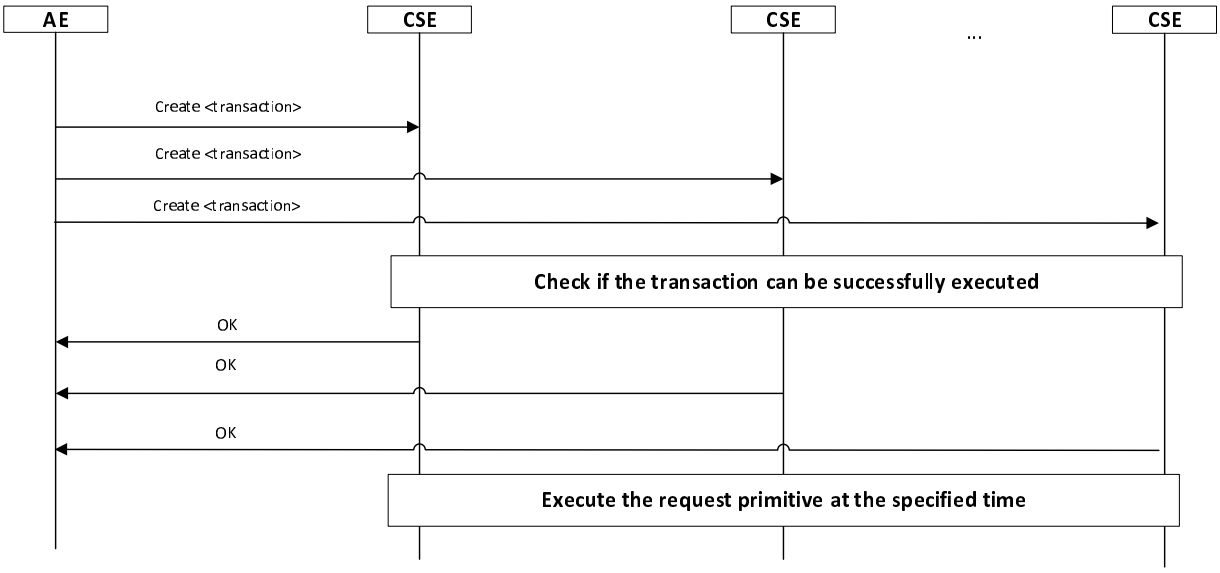


Figure 10.2.18.1-5: Transaction Management by an AE - Successful Case

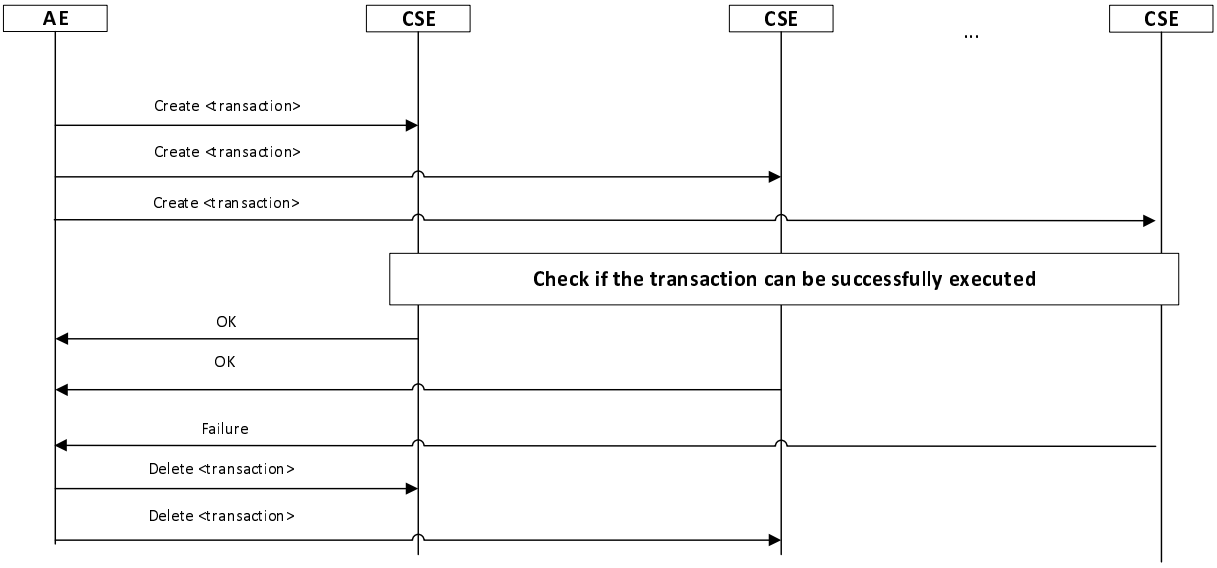


Figure 10.2.18.1-6: Transaction Management by an AE - Failure Case

10.2.18.2 Create <transactionMgmt>

This procedure shall be used for creating a <*transactionMgmt*> resource.

Table 10.2.18.2-1: *<transactionMgmt>* CREATE

***<transactionMgmt>* CREATE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content**:* The resource content shall provide the information as defined in clause 9.6.46.
Processing at Originator before sending Request	The Originator shall request to Create a *<transactionMgmt>* resource by using the CREATE operation. The request shall address a *<CSEBase>, <remoteCSE> or <AE>* resource of a Hosting CSE. The Originator shall be an AE. The Originator shall configure the *requestPrimitives* attribute with a list of requests. The Originator may also configure other optional attributes defined in clause 9.6.47.
Processing at Receiver	For the CREATE procedure, the Receiver shall: • Check if the Originator has CREATE permissions on the target resource • Check the validity of the provided attributes • Upon successful validation of the provided attributes, create a new *<transactionMgmt>* resource. • Process the request as described in clause 10.2.18.1. • Respond to the Originator with the appropriate generic Response with the representation of the *<transactionMgmt>* resource, and the address of the created *<transactionMgmt>* resource if the CREATE was successful.
Information in Response message	The representation of the *<transactionMgmt>* resource.
Processing at Originator after receiving Response	If the *transactionMode* attribute is set to CREATOR_CONTROLLED, then the Originator shall perform subsequent update(s) to the *transactionControl* attribute to manage the transaction.
Exceptions	No change from the basic procedure in clause 10.1.2.

10.2.18.3 Retrieve *<transactionMgmt>*

This procedure shall be used for retrieving *<transactionMgmt>* resource.

Table 10.2.18.3-1: *<transactionMgmt>* RETRIEVE

***<transactionMgmt>* RETRIEVE**	
Information in Request message	***From:*** Identifier of the AE that initiates the Request ***To:*** The address of the *<transactionMgmt>* resource
Processing at Originator before sending Request	The Originator shall request to obtain *<transactionMgmt>* resource information by using the RETRIEVE operation. The request shall address the specific *<transactionMgmt>* resource of a Hosting CSE. The Originator shall be an AE
Processing at Receiver	No change from the basic procedure in clause 10.1.3
Information in Response message	No change from the basic procedure in clause 10.1.3
Processing at Originator after receiving Response	None
Exceptions	No change from the basic procedure in clause 10.1.3

10.2.18.4 Update <*transactionMgmt*>

This procedure shall be used for updating an existing <*transactionMgmt*> resource.

Table 10.2.18.4-1: <*transactionMgmt*> UPDATE

<*transactionMgmt*> UPDATE	
Information in Request message	***From:*** Identifier of the AE that initiates the Request ***To:*** The address of the <*transactionMgmt*> resource
Processing at Originator before sending Request	The Originator shall request to update attributes of an existing <*transactionMgmt*> resource by using an UPDATE operation. The Request shall address the specific <*transactionMgmt*> resource of a CSE. The Originator shall be an AE
Processing at Receiver	The UPDATE procedure shall be: • Check if the Originator has UPDATE permissions on the <*transactionMgmt*> resource. • Check the validity of provided attributes • Upon successful validation of the provided attributes, update the <*transactionMgmt*> resource on the Hosting CSE • Process the request as described in clause 10.2.18.1. • Respond to the Originator with the appropriate generic response with the representation of the <*transactionMgmt*> resource if the UPDATE is successful
Information in Response message	The representation of the <*transactionMgmt*> resource
Processing at Originator after receiving Response	None
Exceptions	No change from the basic procedure in clause 10.1.4

10.2.18.5 Delete <*transactionMgmt*>

This procedure shall be used for deleting an existing <*transactionMgmt*> resource.

Table 10.2.18.5-1: <*transactionMgmt*> DELETE

<*transactionMgmt*> DELETE	
Information in Request message	***From:*** Identifier of the AE that initiates the Request ***To:*** The address of the <*transactionMgmt*> resource
Processing at Originator before sending Request	The Originator shall request to delete an existing <*transactionMgmt*> resource by using the DELETE operation. The request shall address the specific <*transactionMgmt*> resource of a Hosting CSE. The Originator shall be an AE
Processing at Receiver	Besides the basic procedure in clause 10.1.5, the receiver shall process the request as described in clause 10.2.18.1
Information in Response message	No change from the basic procedure in clause 10.1.5
Processing at Originator after receiving Response	None
Exceptions	No change from the basic procedure in clause 10.1.5

10.2.18.6 Create <transaction>

This procedure shall be used for creating a <*transaction*> resource.

Table 10.2.18.6-1: *<transaction>* CREATE

***<transaction>* CREATE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content**:* The resource content shall provide the information as defined in clause 9.6.48.
Processing at Originator before sending Request	The Originator shall request to Create a *<transaction>* resource by using the CREATE operation. The Originator shall be a CSE hosting a *<transactionMgmt>* resource or an AE.
Processing at Receiver	For the CREATE procedure, the Receiver shall: • Check if the Originator has CREATE permissions on the target resource • Check the validity of the provided attributes • Upon successful validation of the provided attributes, create a new *<transaction>* resource. • Process the request as described in clause 10.2.18.1. • Respond to the Originator with the appropriate generic Response with the representation of the *<transaction>* resource, and the address of the created *<transaction>* resource if the CREATE was successful.
Information in Response message	The representation of the *<transaction>* resource.
Processing at Originator after receiving Response	See clause 10.2.18.1.
Exceptions	No change from the basic procedure in clause 10.1.2.

10.2.18.7 Retrieve *<transaction>*

This procedure shall be used for retrieving *<transaction>* resource.

Table 10.2.18.7-1: *<transaction>* RETRIEVE

***<transaction>* RETRIEVE**	
Information in Request message	***From:*** Identifier of the CSE that initiates the Request ***To:*** The address of the *<transaction>* resource
Processing at Originator before sending Request	The Originator shall request to obtain *<transaction>* resource information by using the RETRIEVE operation. The request shall address the specific *<transaction>* resource of a Hosting CSE. The Originator shall be a CSE hosting a *<transactionMgmt>* resource or an AE.
Processing at Receiver	No change from the basic procedure in clause 10.1.3
Information in Response message	No change from the basic procedure in clause 10.1.3
Processing at Originator after receiving Response	None
Exceptions	No change from the basic procedure in clause 10.1.3

10.2.18.8 Update <*transaction*>

This procedure shall be used for updating an existing <*transaction*> resource.

Table 10.2.18.8-1: <*transaction*> UPDATE

<*transaction*> UPDATE	
Information in Request message	***From:*** Identifier of the CSE that initiates the Request ***To:*** The address of the <*transaction*> resource
Processing at Originator before sending Request	The Originator shall request to update attributes of an existing <*transaction*> resource by using an UPDATE operation. The Request shall address the specific <*transaction*> resource of a CSE. The Originator shall be a CSE hosting a <*transactionMgmt*> resource or an AE.
Processing at Receiver	The UPDATE procedure shall be: • Check if the Originator has UPDATE permissions on the <*transaction*> resource. • Check the validity of provided attributes • Upon successful validation of the provided attributes, update the <*transaction*> resource on the Hosting CSE • Process the request as described in clause 10.2.18.1 • Respond to the Originator with the appropriate generic response with the representation of the <*transaction*> resource if the UPDATE is successful
Information in Response message	The representation of the <*transaction*> resource
Processing at Originator after receiving Response	None
Exceptions	No change from the basic procedure in clause 10.1.4

10.2.18.9 Delete <*transaction*>

This procedure shall be used for deleting an existing <*transaction*> resource.

Table 10.2.18.9-1: <*transaction*> DELETE

<*transaction*> DELETE	
Information in Request message	***From:*** Identifier of the CSE that initiates the Request ***To:*** The address of the <*transaction*> resource
Processing at Originator before sending Request	The Originator shall request to delete an existing <*transaction*> resource by using the DELETE operation. The request shall address the specific <*transaction*> resource of a Hosting CSE. The Originator shall be a CSE hosting a <*transactionMgmt*> resource or an AE
Processing at Receiver	Besides the basic procedure in clause 10.1.5, the receiver shall process the request as described in clause 10.2.18.1
Information in Response message	No change from the basic procedure in clause 10.1.5
Processing at Originator after receiving Response	None
Exceptions	No change from the basic procedure in clause 10.1.5

10.2.19 Multimedia session management

10.2.19.1 Create <*multimediaSession*>

This procedure shall be used for creating an <*multimediaSession*> resource.

Table 10.2.19.1-1: *<multimediaSession>* CREATE

***<multimediaSession>* CREATE**	
Information in Request message	All parameters defined in table 8.1.2-2 apply with the specific details for: ***To:*** Address of *<AE>. Only an originating AE shall create an multimediaSession* resource over Mca.
Processing at Originator before sending Request	According to clause 10.1.2
Processing at Receiver	According to clause 10.1.2 with the following complement: The hosting CSE shall check if the *<AE>* resource has a *sessionCapabilities* attribute that is set, and if not the request shall be rejected.
Information in Response message	According to clause 10.1.2
Processing at Originator after receiving Response	The Originator shall create the *<subscription>* resource as the child of created *<multimediaSession>* resource to get notified of session acceptance, status.
Exceptions	According to clause 10.1.2

10.2.19.2 Retrieve *<multimediaSession>*

This procedure shall be used for retrieving the attributes of an *<multimediaSession>* resource.

Table 10.2.19.2-1: *<multimediaSession>* RETRIEVE

***<multimediaSession>* RETRIEVE**	
Information in Request message	All parameters defined in table 8.1.2-2 apply
Processing at Originator before sending Request	According to clause 10.1.3
Processing at Receiver	According to clause 10.1.3
Information in Response message	All parameters defined in table 8.1.3-1 apply
Processing at Originator after receiving Response	According to clause 10.1.3
Exceptions	According to clause 10.1.3

10.2.19.3 Update *<multimediaSession>*

This procedure shall be used for updating attributes of an *<multimediaSession>* resource.

Table 10.2.19.3-1: *<multimediaSession>* UPDATE

***<multimediaSession>* UPDATE**	
Information in Request message	All parameters defined in table 8.1.2-2 apply.
Processing at Originator before sending Request	According to clause 10.1.4
Processing at Receiver	The Hosting CSE shall first check if the UPDATE is modifying the *sessionState*, *offeredSessionDescriptions* or *acceptedSessionDescription* attributes. If the request is not, then it will be processed according to clause 10.1.4. Otherwise, the Hosting CSE shall perform special handling of the UPDATE according to the following procedure. The Hosting CSE shall check if the *sessionState* attribute has a value of ONLINE. If ONLINE, and the UPDATE is not modifying *sessionState* to OFFLINE but the UPDATE is modifying *offered SessionDescriptions* or *acceptedSessionDescription* then the Hosting CSE shall reject the request and return and error to the originator. Otherwise the Hosting CSE shall perform the UPDATE.
Information in Response message	According to clause 10.1.4
Processing at Originator after receiving Response	According to clause 10.1.4
Exceptions	According to clause 10.1.4

10.2.19.4 Delete <*multimediaSession*>

This procedure shall be used for deleting an <*multimediaSession*> resource.

Table 10.2.19.4-1: *<multimediaSession>* DELETE

***<multimediaSession>* DELETE**	
Information in Request message	All parameters defined in table 8.1.2-2 apply
Processing at Originator before sending Request	According to clause 10.1.5
Processing at Receiver	According to clause 10.1.5
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	According to clause 10.1.5
Exceptions	According to clause 10.1.5

10.2.20 Background Data Transfer Management

10.2.20.1 Introduction

This clause describes the procedures for creation, retrieval, update and deletion of the <*backgroundDataTransfer*> resource. The corresponding procedures over the Mcn reference point are described in ETSI TS 118 126 [15].

10.2.20.2 Create *<backgroundDataTransfer>*

This procedure shall be used for creating an *<backgroundDataTransfer>* resource

Table 10.2.20.2-1: *<backgroundDataTransfer>* CREATE

***<backgroundDataTransfer>* CREATE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content:*** The resource content shall provide the information as defined in clause 9.6.9
Processing at Originator before sending Request	According to clause 10.1.2
Processing at Receiver	According to clause 10.1.2
Information in Response message	According to clause 10.1.2
Processing at Originator after receiving Response	According to clause 10.1.2
Exceptions	According to clause 10.1.2

10.2.20.3 Retrieve <backgroundDataTransfer>

This procedure shall be used for retrieving the representation of the *<backgroundDataTransfer>* resource.

Table 10.2.20.3-1: *<backgroundDataTransfer>* RETRIEVE

***<backgroundDataTransfer>* RETRIEVE**	
Information in Request message	All parameters defined in table 8.1.2-3
Processing at Originator before sending Request	According to clause 10.1.3
Processing at Receiver	According to clause 10.1.3
Information in Response message	All parameters defined in table 8.1.3-1 apply with the specific details for: ***Content***: attributes of the *<backgroundDataTransfer>* resource as defined in clause 9.6.9
Processing at Originator after receiving Response	According to clause 10.1.3
Exceptions	According to clause 10.1.3

10.2.20.4 Update <backgroundDataTransfer>

This procedure shall be used for updating the attributes and the actual data of an *<backgroundDataTransfer>* resource.

Table 10.2.20.4-1: *<backgroundDataTransfer>* UPDATE

***<backgroundDataTransfer>* UPDATE**	
Information in Request message	All parameters defined in table 8.1.2-3 apply with the specific details for: ***Content*:** attributes of the *<backgroundDataTransfer>* resource as defined in clause 9.6.9 which need be updated
Processing at Originator before sending Request	According to clause 10.1.4
Processing at Receiver	According to clause 10.1.4
Information in Response message	According to clause 10.1.4
Processing at Originator after receiving Response	According to clause 10.1.4
Exceptions	According to clause 10.1.4

10.2.20.5 Delete <*backgroundDataTransfer*>

This procedure shall be used for deleting the <*backgroundDataTransfer*> resource with all related information.

Table 10.2.20.5-1: <*backgroundDataTransfer*> DELETE

<*backgroundDataTransfer*> DELETE	
Information in Request message	All parameters defined in table 8.1.2-3 apply
Processing at Originator before sending Request	According to clause 10.1.5
Processing at Receiver	According to clause 10.1.5
Information in Response message	According to clause 10.1.5
Processing at Originator after receiving Response	According to clause 10.1.5
Exceptions	According to clause 10.1.5

11 Trust Enabling Architecture

11.0 Overview

The Trust Enabling Architecture serves the purpose of establishing security and trust between all parties involved in the M2M ecosystem. It comprises the following infrastructure functions which may be external to the CSEs:

- M2M Enrolment functions(MEF), which manage the enrolment and configuration of M2M Nodes and M2M applications for access to M2M Services provided by an M2M Service Provider, prior to service operation (see clause 11.2). The credentials provisioned by a MEF can be used for Security Association Establishment Framework, End-to-End Security of Primitives or End-to-End Security of Data.

- M2M Authentication functions(MAF), which may facilitate identification and authentication of CSEs and AEs (see clause 11.3), End-to-End Security of Primitives (see clause 11.4.2) and End-to-End Security of Data (see clause 11.4.1) during M2M service operation. A single MAF may support all of the above security services or only a selection of them.

- Dynamic Authorization Systems and Role Authorities, which manage authorization privileges to access resources that may be assigned during operation (see clauses 11.3.4 and 11.5).

- Distributed Authorization Systems allow for four authorization functional components (i.e. PEP, PDP, PRP and PIP) to be distributed in different CSEs (see clauses 11.3.4 and 11.6).

- Privacy Policy Managers that assist in the management of privacy preferences expressed by data subject with respect to service requirements and applicable regulations.

The above functionalities are assumed to be operated by trusted parties (generally M2M Service Providers but possibly other trusted third parties called M2M Trust Enablers (MTE)). These functions are all detailed in ETSI TS 118 103 [2].

A MAF interacts with a Security Principal via the reference point Mmaf. A MEF interacts with a Security Principal via the reference point Mmef. The Mmef reference point optionally supports triggering Device Configuration (defined in ETSI TS 118 122 [10]) and Certificate Provisioning (specified in ETSI TS 118 103 [2]). The communication protocols used on the reference points Mmaf and Mmef are specified in ETSI TS 118 132 [13].

11.1 Enrolling M2M Nodes and M2M Applications for oneM2M Services

Though M2M Nodes in the field domain are assumed to communicate without human involvement, individuals or organizations remain responsible for setting the access control policies used to authorize their M2M Nodes to access M2M services. In the following text, M2M Nodes refers to M2M field nodes.

In particular, individuals or organizations acquiring M2M Nodes can subscribe to a contract with an M2M Service provider (M2M Service Subscription) under which they enrol their M2M Nodes (e.g. using identifiers pre-provisioned on the nodes, such as Node-ID). This in turn may require an M2M Service provisioning step (including Security provisioning) that takes place on the target M2M Nodes themselves, for which interoperable procedures are specified by oneM2M (see clause 11.2.1). Following M2M service provisioning, the nodes can be identified and authenticated for association with an M2M Service Subscription, whose properties reflect the contractual agreement established between their owner and the M2M Service Provider.

Similarly, it may be possible for an M2M Service Provider to mandate that an M2M Application accessing M2M services be associated with a security credentials used to authorize specific operations to instance of that M2M Application, i.e. AEs (see clause 11.2.2). This step facilitates the deployment and management of M2M Applications that are instantiated in great numbers, as it enables all instances of an M2M Application to be managed through common security policies that are set once for all. It also enables keeping control over M2M Applications issued by untrusted sources.

The above steps may be delegated to an M2M trust enabler, when this role is not assumed by the M2M Service Provider.

11.2 M2M Initial Provisioning Procedures

11.2.1 M2M Node Enrolment and Service Provisioning

M2M service provisioning is the process by which M2M Nodes are loaded with the specific information needed to seamlessly access the M2M Services offered by an M2M Service Provider. This is an initial step performed only when an M2M Node is enrolled for using the M2M services of an M2M Service Provider. Though this process can be performed during device manufacturing, there is a need to enable this process to take place during field deployment in an interoperable way. M2M service provisioning assumes the existence of an M2M service subscription contracted with the target M2M Service Provider for the target M2M Node. Remote provisioning scenarios require the M2M Node to be mutually authenticated using pre-existing credentials (e.g. Node-ID and associated credential) with an M2M enrolment function, to securely exchange the provisioning information with the contracted M2M Service Provider. The M2M Service Provisioning takes place between an M2M Node (without provisioned CSE) and an M2M Service Provider via an M2M enrolment function. As a result of provisioning, M2M Nodes are provided with necessary credentials and possibly other M2M service related parameters (e.g. CSE-ID, M2M-Sub-ID).

The first step of M2M service provisioning is the security provisioning procedure, by which M2M service provider specific credentials are either shared between two M2M Nodes, or shared between the M2M Node in the field domain and an M2M authentication function in the infrastructure. Authenticated M2M Nodes can then be associated with an M2M Service Subscription used to determine their specific authorizations.

The following security provisioning scenarios are supported by the oneM2M architecture:

- Pre-provisioning:
  - Pre-provisioning includes all forms of out-of-band provisioning, e.g. provisioning M2M Nodes with M2M subscription information during the manufacturing stage.
- Remote provisioning:
  - Remote provisioning relies on pre-existing credentials in M2M Nodes (e.g. digital certificates or network access credentials) to provision subscription related parameters through a secure session with an M2M Enrolment Function. This form of provisioning enables M2M Nodes already in the field (e.g. operational M2M Nodes) to be provisioned with M2M Service subscription.
  - When supported, remote provisioning procedure shall be implemented as described in the ETSI TS 118 103 [2].
  - Following M2M service provisioning, the provisioned entity securely stores credentials used for authentication , with an associated lifetime (e.g. corresponding to the duration of the contractual agreement embodied by the M2M service subscription).

11.2.2 M2M Application Enrolment

This procedure is an optional step that enables the M2M SP and/or M2M Application provider to control which M2M Applications are allowed to use the M2M services. It assumes that the M2M Application is associated of a credential used for controlling authorization to M2M services The security credential associated with the App-ID or AE-ID may be used to grant specific authorization to M2M Application instances to access an approved list of M2M services, or revoke access to all instances of undesirable M2M Applications. Such authorization shall take place between registrar CSE and AE as specified in the present document and the ETSI TS 118 103 [2].

11.3 M2M Operational Security Procedures

11.3.0 Overview

This clause introduces the high level procedures, following M2M Enrolment, that shall be performed before any other procedure on Mcc and Mca can take place. These procedures shall be implemented as specified in the ETSI TS 118 103 [2].

NOTE: The detailed specifications of the security procedures in ETSI TS 118 103 [2] uses different labels for the steps shown in figures 11.3.0-1 and 11.3.0-2:

- Step 1: Provisioning maps to *Credential Configuration*;
- Step 2: Identification maps to *Association Configuration*;
- Step 3: Authentication maps to *Association Handshake* in ETSI TS 118 103 [2].

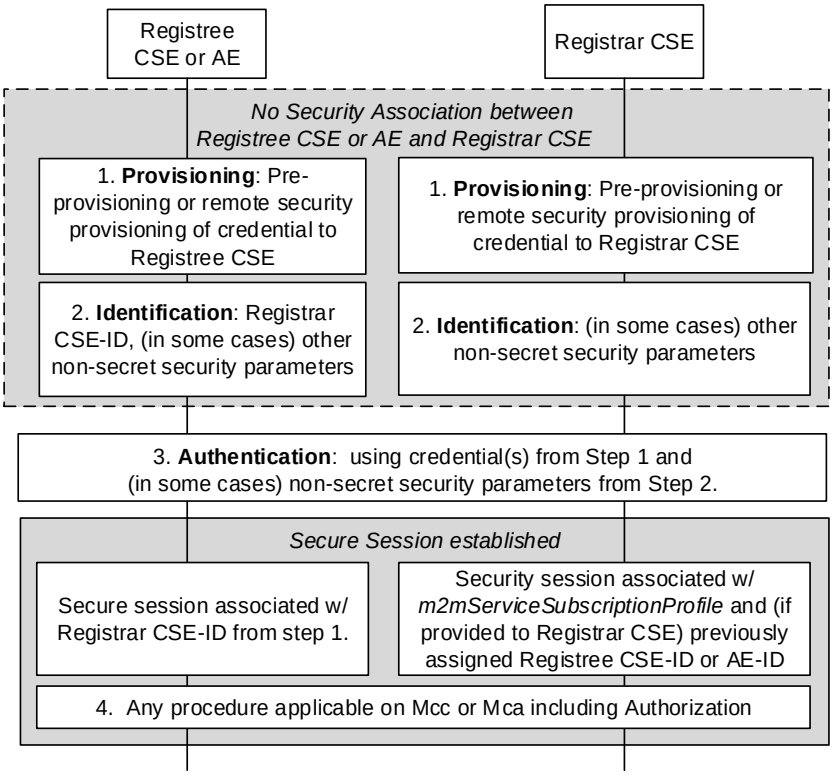


Figure 11.3.0-1: High Level Procedures on Mcc or Mca without MAF

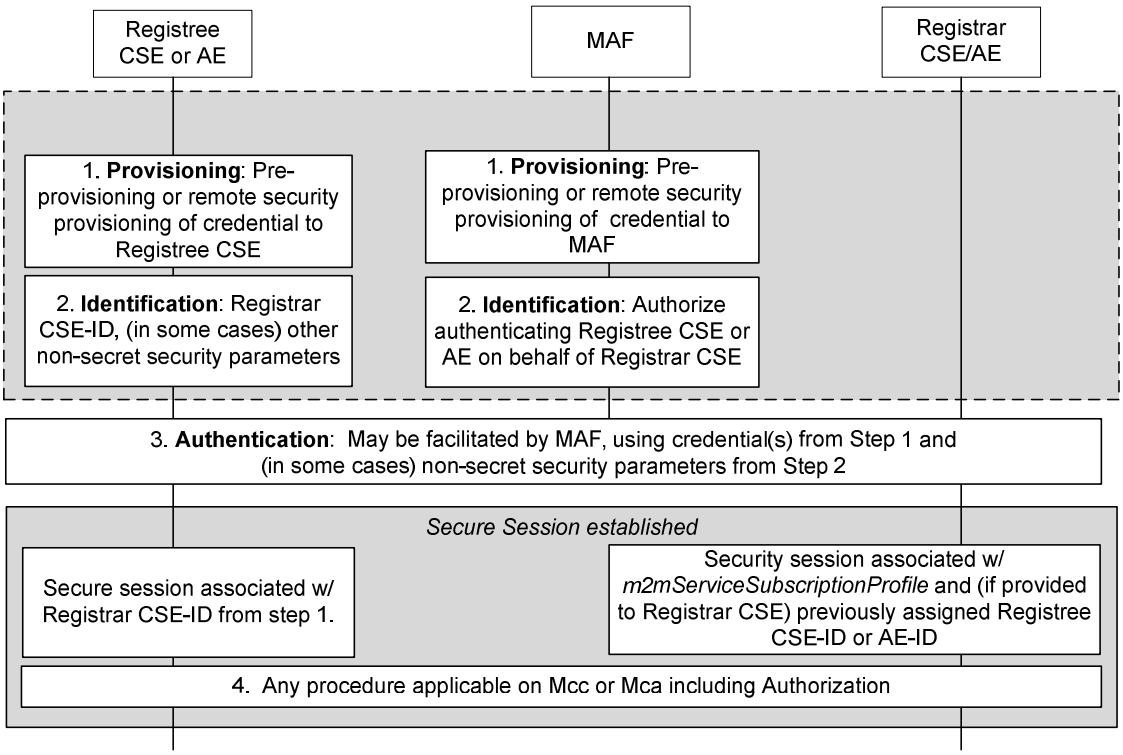


Figure 11.3.0-2: MAF assisted High Level Procedures on Mcc or Mca

11.3.1 Identification of CSE and AE

Once a CSE or AE is provisioned with its security credentials, there is no need to configure long-term secret information to the CSE or AE. However, additional non-secret information may need to be configured using the same security procedures.

Prior to a CSE or AE initiating security association establishment, the Registree CSE or AE is configured with the Registrar CSE-ID so that the Registree knows who to establish the security association with. This process is called "Association Configuration" in ETSI TS 118 103 [2].

11.3.2 Authentication and Security Association of CSE and AE

The association security handshake (see ETSI TS 118 103 [2]) provides:

a) mutual authentication of CSE and AE; and

b) session key derivation.

Prior to granting access to M2M services, the credentials resulting from the M2M Node and M2M application enrolment procedures shall be used, together with the information supplied in the identification step (clause 11.1), to perform mutual authentication of the Registree CSE or AE with the Registrar CSE. Upon mutual authentication:

- Registree CSE or AE associates, with the Registrar CSE, the CSE-ID supplied in the identification step (clause 11.1).
- If the Registree CSE or AE has previously registered successfully with the Registrar CSE and the Registrar CSE has retained the applicable M2M service subscription and CSE-ID or AE-ID, then the Registrar CSE can use this information.
- In other cases, the Registrar CSE determines the applicable M2M service subscription and CSE-ID or AE-ID as described in clause 10.2.2 in the present document.

The Registree receives authorization to access the M2M services defined in the <*m2mServiceSubscription*> resources by checking privileges defined in <*accessControlPolicy*>, <*token*> or <*role*> resources.

NOTE: The authorization procedure to access the M2M services is further described in clause 11.3.4 and specified in detail in clause 7 of ETSI TS 118 103 [2].

Session keys are then derived for providing desired security services to the communicating entities, such as confidentiality and/or integrity of information exchange (these security services may be provided through establishment of a secure channel between the communicating entities or through object based security where only relevant information is encrypted prior to being shared). The lifetime of a security association shall be shorter than the lifetime of the credential used for authentication from which it is derived: It may be valid for the duration of a communication session, or be determined according to the validity period of the protected data. In case of a security association between two AEs, the lifetime of the security association can result from a contractual agreement between the subscribers of the communicating AEs.

11.3.3 Void

11.3.4 M2M Authorization Procedure

The M2M authorization procedure controls access to resources and services by CSEs and AEs. This procedure requires that the Originator has been identified to an M2M Authentication Function and mutually authenticated and associated with an M2M Service Subscription. Authorization depends on:

- The privileges set by the M2M Service Subscription associated with the Originator (e.g. service/role assigned to the Originator).
- These privileges are set-up based on the access control policies associated with the accessed resource or service. They condition the allowed operations (e.g. CREATE) based on the Originator's privileges and other access control attributes (e.g. contextual attributes such as time or geographic location).
- Role-IDs which have been associated with the Originator.

The authorization/access grant involves an Access Decision step to determine what the authenticated CSE or AE can actually access, by evaluating applicable access control policies based on the CSE or AE privileges. Access Decision is described in ETSI TS 118 103 [2].

The following set of access control policy attributes shall be available for an Access Decision:

- Access control attributes of Originator and Originator's Role (e.g. Role-IDs, CSE_IDs, AE-IDs, etc.).
- Access control attributes of Environment/Context (e.g. time, day, IP address, etc.).
- Access control attributes of Operations (e.g. Create, Execute, etc.).

The M2M Service Provider/administrator and owner of resources are responsible to establish access control policies that determine by whom, in what context and what operations may be performed upon those resources. If the request satisfies the owner's access control policy, then the access to the resource is granted.

Dynamic Authorization: Dynamic Authorization encompasses:

a) authorizing the creation of a limited-lifetime access control policy authorizing the Originator to perform specific operations on the requested resource; and

b) issuing limited-lifetime Tokens associating the Originator with Role-IDs and/or access control policies for identified resources.

Two forms of Dynamic Authorization are supported: Direct Dynamic Authorization and Indirect Dynamic Authorization.

In the event that the request does not satisfy any of the owner's access control policies, then Dynamic Authorization may be requested from Dynamic Authorization System (DAS) Servers; this is called *Direct Dynamic Authorization*, and relevant details are provided clause 11.5.2. The request is then re-evaluated to determine if the owner's access control policy is now satisfied and access is granted.

If access is still denied, then the Originator is provided with ***Token Request Information*** used to request the issuance of Tokens by a Dynamic Authorization System. A Token identifies Role-IDs and/or access control policies (for identified resources) which have been temporarily associated with the Originator. The Originator then resends the request from the Originator, this time adding any Token or Token-IDs received from the Dynamic Authorization System. This is called *Indirect Dynamic Authorization*, and relevant details are provided clause 11.5.3.

NOTE: A DAS Server can be triggered, by Dynamic Authorization, to update the access control policy configuration using oneM2M request primitives.

In the event that the requesting entity does not satisfy the owner's access control policy, a Hosting CSE shall check to see if the resource (or one of its parents) has a *dynamicAuthorizationConsultationIDs* which links to a valid <*dynamicAuthorizationConsultation*> resource. If there is no valid <*dynamicAuthorizationConsultation*> resource or if the dynamicAuthorizationEnabled attribute is set to "false", then then the Hosting CSE shall not attempt to perform direct dynamic authorization on behalf of the requesting entity. However, if there is a valid <*dynamicAuthorizationConsultation*> resource available and if the dynamicAuthorizationEnabled attribute is set to "true", then the Hosting CSE shall initiate a direct dynamic authorization request to the specified dynamicAuthorizationPoA. If direct dynamic authorization results in sufficient privileges being granted to the requesting entity, the Hosting CSE shall grant it access. In addition the Hosting CSE may also dynamically create a new access control policy and configure it with the granted privileges along with any specified lifetime associated with the privileges based on a resource creation process initiated by the dynamic authorization system.

This function shall fetch the subscription related information in order to check if a Role-ID used in a request is allowed by the M2M service subscription. The authorization procedure shall be implemented as specified in the ETSI TS 118 103 [2].

Distributed Authorization

A distributed authorization system may comprise four functional components: Policy Enforcement Point (PEP), Policy Decision Point (PDP), Policy Retrieval Point (PRP) and Policy Information Point (PIP). A PEP that coexists with the Hosting CSE enforces the access control decision. PDP, PRP and PIP are responsible for making access control decisions, providing applicable access control policies and obtaining access control information required by access control policy evaluation procedures respectively. In a distributed authorization system these components may be distributed in different CSEs. Details of these components are described in ETSI TS 118 103 [2].

Three resource types are defined for representing PDPs, PRPs and PIPs: <*authorizationDecision*>, <*authorizationPolicy*> and <*authorizationInformation*>. For details about these resource types see clauses 9.6.41, 9.6.42 and 9.6.43.

Three attributes are defined in the <*accessControlPolicy*> resource type for providing the addresses of PDPs, PRPs and PIPs: *authorizationDecisionResourceIDs*, *authorizationPolicyResourceIDs* and *authorizationInformationResourceIDs*. For details about these resource attributes see clause 9.6.2.

A high level description of the distributed authorization framework and procedures is provided in clause 11.6.

11.4 Functional Architecture Specifications for End-to-End Security Procedures

11.4.1 Functional Architecture Specifications for End-to-End Security of Data (ESData)

End-to-End Security for Data (ESData) provides an interoperable framework for protecting data that ends up transported using oneM2M reference points, in order that so transited CSEs do not need to be trusted with that data. The data shall comprise either:

- All or part of the value of a single attribute (e.g. *content* attribute value of a <*contentInstance*> resource or *customAttribute* of a <*flexContainer*> resource) or a single addressable element within the attribute.
- All or part of a single primitive parameter value (e.g. a signed, self-contained access token communicated in a request primitive to obtain dynamic authorization).

11.4.2 Functional Architecture Specifications for End-to-End Security of Primitives (ESPrim)

End-to-End Security for Primitives (ESPrim) provides an interoperable framework for securing oneM2M primitives so CSEs do not need to be trusted with the confidentiality and integrity of the primitive. ESPrim provides mutual authentication, confidentiality, integrity protection and a freshness guarantee (bounding the age of secured primitives).

The credential management aspects and data protection aspects for ESPrim are specified in ETSI TS 118 103 [2]. The present clause specifies the transport of secured primitives.

The primitive to be secured is called the *inner primitive*, and the primitive which is used to transport a secured inner primitive is called the *outer primitive*. The inner primitive is protected using an encryption and integrity protection, which takes a symmetric key *sessionESPrimKey* as input. The *sessionESPrimKey* is derived from a *pairwiseESPrimKey*, established between the Originator and Receiver, and a *receiverESPrimRandObject* and *originatorESPrimRandObject*. The receiverESPrimRandObject and originatorESPrimRandObject are specified in ETSI TS 118 103 [2].

The transport details for the ESPrim Procedure are shown in figures 11.4.2-1 and 11.4.2-2, and described in the following text.

NOTE 1: The outer primitive is not acting on resources because the outer primitive is only used to transport the ESPrim object securing the inner primitive. This is the reason that the NOTIFY procedure is used for the outer primitive.

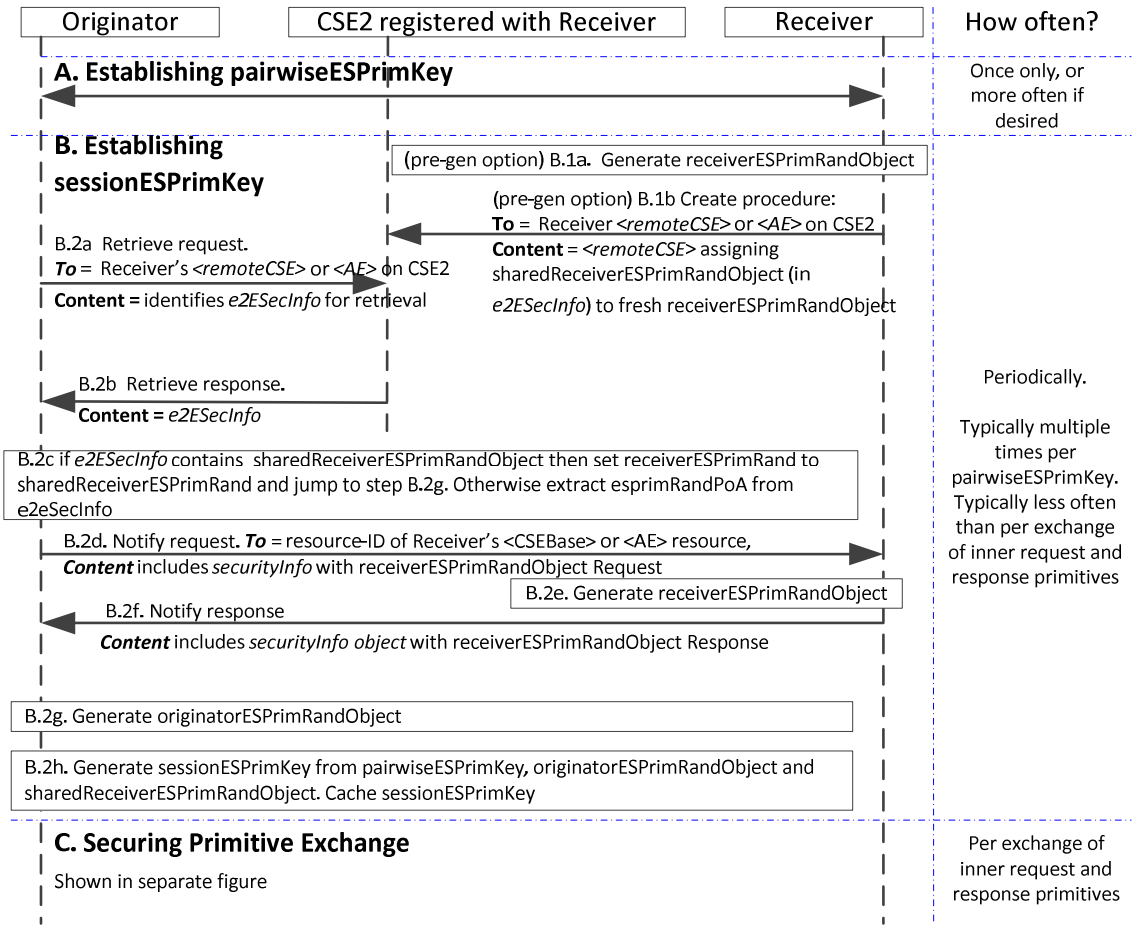


Figure 11.4.2-1: The transport details for establishing pairwiseESPrimKey and establishing sessionESPrimKey in the End-to-End Security of Primitives (ESPrim) Procedure

This message flow shows the sequence of events for Blocking Mode:

A. **Establishing *pairwiseESPrimKey*:** The *pairwiseESPrimKey* shall be established as specified in clause 8.4.2 "End-to-End Security of Primitives (ESPrim) Architecture" in ETSI TS 118 103 [2].

B. **Establishing *sessionESPrimKey*:** The Receiver shall select to either (a) pre-generate a receiverESPrimRandObject which is distributed for used by multiple Originators for establishing sessionESPrimKey, or (b) generate a unique receiverESPrimRand Object upon request (in which case no action is required prior to receiving such a request).

B.1. **(Optional) Receiver pre-generates and distributes *receiverESPrimRandObject*.** If the Receiver selected to pre-generate and distribute a receiverESPrimRandObject, the Receiver performs the following steps every time the Receiver wishes to provide a new shared receiverESPrimRandObject:

B.1a The Receiver shall generate a *recei*verESPrimRandObject as described in ETSI TS 118 103 [2].

B.1b The Receiver shall update the Receiver's <*remoteCSE*> or <*AE*> resource on all CSEs to which the Receiver is registered, with the *sharedReceiverESPrimRand Object* parameter of the *e2eSecInfo* attribute containing the generated *receiverESPrimRandObject*.

In the latter case, the Receiver shall ensure that the *sharedReceiverESPrimRandObject* parameter is not present in the *e2eSecInfo* attribute in the Receiver's <*remoteCSE*> or <*AE*> resource on all CSEs to which the Receiver is registered. The absence of the *sharedReceiverESPrimRand Object* parameter indicates that the Receiver will provide a unique *receiverESPrimRand Object* upon request.

B.2. Originator obtains *receiverESPrimRandObject:*

B.2a The Originator shall perform a Retrieve on the *e2eSecInfo* attribute in the Receiver's <*remoteCSE*> or <*AE*> resource on a CSE, here denoted CSE2, with which the Receiver is registered.

B.2b If the *e2eSecInfo* attribute is present in the Receiver's <*remoteCSE*> or <*AE*> resource on CSE2, then CSE2 shall returns the *e2eSecInfo* attribute. Otherwise CSE2 shall return an appropriate error message.

B.2c (This step is also described in ETSI TS 118 103 [2]. Where there is a conflict, ETSI TS 118 103 [2] is to be treated as the authoritative description). The Originator determines if the Receiver supports ESPrim, which requires that the *e2eSecInfo* attribute is present and the *e2eSecInfo* attribute indicates support for ESPrim:

B.2c.1 If the Receiver does not support ESPrim, then the Originator aborts the procedure.

B.2c.2 If the Receiver supports ESPrim, and the e2eSecInfo attribute includes a sharedReceiverESPrimRandObject parameter, then the Originator shall examine the ESPrimRandExpiry in this parameter to determine if the sharedReceiverESPrimRandObject has expired. If the sharedReceiverESPrimRandObject has not expired, then the Originator sets receiverESPrimRandObject to the value of receiverESPrimRandObject and proceeds to step B.2g. If the sharedReceiverESPrimRandObject has expired, then the Originator sets receiverESPrimRandObject to the value of receiverESPrimRandObject and proceeds to step B.2d.

B.2c.3 If the Receiver supports ESPrim, and the e2eSecInfo attribute does not include a sharedReceiverESPrimRandObject parameter, then the Originator proceeds to step B.2d.

B.2d The Originator shall send a NOTIFY request to the Receiver with the To parameter set to the address of the Receiver's <CSEBase> or <AE> resource, and the *securityInfo Type* element of the *securityInfo* object in the ***Content*** indicating that this NOTIFY request is a "receiverESPrimRandObject request".

NOTE 2: When the Receiver is a CSE, the Originator can use the Receiver's CSE-ID followed by "/." as the address of the Receiver's <*CSEBase*>.

B.2e The Receiver, upon receiving such a NOTIFY request, shall generate a receiverESPrimRandObject as described in ETSI TS 118 103 [2].

B.2f The Receiver shall send a NOTIFY response to the Originator with the *securityInfoType* element of the *securityInfo* object in the ***Content*** indicating that this is a "receiverESPrimRandObject request" and containing the receiverESPrimRandObject.

B.2g The Originator shall generate an originatorESPrimRandObject as described in clause 8.4.2 of ETSI TS 118 103 [2].

B.2h The Originator shall generate the sessionESPrimKey from the pairwiseESPrimKey, originatorESPrimRandTuple and receiverESPrimRandObject as described in clause 8.4.2 of ETSI TS 118 103 [2].

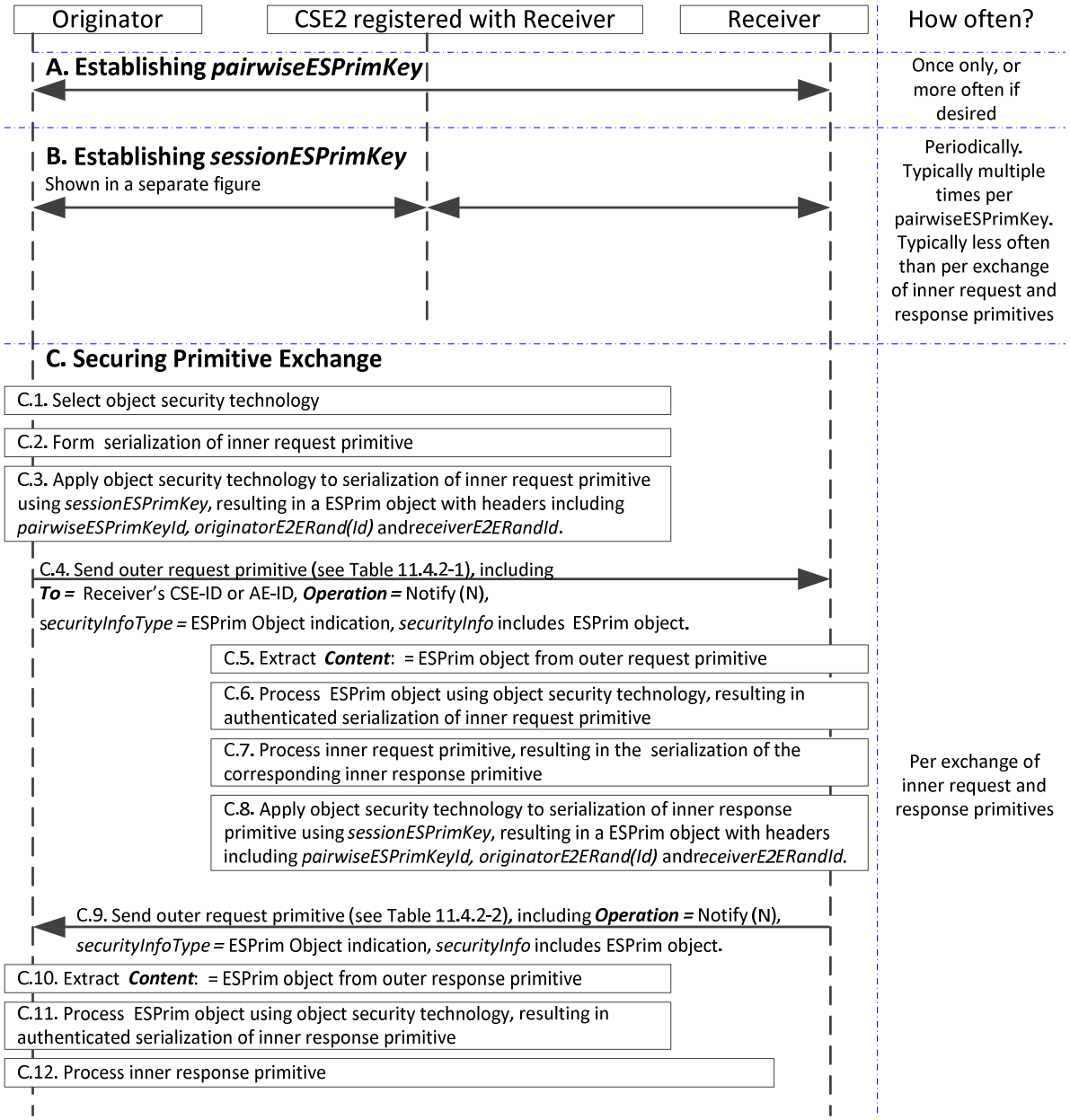


Figure 11.4.2-2: The transport details for Securing a Primitive Exchange in the End-to-End Security of Primitives (ESPrim) Procedure

This message flow shows the sequence of events for Blocking Mode:

C. **Securing a Primitive Exchange**

C.1 The Originator selects the object security technology as described in clause 8.4.2 of ETSI TS 118 103 [2].

C.2 The Originator shall form the serialization of the inner request primitive.

C.3 The Originator shall produce a ESPrim Object from the serialization of the inner request primitive by applying the selected object security technology using the established parameters, as described in clause 8.4.2 of ETSI TS 118 103 [2].

C.4 The Originator shall send the ESPrim Object to the Receiver in the *securityInfo* object in the *Content* of an outer request primitive, and including the indication that *securityInfo* contains an ESPrim Object. The outer request primitive shall be a NOTIFY request primitive with *To* set to the address of the Receiver's <*CSEBase*> or <*AE*> resource. See note 2. The parameters of the outer request primitive shall be assigned as described in table 11.4.2-1.

C.5 The Receiver shall process the outer request primitive as for normal NOTIFY request primitives. The Receiver shall extract *securityInfo*, process the indication that it contains an ESPrim Object, and extract the ESPrim Object containing the secured inner request primitive.

C.6 The Receiver shall process the ESPrim Object according to the indicated object security technology resulting in the verified serialization of the inner request primitive. This processing is described in clause 8.4.2 of ETSI TS 118 103 [2].

C.6a If this processing is unsuccessful, then the Receiver shall generate an error message:

C.6a.1 If the Receiver knows a currently valid *sessionESPrimKey* previously established with the Originator, then the receiver shall secure the error message using ESPrim as described in clause 8.4.2 of ETSI TS 118 103 [2]. In this case the message flow skips to step C.9.

C.6a.2 If Receiver does not know a currently valid *sessionESPrimKey* previously established with the Originator, then the Receiver shall send a NOTIFY response with the (unsecured) error message in the *Content* parameter. The Originator processes the response as for a normal error case.

C.7 The Receiver shall process the inner request primitive, resulting in a serialization of the corresponding inner response primitive.

NOTE 3: Steps C.3 to C.7 are mirrored closely by C.10 to C.16, with the Originator and Receiver swapping their participation in the exchange, and the request primitives replaced by response primitives.

C.8 The Receiver shall produce a ESPrim Object from the serialization of the inner response primitive by applying the selected object security technology using the established parameters, as described in clause 8.4.2 of ETSI TS 118 103 [2].

C.9 The Receiver shall send the ESPrim Object to the Originator in the *securityInfo* object of an outer response primitive, including the indication that *securityInfo* contains an ESPrim Object. The outer response primitive shall be a NOTIFY response primitive. The parameters of the outer request primitive shall be assigned as described in table 11.4.2-2.

C.10 The Originator shall process the outer response primitive as for normal NOTIFY response primitives. The Originator shall extract the *securityInfo* object, process the indication in *securityInfoType* that *securityInfo* contains an ESPrim Object, and extract the ESPrim Object containing the secured inner response primitive.

C.11 The Originator shall process the ESPrim Object according to the indicated object security technology resulting in the verified serialization of the inner response primitive or an error message. This processing is described in clause 8.4.2 of ETSI TS 118 103 [2].

C.12 The Originator shall process the inner response primitive or error message.

Table 11.4.2-1: NOTIFY Request Message Parameters when using ESPrim

Request message parameter		Mandatory/ Optional for ESPrim	Details
Mandatory	***Operation*** - operation to be executed	M	NOTIFY
	To - the address of the target resource on the target CSE	M	Address of the Receiver's <CSEBase> or <AE> resource
	From - the identifier of the message Originator	M	
	Request Identifier - uniquely identifies a Request message	M	May be independent of the Request Identifier of the inner request primitive
Operation dependent	***Content*** - to be transferred	NP	
	Resource Type - of resource to be created	N/A	N/A
Optional	***Originating Timestamp*** - when the message was built	O	Time when the outer request primitive was built
	Request Expiration Timestamp - when the request message expires	O	Copied from the corresponding parameter in the inner request primitive
	Result Expiration Timestamp - when the result message expires	O	Copied from the corresponding parameter in the inner request primitive
	Operational Execution Time - the time when the specified operation is to be executed by the target CSE	N/A	The operation execution here is the cryptographic operations performed by the Receiver, which shall be executed immediately
	Response Type - type of response that shall be sent to the Originator	O	Any mode may be applied
	Result Persistence - the duration for which the reference containing the responses is to persist	N/A	N/A for NOTIFY
	Result Content - the expected components of the result	N/A	The result content here is the Result Content of the outer primitive, which is always ESPrim Object
	Event Category - indicates how and when the system should deliver the message	O	Copied from the corresponding parameter in the inner request primitive
	Delivery Aggregation - aggregation of requests to the same target CSE is to be used	O	Copied from the corresponding parameter in the inner request primitive
	Group Request Identifier - Identifier added to the group request that is to be fanned out to each member of the group	N/A	This parameter may be present in the inner request primitive, but shall not be present in the outer primitive
	Filter Criteria - conditions for filtered retrieve operation	N/A	N/A for NOTIFY
	Desired Identifier Result Type - format of resource identifiers returned	N/A	N/A for NOTIFY

Table 11.4.2-2: NOTIFY Response Message Parameters when using ESPrim

Response message parameter/success or not	Mandatory/ Optional for ESPrim	Details
Request Identifier - uniquely identifies a Request message	M	Matches corresponding parameter in outer request primitive
Content - to be transferred	NP	ESPrim Object
To - the identifier of the Originator or the Transit CSE that sent the corresponding non-blocking request	O	As for NOTIFY
From - the identifier of the Receiver	O	As for NOTIFY
Originating Timestamp - when the message was built	O	Time when the outer request primitive was built.
Result Expiration Timestamp - when the message expires	O	Copied from the corresponding parameter in the inner request primitive.
Event Category - what event category shall be used for the response message	O	Copied from the corresponding parameter in the inner request primitive.
Content Status	N/A	N/A for NOTIFY
Content Offset	N/A	N/A for NOTIFY

11.4.3 Functional Architecture Specifications for Direct End-to-End Security Certificate-based Key Establishment (ESCertKE)

The ESCertKE procedure comprises the exchange of TLS handshake protocol parameters in four ESCertKE Messages, specified in ETSI TS 118 103 [2]. The AE or CSE initiating the procedure is the *Initiating End-Point* and the *Terminating End-Point* is the AE or CSE with which the ESCertKE Initiating End-Point intends to establish the pairwiseE2EKey.

If an AE or CSE supports ESCertKE, then an indication shall be present in the *e2eSecInfo* attribute in an AE's <*AE*> resource, or a CSE's <*CSEBase*> resource or a CSE's <*remoteCSE*> resource.

The ESCertKE messages and associated processing for ESCertKE are specified in clause 8.7 "End-to-End Certificate-based Key Establishment (ESCertKE)" in ETSI TS 118 103 [2]. The transport details for the ESCertKE Procedure are shown in figure 11.4.3-1, and described in the following text.

NOTE: The outer primitive is not acting on resources because the outer primitive is only used to transport the ESCertKE messages. This is the reason that the NOTIFY procedure is used for the outer primitive.

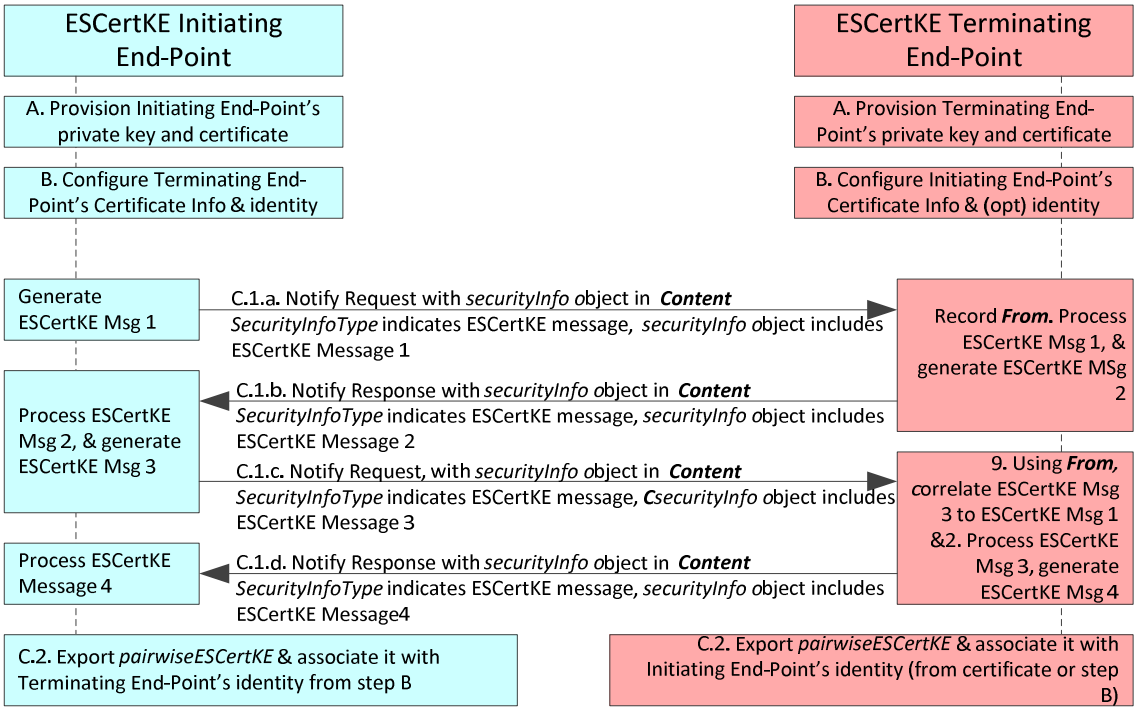


Figure 11.4.3-1: The transport details for the ESCertKE Procedure

A. **Provisioning Certificates:** Each End-Points shall be provisioned with their own private keys and corresponding certificate and optional certificate chain.

B. **Triggering:** The Initiating End-Points decide to initiate the ESCertKE procedure with an identified Terminating End-Point.

C. **Establishing pairwiseE2EKey**

C.1 The Initiating End-Point and Terminating End-Point exchange the sequence of four ESCertKE Messages specified in clause 8.7 "End-to-End Certificate-based Key Establishment" in ETSI TS 118 103 [2]. The ESCertKE Messages are exchange in two sequential NOTIFY procedures:

C.1a ESCertKE Message 1 is sent in a first NOTIFY request from the Initiating End-Point to the End-Point. The Terminating End-Point records the identity of the Initiating End-Point in the ***From*** primitive parameter.

C.1b ESCertKE Message 2 is sent in the resulting NOTIFY response from the Terminating End-Point to the Initiating End-Point.

C.1c ESCertKE Message 3 is sent in a second NOTIFY request from the Initiating End-Point to the End-Point. The Terminating End-Point shall correlate this ESCertKE message with the corresponding ESCertKE Message 1 using the identity of the Initiating End-Point in the ***From*** primitive parameter.

C.1.d ESCertKE Message 4 is sent in the resulting NOTIFY response from the Terminating End-Point to the Initiating End-Point.

The parameters of the NOTIFY primitives shall be assigned as per normal, with the following details specific to ESCertKE:

- *securityInfo Type*: indicating that the ***Content*** contains an ESCertKE Message.
- ***Content***: an ESCertKE Message.

C.2 If the TLS handshake protocol is successful, then the Initiating and Terminating End-Points shall generate and cache a pairwiseE2EKey as described 8.7 "End-to-End Certificate-based Key Establishment" in ETSI TS 118 103 [2].

11.5 Functional Architecture Specifications for Dynamic Authorization

11.5.1 Dynamic Authorization Reference Model

The Dynamic Authorization reference model is shown in figure 11.5.1-1.

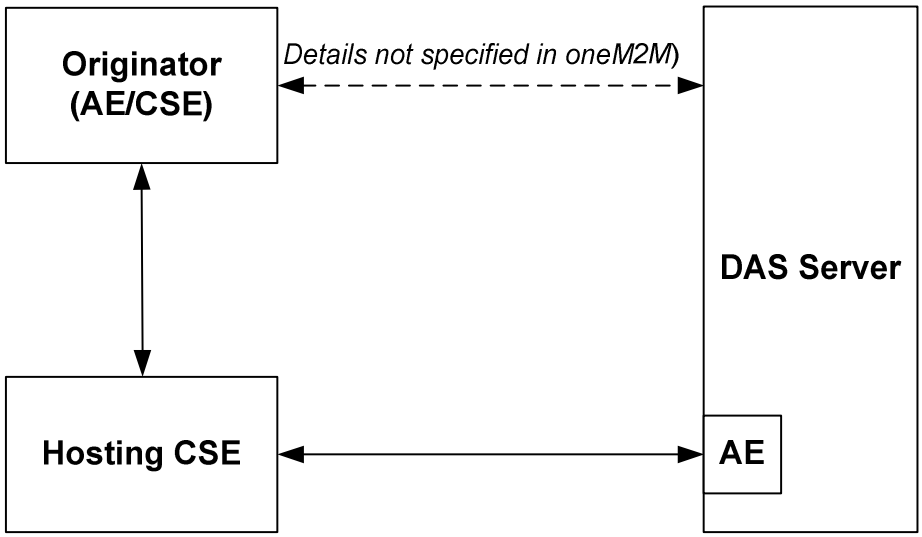


Figure 11.5.1-1: Dynamic Authorization reference model

The Dynamic Authorization reference model introduces the following systems and entities:

- Dynamic Authorization System (DAS): A system supporting dynamically authorization on behalf of resources owners. The present document does not describe the processing and exchange of messages within the Dynamic Authorization System. This system may reside either internally or externally within the service provider network.
- Dynamic Authorization System (DAS) Server: A server configured with policies for dynamic authorization, and provided with credentials for issuing Tokens. The DAS Server may include an AE for interaction with the oneM2M system.

The following Dynamic Authorization procedures are specified:

- **Direct Dynamic Authorization**, summarized in figure 11.5.1-2. In this procedure, Hosting CSE interacts with the DAS Server to obtain Dynamic Authorization. When AE, Hosting CSE and the DAS server support to create the Authorization Relationship Mapping Record, steps 5-7 will be applied.

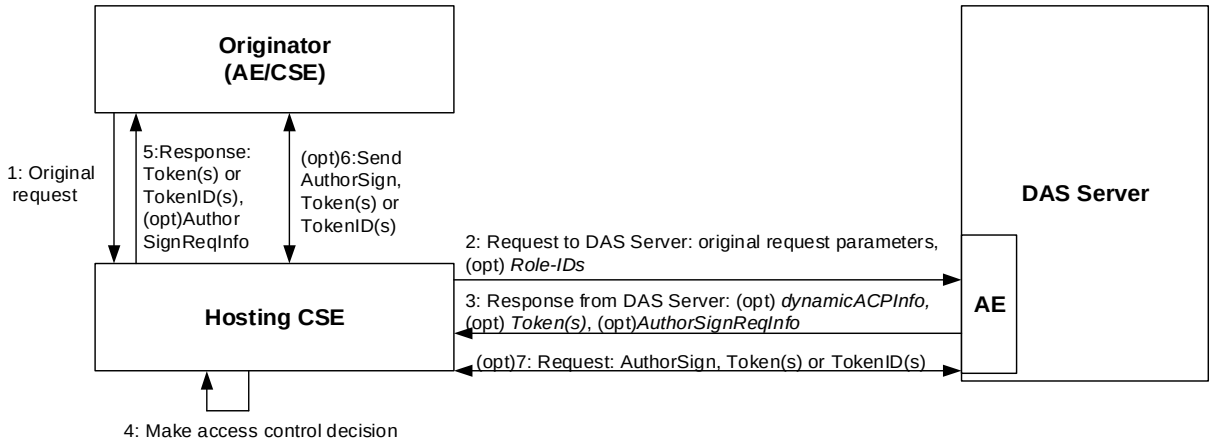


Figure 11.5.1-2: Direct Dynamic Authorization

- **Indirect Dynamic Authorization**, summarized in figure 11.5.1-3:

  - Steps 1-2: The Hosting CSE may provide the Originator with ***Token Request Information*** in the unsuccessful response.

  - Step 3: The Originator interacts with the DAS Server with the intention that the DAS Server issue *Tokens* authorizing the Originator, and the Originator is provided with the Token or a Token-ID. If the Originator is an AE, whose AE-ID-Stem is assigned by the registrar CSE, and both AE and DAS server support to create the Authorization Relationship Mapping Record, the DAS Server shall request the AE to create the authorization relationship mapping record. The interaction is not described in the present specification.

  - Step 4: If the DAS Server starts the process of Authorization Relationship Mapping Record creation in step 3, the AE shall request to create the Authorization Relationship Mapping Record in the DAS Server.

  - Steps 5-8: The Originator provides the Hosting CSE with a *Token, Token-ID* to indicate that the Token is to be considered in the access decision. In the case of a token-ID, the Hosting CSE retrieves the corresponding Token via an AE of the DAS Server. These are then used in the access decision. If the Authorization Relationship Mapping Record is created in step 4, the originator shall also indicate the related information to the Hosting CSE. The Hosting CSE may provide the Originator with a *Local-Token-ID* may be used to identify the Token.

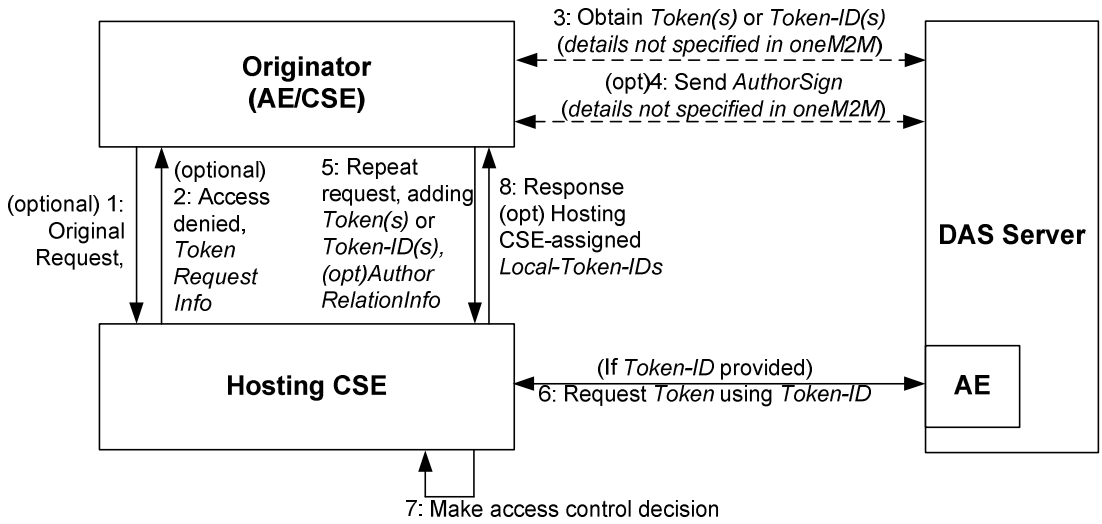


Figure 11.5.1-3: Indirect Dynamic Authorization

11.5.2 Direct Dynamic Authorization

The parameters exchanged for Direct Dynamic Authorization, and the corresponding processing, are specified in clause 7.3.2.2 of ETSI TS 118 103 [2]. The present clause specifies the transportation of parameters when oneM2M primitives are used. The step numbers are aligned with the procedure in clause 7.3.2.2 of ETSI TS 118 103 [2]. Further details for each step in the present clause can be obtained by examining the corresponding steps in clause 7.3.2.2 of ETSI TS 118 103 [2].

The message flow for Direct Dynamic Authorization is shown in figure 11.5.2-1, and described in the following text. This call flow assumes that the Hosting CSE has already received the resource access request from the Originator.

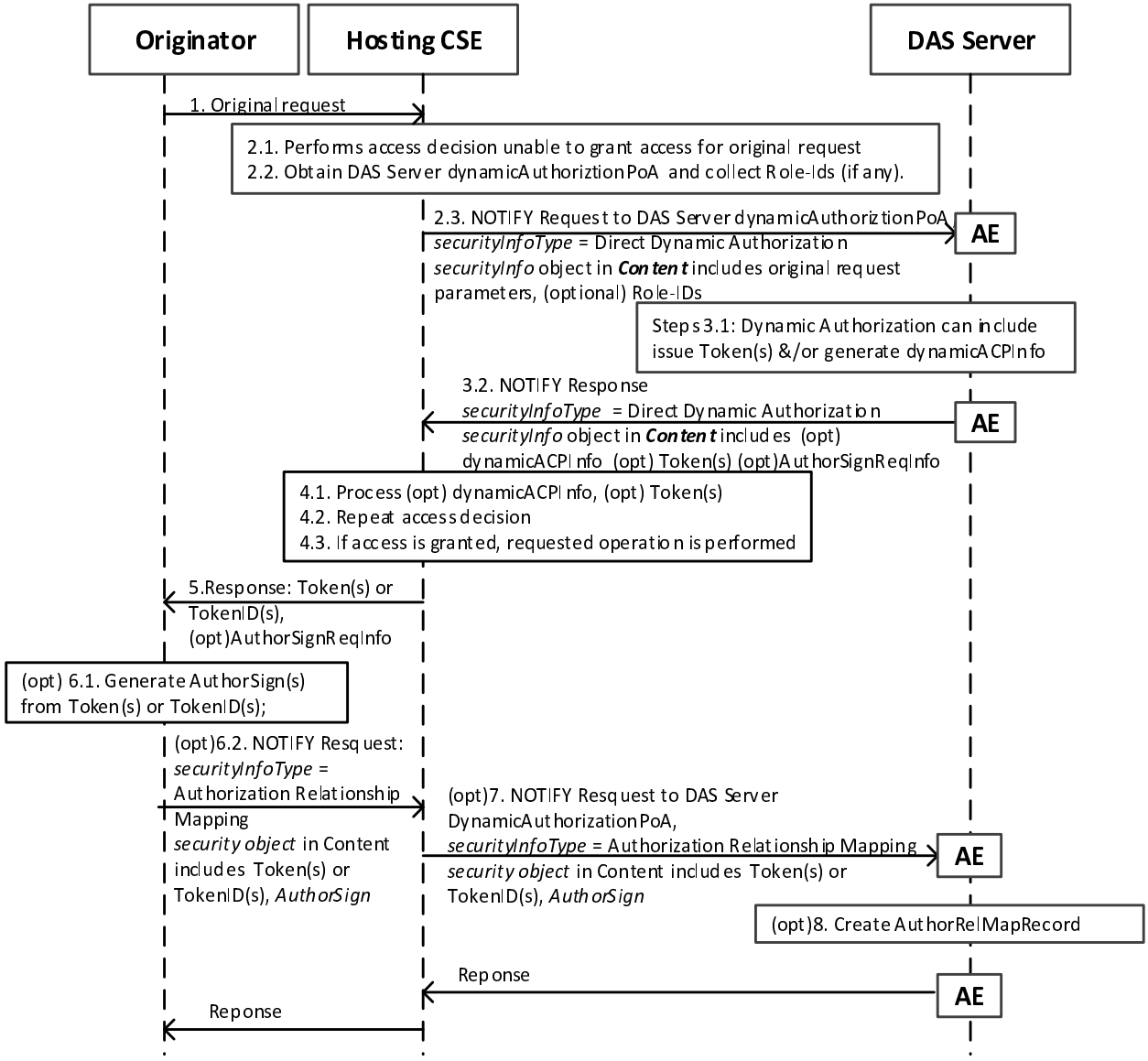


Figure 11.5.2-1: Message flow showing transport details for Direct Dynamic Authorization

- The Originator sends request (called the request from the Originator for this message flow) to the Hosting CSE. This request may include ***Tokens, Token IDs*** or ***LocalToken IDs***; see the clause 11.5.3 "Indirect Dynamic Authorization".

- Initial Hosting CSE processing:

  2.1 If the request from the Originator includes ***Token, Token IDs*** or ***Local Token IDs*** then these are processed as described in clause 11.5.3 "Indirect Dynamic Authorization". The Hosting CSE evaluates the access decision algorithm, but is unable to grant access for the request from the Originator based on configured access control policies.

2.2 The Hosting CSE examines the <*accessControlPolicy*> resources and <*dynamicAuthorizationConsultation*> resources to obtain the DAS Server dynamicAuthorizationPoA with which it may perform Direct Dynamic Authorization. The Hosting CSE selects a DAS Server and forms the set of applicable Role-IDs (if any) to send to the corresponding DAS Server.

2.3 The Hosting CSE shall send a Notify request primitive to the DAS Server AE, with the following details specific to Direct Dynamic Authorization:

- The *securityInfo Type* element shall indicate that the Notify request primitive is for Direct Dynamic Authorization.
- The ***Content*** parameter shall contain information that the DAS Server can use in deciding what Dynamic Authorizations should be applied. This information includes primitive parameters from the request from the Originator and the set of applicable Role-IDs (if any). Clause 7.3.2.2 of ETSI TS 118 103 [2] lists the primitive parameters to be included.

- DAS Server processing:

3.1 The DAS Server shall extract and parse the ***Content*** parameter of the received message. The DAS Server may issue *Token(s)* and/or generate dynamicACPInfo which will be used by the Hosting CSE to create a dynamic <*accesscontrolPolicy*> resource.

3.2 The DAS Server shall send a Notify response primitive via the DAS Server AE to the Hosting CSE, with the following details specific to Direct Dynamic Authorization:

- The *securityInfo Type* element shall indicate that the Notify response primitive is for Direct Dynamic Authorization.
- If step 3.1 resulted in a Token(s) and/or dynamicACPInfo parameter, then these parameters shall be included in the ***Content*** parameter, otherwise the ***Content*** parameter shall not be present.
- In the case the DAS Server issues a Token(s), if in the step 3.1 DAS Server extracts the *AuthorSignIndicator* from the received message and the DAS server itself also supports to trigger creating the authorization relationship mapping record, an *AuthorSignReqInfo* shall be included in the ***Content*** parameter.

- Hosting CSE Processing:

4.1 The Hosting CSE shall process the ***Content*** parameter (if present) of the NOTIFY Response from the DAS Server:

- The Hosting CSE shall verify and cache the Token(s) in the list (if present), described in clause 7.3.2.2 of ETSI TS 118 103 [2].
- The Hosting CSE shall create a dynamic <*accessControlPolicy*> resource from dynamicACPInfo (if present).

4.2 The Hosting CSE repeats the access decision mechanism.

4.3 If access is granted, then the Hosting CSE performs the operation requested in the request from the Originator.

- If in the step 4.1, the Hosting CSE gets AuthorSignReqInfo from the ***Content*** parameter, the Hosting CSE shall forward AuthorSignReqInfo in a response primitive to Originator to request the AuthorSign. If the AuthorSignReqInfo is not included in the Content, then the steps 6-8 are not applied.

- Originator Processing:

6.1 If the Originator receives *AuthorSignReqInfo*, it shall generate AuthorSign for each Token. How the AuthorSign are generated is described in clause 7.3.2.2 of ETSI TS 118 103 [2].

6.2 The Originator shall send a Notify request primitive to the Hosting CSE, with the following details:

- The *securityInfo Type* element shall indicate that the Notify request primitive is for Authorization Relationship Mapping.

- AuthorSign with the corresponding Token(s) or TokenID(s) shall be included in the ***Content*** parameter.

- The Hosting CSE Processing shall forward this Notify request primitive to the DAS Server AE.
- The DAS server AE shall create authorization relationship mapping record using the AuthorSign described in clause 7.3.2.2 of ETSI TS 118 103 [2].

11.5.3 Indirect Dynamic Authorization

The parameters exchanged for Indirect Dynamic Authorization, and the corresponding processing, are specified in clause 7.3.2.3 of ETSI TS 118 103 [2]. The present clause specifies the transportation of parameters when oneM2M primitives are used. Further details for each step in the present clause can be obtained by examining the corresponding steps in clause 7.3.2.3 of ETSI TS 118 103 [2].

The message flow for the Indirect Dynamic Authorization Procedure is shown in figure 11.5.3-1, and described in the following text.

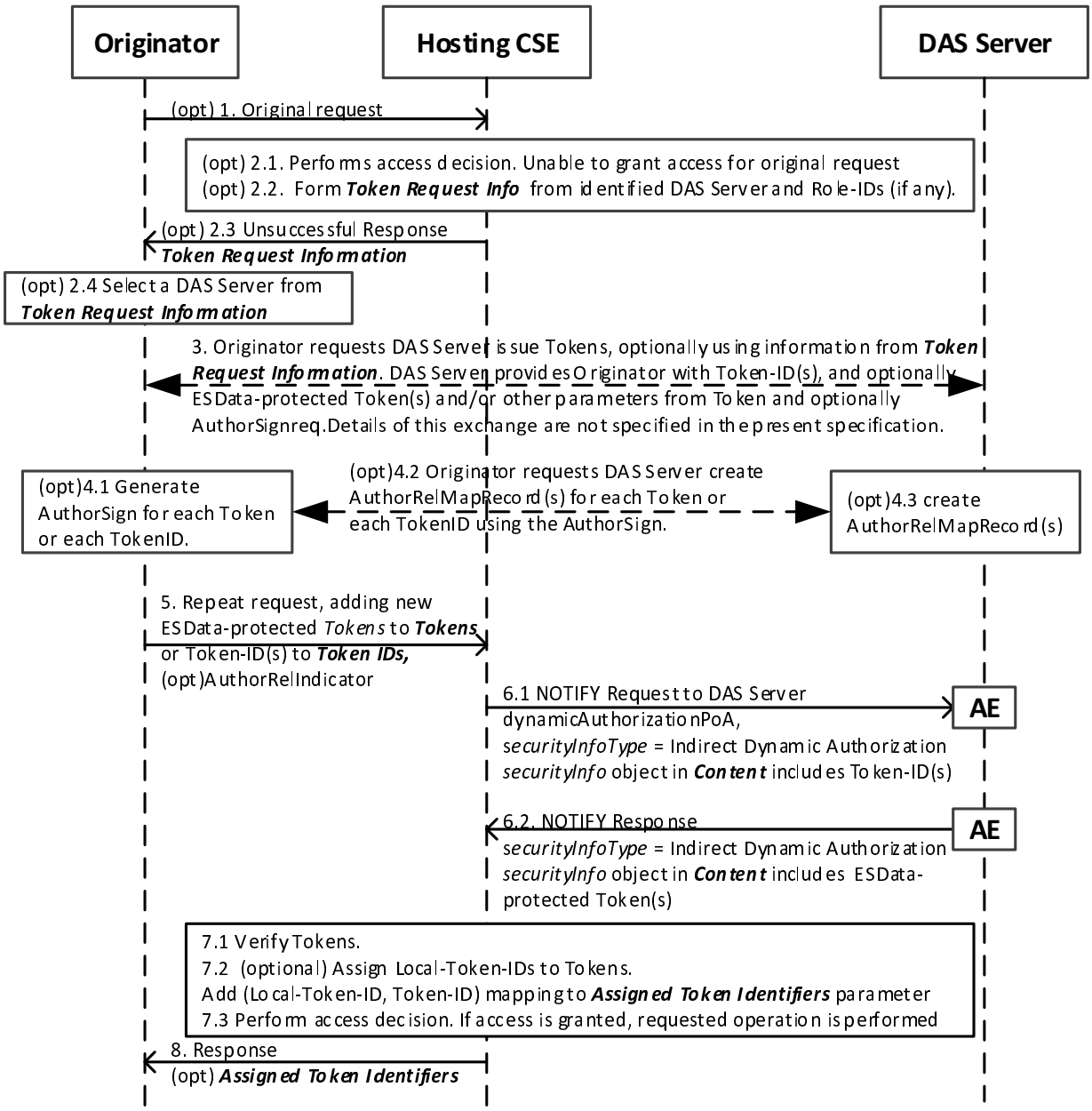


Figure 11.5.3-1: Message flow for Indirect Dynamic Authorization

- (Optional) The Originator sends request to the Hosting CSE. This request may include ***Tokens, Token IDs*** or ***Local Token IDs***, but this message flow assumes that these do not provide sufficient permissions for accessing the requested resource.

- (Optional) Initial Hosting CSE processing:

  2.1 Hosting CSE performs the access decision for the request from the Originator. This call flow assumes that the request from the Originator is denied as a result of the access decision.

  2.2 The Hosting CSE forms the ***Token Request Information*** primitive parameter.

  2.3 The Hosting CSE shall send, to the Originator, an unsuccessful resource access response with the following details specific to the Indirect Dynamic Authorization procedure:

  - The ***Response Status Code*** shall be set to "UNAUTHORIZED".
  - The ***Token Request Information*** primitive parameter shall be included.

  2.4 The Originator selects a DAS Server identified in ***Token Request Information*** primitive parameter.

- The Originator shall interact with the DAS Server to request the issuance of one or more Tokens. The Originator can provide information for the DAS Server provided in the ***Token Request Information***, and parameters from the original resource access request. If the Originator is AE and the AE-ID-Stem is assigned by the registrar CSE of the AE, and the Originator supports to create the authorization relationship mapping record, then the Originator shall provide *AuthorSignIndicator* parameter. The DAS Server issues a Token(s) and provides the Token-ID(s) and optionally the ESData-protected Token(s) to the Originator. The DAS Server can also provide the Originator with other parameters from the Token; for example, the time window in which the Token is valid. If DAS Server receives the *AuthorSignIndicator* from the Originator, and the DAS server supports to create the authorization relationship mapping record, then the DAS server shall provide the Originator with a *AuthorSignReqInfo* to request the Originator to return *AuthorSign(s)* for each Token. This interaction is specific to the Dynamic Authorization System technology being used.

- If the Originator receives an *AuthorSignReqInfo* from DAS server, the Originator shall return the AuthorSignto DAS server:

  4.1 The Originator generates *AuthorSign* for each Token described in clause 7.3.2.2 of ETSI TS 118 103 [2].

  4.2 The Originator sends *AuthorSign* to DAS server with the corresponding Token(s) or TokenID(s).

  4.3 The DAS server shall create, for each Token, the authorization relationship mapping record containing the information listed in table 7.3.2.2-3 of ETSI TS 118 103 [2].

- The Originator shall repeat the original resource access request, with the following changes:

  - ***Tokens*****:** add the ESData-protected Token(s) provided by the DAS Server; and
  - ***Token IDs*****:** add Token-ID if the ESData-protected Token(s) was not provided by the DAS Server.
  - ***AuthorRelIndicator:*** add AuthorRelIndicator to indicate that the relationship between the AE and the Token(s) are maintained in the DAS server.

- (Optional) If the request includes Token-DI(s), then for each Token-ID the Hosting CSE identifies the corresponding DAS Server AE from which to request the corresponding Token, and the following steps shall be performed. The Hosting CSE may collect the Token-ID(s) corresponding to a single DAS Server and perform the following steps once rather than repeating the steps for each token:

  6.1 The Hosting CSE shall send a Notify request primitive to the DAS Server AE, with the following details specific to Indirect Dynamic Authorization:

  - The *securityInfo Type* object parameter shall indicate that the Notify request primitive is for Indirect Dynamic Authorization.
  - The ***Content*** parameter shall contain the Token-ID(s) associated with that DAS Server.

6.2 The DAS Server shall send a Notify response primitive via the DAS Server AE to the Hosting CSE, with the following details specific to Direct Dynamic Authorization:

- The *securityInfo Type* object parameter shall indicate that the Notify response primitive is for Indirect Dynamic Authorization [2].
- The ***Content*** parameter shall contain the valid ESData-protected Token(s) corresponding to the supplied Token-ID(s). The DAS Server shall provide only those Token(s) which are applicable to the Hosting CSE.

- Hosting CSE Processing:

7.1 The Hosting CSE shall process the ESData-protected Token(s) to extract the authenticated Token(s). Additional checking shall also be applied The Hosting CSE may cache the Token(s), as described in clause 7.3.2.3 of ETSI TS 118 103 [2].

7.2 The Hosting CSE may assign Local-Token-ID(s) to cached Token(s).

7.3 The Hosting CSE shall perform the access decision, including the Token(s) identified in the request. If access is granted, then the requested operation shall be performed.

- Response:

8.1 The Hosting CSE may send a response to the Originator. For each new Local-Token-ID(s) has been assigned, the Local-Token-ID and corresponding Token-ID shall be included in the ***Assigned Token Identifiers*** parameter of the response.

8.2 The Originator shall associate the *Local-Token-ID* with Token-ID. In subsequent requests, the Originator may use the Local-Token-ID instead of the *Token* or Token-ID.

11.5.4 AE Authorization Relationship Update

11.5.4.1 AE Direct Authorization Relationship Update

The parameters exchanged for AE Direct Authorization Relationship Update, and the corresponding processing, are specified in clause 7.3.2.7.1 of ETSI TS 118 103 [2]. The present clause specifies the transportation of parameters when oneM2M primitives are used. The step numbers are aligned with the procedure in clause 7.3.2.7.1 of ETSI TS 118 103 [2]. Further details for each step in the present clause can be obtained by examining the corresponding steps in clause 7.3.2.7.1 of ETSI TS 118 103 [2].

The message flow for the Direct Authorization Relationship Update is shown in figure 11.5.4.1-1, which is described in the following text.

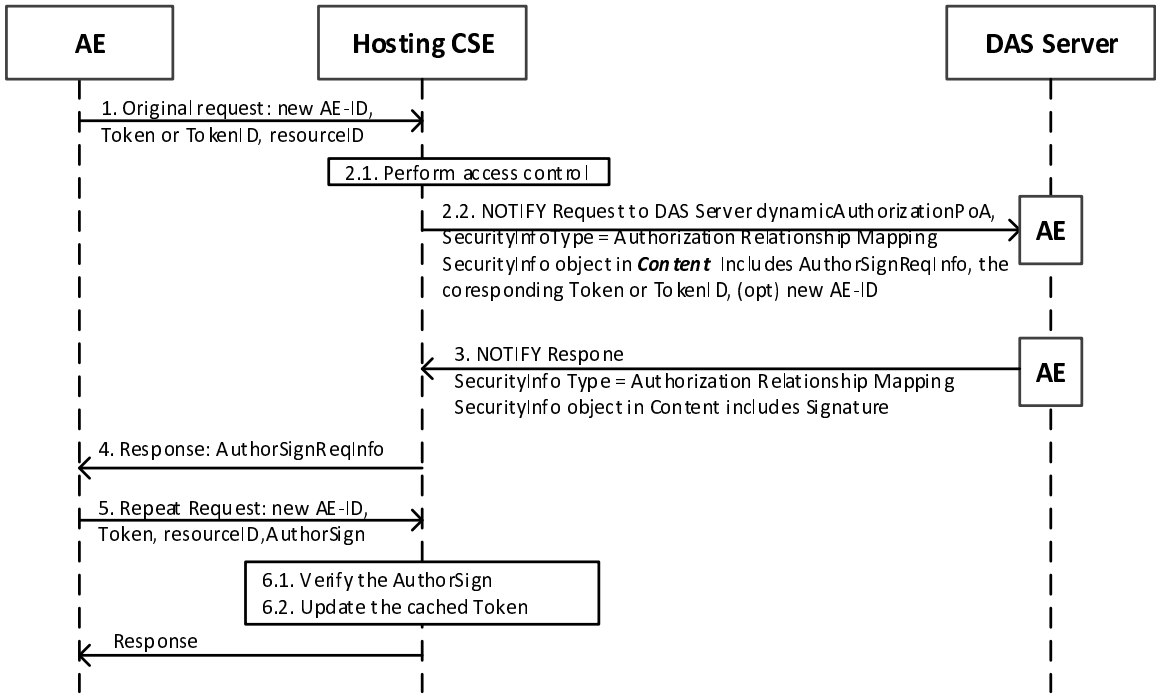


Figure 11.5.4.1-1: AE Direct Authorization Relationship Update

- An AE sends a resource access request message to a Hosting CSE, which carries the new AE-ID, and the Token or the TokenID issued for it.
- Hosting CSE processing

  2.1 The Hosting CSE shall verify this Token or the Token identified by this TokenID. If the result is valid, and the *holder* attribute of the Token is not equal to the new AE-ID of the originator, the Hosting CSE performs the following steps to verify whether the AE has the possession of the Token.

  2.2 The Hosting CSE sends a Notify request primitive to DAS Server AE, with the following details:

  - The *securityInfo Type* element shall indicate that the Notify request primitive is for Authorization Relationship Mapping.
  - The ***Content*** parameter shall contain information: *AuthorSignReqInfo*, the corresponding Token or TokenID received from the AE.

- The DAS Server AE shall extract and parse the ***Content*** parameter of the received message. The DAS Server shall examine its authorization relationship mapping record list. If there is a record whose Token parameter is equal to the Token included in the ***Content*** parameter or the Token identified by the Token ID in the ***Content*** parameter, the DAS Server AE sends a Notify response primitive to the Hosting CSE including the AuthorSign stored in this record:
  - The securityInfo Type element shall indicate that the Notify response primitive is for Authorization Relationship Mapping.
  - The ***Content*** parameter shall contain information: *Signature* which is described in table 7.3.2.2-3 of ETSI TS 118 103 [2].
- The Hosting CSE rejects the request to access the resource, including a AuthorSignReqInfo in the response message to indicate AE to return the *AuthorSign* for this Token.
- The AE sends the repeat request including the information: AuthorSign, resourceID and Token.
- After receiving the AuthorSign, Hosting CSE shall check whether the AuthorSign is equal to the *Signature* returned from DAS Server AE. And if they are the same, the Hosting CSE shall update the cached Token as described in clause 7.3.2.7.1 of ETSI TS 118 103 [2].

11.5.4.2 AE Indirect Authorization Relationship Update

The parameters exchanged for Indirect Authorization Relationship Update, and the corresponding processing, are specified in clause 7.3.2.7.2 of ETSI TS 118 103 [2]. The present clause specifies the transportation of parameters when oneM2M primitives are used. Further details for each step in the present clause can be obtained by examining the corresponding steps in clause 7.3.2.7.2 of ETSI TS 118 103 [2].

The message flow for the Indirect Authorization Relationship Update is shown in figure 11.5.4.2-1, which is described in the following text.

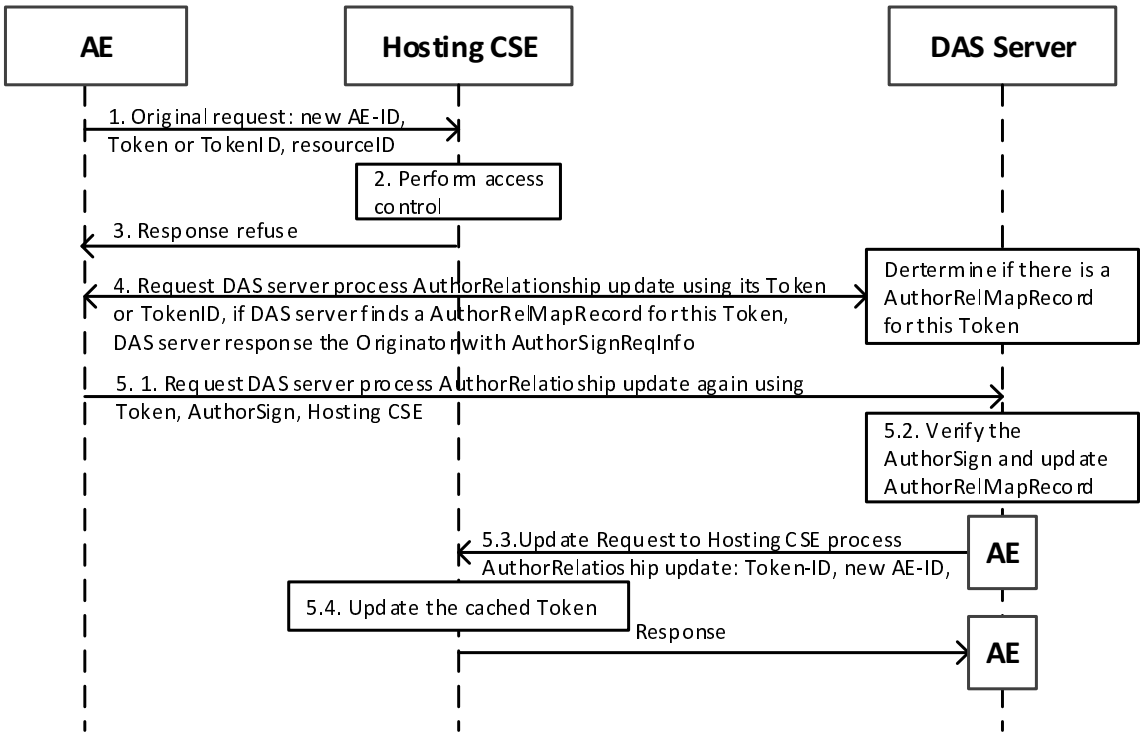


Figure 11.5.4.2-1: AE Indirect Authorization Relationship Update

- The AE sends a request to the Hosting CSE. This request may include the new AE-ID, and Token or TokenID issued for it.
- The Hosting CSE performs the access decision for the request from the Originator. This call flow assumes that the Token or Token identified by the TokenID is valid and the holder attribute of the cached Token is not equal to the new AE-ID of the originator.
- The Hosting CSE shall send, to the Originator, an unsuccessful resource access.
- The AE requests DAS Server to update the authorization relationship using the Token or TokenID, and DAS server shall search if there is an Authorization Relationship Mapping Record of which the Token parameter or TokenID of the Token parameter is the same as the Token or TokenID received from AE. If the result is ok, DAS server shall return an AuthorSignReqInfo to AE to request the AuthorSign for the Token.
- The AE provides the AuthorSign to prove the possession of the token:

  5.1 The AE sends the update request containing Hosting CSE ID, *AuthorSign*, resourceID and Token.

  5.2 After receiving the *AuthorSign*, the DAS Server shall check if this *AuthorSign* is equal to *Signature* stored in the Authorization Relationship Mapping Record corresponding to this Token.

  5.3 If the result in step 5.2 is ok, the DAS Server AE shall send an Update request primitive to the Hosting CSE to update the cached Token.

NOTE: Before the update action, DAS Server AE may get the resource ID of the cached Token locally stored on the Hosting CSE through discovery or other offline ways, there is no limitation on how to get the resource ID of the cached Token.

5.4 The Hosting CSE updates the cached Token as described in clause 7.3.2.7.2 of ETSI TS 118 103 [2].

11.6 Functional Architecture Specifications for Distributed Authorization

11.6.1 Distributed Authorization Reference Model

The Distributed Authorization reference model is shown in figure 11.6.1-1. This reference model comprises four subcomponents:

- Policy Enforcement Point (PEP): This component intercepts resource access requests, makes access control decision requests, and enforces access control decisions. The PEP coexists with the entity that needs authorization services.
- Policy Decision Point (PDP): This component interacts with the PRP and PIP to get applicable authorization polices and attributes needed for evaluating authorization policies respectively, and then evaluates access request using authorization policies for rendering an access control decision.
- Policy Retrieval Point (PRP): This component obtains applicable authorization policies according to an access control decision request. These applicable policies should be combined in order to get a finial access control decision.
- Policy Information Point (PIP): This component provides attributes that are needed for evaluating authorization policies, for example the IP address of the requester, creation time of the resource, current time or location information of the requester.

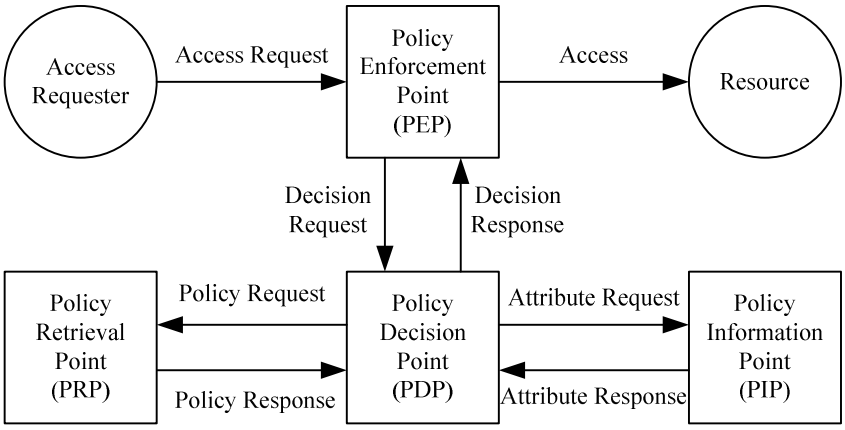


Figure 11.6.1-1: Distributed Authorization reference model

A Distributed Authorization system may comprise any of the subcomponents: PDP, PRP and/or PIP. This means that the subcomponents PEP, PRP, PDP and PIP may be distributed across different nodes. For example the PEP is located in an ASN/MN and the PDP is located in the IN.

The generic distributed authorization procedure is described in clause 7 of ETSI TS 118 103 [2].

11.6.2 Interactions between Authorization Components

Interactions with PDP

A CSE that acts as a PEP or PDP may send an access control decision request to another CSE that acts as a PDP. The access control decision request and response shall be encapsulated into UPDATE request and response respectively. The UPDATE request shall address an <*authorizationDecision*> resource. The relevant details are provided in clause 9.6.41.

In the case the access control decision requester is the Hosting CSE, it obtains the address of an <*authorizationDecision*> resource from the *authorizationDecisionResourceIDs* attribute of the <*accessControlPolicy*> resource that is bound to the target resource that the Originator wants to access. In other cases how the access control decision requester obtains the address of an <*authorizationDecision*> resource is out of scope of the present document.

See clause 7 of ETSI TS 118 103 [2] for further details.

Interactions with PRP

A CSE that acts as a PDP or PRP may send an access control policy request to another CSE that acts as a PRP. The access control policy request and response shall be encapsulated into UPDATE request and response respectively. The UPDATE request shall address an <*authorizationPolicy*> resource. The relevant details are provided in clause 9.6.42.

In the case the access control policy requester is the Hosting CSE, it obtains the address of an <*authorizationPolicy*> resource from the *authorizationPolicyResourceIDs* attribute of the <*accessControlPolicy*> resource that is bound to the target resource that the Originator wants to access. In other cases how the access control policy requester obtains the address of an <*authorizationPolicy*> resource is out of scope of the present document.

See clause 7 of ETSI TS 118 103 [2] for further details.

Interactions with PIP

A CSE that acts as a PDP or PIP may send an access control information request to another CSE that acts as a PIP. The access control information request and response shall be encapsulated into UPDATE request and response respectively. The UPDATE request shall address an <*authorizationInformation*> resource. The relevant details are provided in clause 9.6.43.

In the case the access control information requester is the Hosting CSE, it obtains the address of an <*authorizationInformation*> resource from the *authorizationInformationResourceIDs* attribute of the <*accessControlPolicy*> resource that is bound to the target resource that the Originator wants to access. In other cases how the access control policy requester obtains the address of an <*authorizationPolicy*> resource is out of scope of the present document.

See clause 7 of ETSI TS 118 103 [2] for further details.

12 Information Recording

12.1 M2M Infrastructure Node (IN) Information Recording

12.1.0 Overview

Various informational elements have to be recorded by the M2M infrastructure nodes for a variety of reasons including but not limited to statistics, charging, maintenance, diagnostics, etc.

This clause describes a framework for recording the necessary information by infrastructure nodes.

12.1.1 Information Recording Triggers

Triggers have to be configured in the IN node by the M2M service provider to initiate information recording.

The M2M infrastructure nodes shall be able to initiate recording based on any of the following triggers:

- A request received by the M2M IN over the Mcc reference point.
- A request received by the M2M IN over the Mca reference point.
- A request initiated by the M2M IN over any reference point.
- Timer- based triggers for non- request based information recording. This trigger is used only when the memory size of a container over a period of time is required.

More than one trigger can be simultaneously configured.

The recording triggers may also be configurable, for example, as follows:

- On a per CSE basis, or a group of CSEs for requests originating/arriving from/at the M2M IN.
- On a per AE basis or a group of AEs.
- The default behaviour is that no CSEs/AEs are configured.

12.1.2 M2M Recorded Information Elements

12.1.2.1 Unit of Recording

A unit of recording refers to a number of informational elements recorded by the IN and that can be used as a basis for additional post-processing for a specific purpose such as generating Charging Data Records (CDRs), statistics, etc. In that respect, each unit of recording can be thought of as an M2M information record. The actual informational elements that make up a recording unit shall be described later.

For request-based triggers, as defined in clause 12.1.1, the unit of recording shall include a request and its response.

A unit of recording shall be referred to as an M2M Event Record. This shall apply to all recording triggers as defined in clause 12.1.1.

12.1.2.2 Information Elements within an M2M Event Record

The information elements within an M2M event record are defined in table 12.1.2.2-1.

Every M2M event record shall be tagged to depict its content according to the following classification:

- Data related procedures: represent procedures associated with data storage or retrieval from the M2M IN (e.g. Container related procedures).
- Control related procedures: represent all procedures that are not associated with data storage/retrieval from the M2M IN with the exclusion of group and device management related procedures (e.g. subscription procedures, registration).
- Group related procedures: represent procedures that handle groups. The group name may be derived from the target resource in these cases.
- Device Management Procedures.
- Occupancy based trigger for recording the occupancy as described in clause 12.1.1.

Table 12.1.2.2-1: Information Elements within an M2M Event Record

Information Element	For request based triggers Mandatory / optional	For timer based triggers Mandatory / optional	Description
M2M Service Subscription Identifier	M	M	The M2M Service Subscription Identifier associated with the request. This is inserted by the IN (see clause 12.1.3)
Application Entity ID	CM (when applicable)	NA	The M2M Application Entity ID if applicable
External ID	CM (when Applicable)	NA	The external ID to communicate over **Mcn** where applicable
Receiver	M	NA	Receiver of an M2M request (can be any M2M Node)
Originator	M	NA	Originator of the M2M request (can be any M2M Node)
Hosting CSE-ID	O	NA	The hosting CSE-ID for the request in case the receiver is not the host, where applicable
Target ID	M	NA	The target URL for the M2M request if available. Alternatively can be the target resource identifier
Protocol Type	O	NA	Used Protocol Binding (e.g. HTTP, CoAP, MQTT)
Request Operation	O	NA	Request Operation as defined in clause 8.1.2
Request Headers size	O	NA	Number of bytes for the headers in the Request (All Request parameters of the used protocol per the Protocol Type information element)
Request Body size	O	NA	Number of bytes of the body transported in the Request if applicable
Response Headers size	O	NA	Number of bytes for the headers in the Response (All Response parameters of the used protocol per the Protocol Type information element)
Response Body size	O	NA	Number of bytes of the body transported in the Response if applicable
Response Status Code	O	NA	
Time Stamp	M	M	Time of recording the M2M event
M2M-Event-Record-Tag	M	M	A Tag for the M2M event record for classification purposes. This tag is inserted by the IN and is M2M SP specific
Control Memory Size	O	NA	Storage Memory (in bytes), where applicable, to store control related information associated with the M2M event record(excludes data storage associated with container related operations)
Data Memory Size	O	NA	Storage Memory (in bytes), where applicable, to store data associated with container related operations
Access Network Identifier	O	O	Identifier of the access network associated with the M2M event record
Additional Information	O		Vendor specific information
Occupancy	NA	M	Overall size (in Bytes) of the containers generated by a set of AEs identified by the M2M Service Subscription Identifier
Group Name	CM	NA	The Group name (not necessarily unique) shall be included by the IN-CSE in the case where the fanning operations initiated by the M2M IN-CSE
maxNrOfMembers	O	NA	Maximum number of members of the group for Create and Update operation
currentNrOfMembers	O	NA	Current number of members in a group. The request shall be logged and information elements shall be recorded from the request before processing it or sending it out. After obtaining corresponding response, *currentNrOfMembers* shall be updated with the values from the response
Subgroup Name	CM	NA	Subgroup name (not necessarily unique) shall be included i in the case when the IN-CSE initiates a fanning operation
M2M-Node-Id	M	NA	The node Id for the node generating the Accounting-Record-Number for the Diameter ACR. This shall be set to the CSE-ID for the IN-CSE node

The choice for the mandatory elements is motivated by the need to include all M2M identifiers within an M2M event record so that it is possible to support multiple charging scenarios.

For all non-mandatory elements, the M2M IN shall be configurable by the M2M service provider to select any additional desired information to be recorded in addition to the mandatory elements.

12.1.3 Identities Associations in Support of Recorded Information

To enable the M2M IN to record the necessary information, as described above, the following associations shall be maintained by the M2M service provider:

- The CSE-ID (for all M2M Nodes in the M2M framework) and the allocated M2M Service Subscription Identifier.
- The AE-ID and the allocated M2M Service Subscription Identifier.

For established associations, as described above, the M2M IN shall derive the appropriate M2M Service Subscription Identifier for insertion in the M2M record event.

12.2 Offline Charging

12.2.1 Architecture

Figure 12.2.1-1 depicts the charging architecture. Charging information, in the form of charging data records (CDRs), shall be derived from recorded information, and transferred to a Charging Server. As such, it is essential that all information required for charging shall be first selected for recording. There shall be a 1 to 1 mapping between a M2M Event Record and a CDR.

The Charging Function (CHF included within the SCA CSF) embedded within the M2M IN is responsible for interaction with the Charging Server using the Mch reference point.

Billing aspects are out of scope.

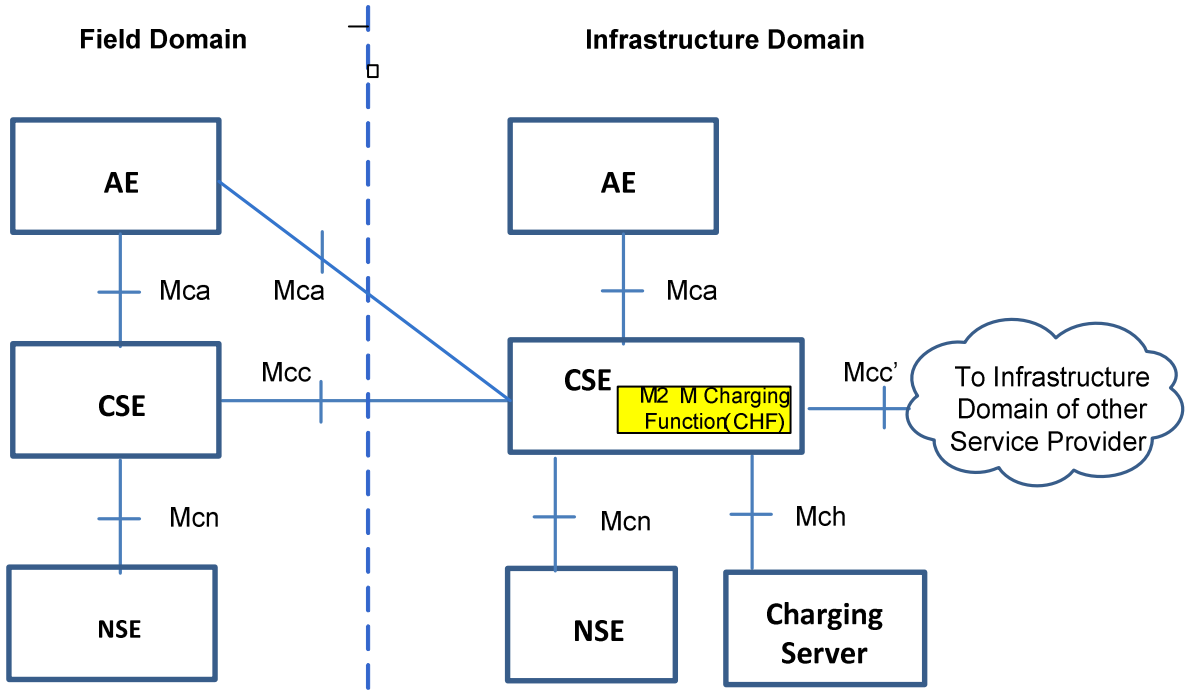


Figure 12.2.1-1: Offline Charging Architecture

Communication flows which transfer CDRs generated by the IN to an external charging server cross the Mch reference point. The Mch reference point may be mapped to reference points of other specifications. E.g. for a 3GPP Underlying Network, the Mch reference point maps to the Rf reference point enabling a 3GPP charging server to be used for oneM2M CDRs.

12.2.2 Filtering of Recorded Information for Offline Charging

Recorded information is the basis for offline charging. To fulfil the needs of different billing systems not all recorded information is required in all cases. Hence, the M2M Charging Function shall be configurable to only select the desired information from the recorded information for transfer to the Charging Server. This configuration shall support selecting the desired information based on the following capabilities:

- On a per CSE basis, or a group of CSEs, for requests originating/arriving from/at the IN. This applies to all M2M Nodes within the M2M framework.
- On a per AE basis or a group of AEs.
- The default behaviour is that no CSEs/AEs are configured.

The charging function shall ensure that information selected for transfer to the charging server has also been selected for recording before a configuration is deemed acceptable for execution.

12.2.3 Examples of Charging Scenarios

12.2.3.0 Overview

Charging scenarios refer to scenarios for which an M2M entity can be billed if the scenario is deemed billable by the M2M service provider. Some charging scenarios may require single CDR. Other scenarios may require multiple CDRs, and suitable correlation information shall have to be identified to select the CDRs for the charging scenario in this case.

The following clause lists some potential charging scenarios as examples only. Each scenario shall require the appropriate configuration of the CHF, and for that matter the M2M recording functions, to ensure that all pertinent data is available.

12.2.3.1 Example Charging Scenario 1 - Data Storage Resource Consumption

In this scenario, the M2M entity that stores application data, using container procedures for that purpose, will be billed, for storage resources within the M2M IN, until such time as the resources are deleted. This scenario will require correlation between multiple CDRs to identify the entity that stored the data, the entity that deleted the same data, and the duration and amount of storage.

12.2.3.2 Example Charging Scenario 2 - Data transfer

In this scenario, the M2M entity that retrieves/stores container data will be billed for the amount of transferred data.

12.2.3.3 Example Charging Scenario 3 - Connectivity

This scenario is relevant for an M2M entity that contacts the M2M IN frequently to transfer small amounts of data for storage. In this scenario, the M2M entity will be charged for the connectivity as opposed to the stored amount of data. The same applies to an M2M entity that also contacts frequently the M2M IN to retrieve stored data.

12.2.4 Definition of Charging Information

12.2.4.0 Overview

Charging information in the form of CDR is essentially a subset of the information elements within the M2M event records recorded by the M2M IN for transmission over the Mch reference point.

12.2.4.1 Triggers for Charging Information

The charging function within the M2M IN shall initiate transmission of CDRs if configured for that purpose in accordance with clause 12.2.2.

12.2.4.2 Charging Messages over Mch Reference Point

The Mch shall be used in case the CDRs are to be transferred to an external Charging Server. It is assumed that the Mch is equivalent to the Rf reference point as defined in [i.15] and [i.16].

Hence, every CDR shall be transferred in a single message, namely Accounting-Request and that elicits a response, namely Accounting-Answer.

Table 12.2.4.2-1 describes the use of these messages for offline charging.

Table 12.2.4.2-1: Offline charging messages reference table

Request-Name	Source	Destination	Abbreviation
Accounting-Request	M2M IN	Charging Server	ACR
Accounting-Answer	Charging Server	M2M IN	ACA

12.2.4.3 Structure of the Accounting Message Formats

12.2.4.3.1 Accounting-Request Message

Table 12.2.4.3.1-1 illustrates the basic structure of an ACR message generated from the M2M IN for offline charging in accordance with [i.15], [i.16], [i.8] and [i.11].

Table 12.2.4.3.1-1: Accounting-Request (ACR) message contents

Informational Element	Category	Description
Session-Id	M	This field identifies the operation session. The usage of this field is left to the M2M SP.
Origin-Host	M	This field contains the identification of the source point of the operation and the realm of the operation Originator.
Origin-Realm	M	This field contains the realm of the operation Originator.
Destination-Realm	M	This field contains the realm of the operator domain. The realm will be addressed with the domain address of the corresponding public URI.
Accounting-Record-Type	M	This field defines the transfer type: This field shall always set to event based charging.
Accounting-Record-Number	M	This field contains the sequence number of the transferred messages.
Acct-Application-Id	O_C	Advertises support for accounting for M2M.
Origin-State-Id	O_C	This is a monotonically increasing value that is advanced whenever a Diameter entity restarts with loss of previous state, for example upon reboot.
Event-Timestamp	O	Defines the time when the event occurred.
Destination-Host	O_C	This is the intended destination for the message.
Proxy-Info	O_C	Includes host information about a proxy that added information during routing of the message.
Route-Record	O_C	This field contains an identifier inserted by a relaying or proxying charging node to identify the node it received the message from.
Service-Context-Id	M	This field identifies the M2M domain.
Service-Information	M	This is a grouped field that holds the M2M specific parameters.
Subscription-Id	M	Identifies the M2M Service Subscription Identifier.
M2M Information	M	This parameter holds the M2M informational element specified in table 12.1.2.2-1 with the exception of the M2M Service Subscription Identifier.
Proprietaryinformation	O	This is for proprietary information.
O_C	This is a parameter that, if provisioned by the service provider to be present, shall be included in the CDRs when the required conditions are met. In other words, an O_C parameter that is configured to be present is a conditional parameter.	

12.2.4.3.2 Accounting-Answer Message

Table 12.2.4.3.2-1 illustrates the basic structure of an ACA message generated by the charging server as a response to an ACR message.

Table 12.2.4.3.2-1: Accounting-Answer (ACA) message contents

Information element	Category	Description
Session-Id	M	Same as table 12.2.4.3.1-1
Origin-Host	M	Same as table 12.2.4.3.1-1
Origin-Realm	M	Same as table 12.2.4.3.1-1
Accounting-Record-Type	M	Same as table 12.2.4.3.1-1
Accounting-Record-Number	M	Same as table 12.2.4.3.1-1
Acct-Application-Id	O_C	Same as table 12.2.4.3.1-1
Origin-State-Id	O_C	This is a monotonically increasing value that is advanced whenever a Diameter entity restarts with loss of previous state, for example upon reboot
Event-Timestamp	O	Same as table 12.2.4.3.1-1
Proxy-Info	O_C	Same as table 12.2.4.3.1-1
Proprietary Information	O	Same as table 12.3.4.3.1-1
Result-Code	M	Indicates whether a particular request was completed successfully or whether an error occurred
O_C	This is a parameter that, if provisioned by the operator to be present, shall be included in the CDRs when the required conditions are met. In other words, an O_C parameter that is configured to be present is a conditional parameter.	

Annex A (informative): Mapping of Requirements with CSFs

Table A-1 illustrates the mapping of the Requirements specified in ETSI TS 118 102 [i.1] with the CSFs specified in the present document.

Table A-1: Mapping of Requirements to CSFs

CSF Name	Supported Sub-Functions	Associated Requirements	Notes
Addressing and Identification (AID)	• Management of identifiers	OSR-026 OSR-023 OSR-024 OSR-025	Overlap w/: DIS for OSR-023, OSR-024, and OSR-025
Communication Management/Delivery Handling (CMDH)	• Providing communications with other CSEs, AEs, and NSEs • Communications management: best effort • Communications policy management • Underlying Network connectivity management • Communications management: data store and forward • Ability to trigger off-line device	OSR-001 OSR-002 OSR-005 OSR-006 OSR-008 OSR-009 OSR-012 OSR-013 OSR-014 OSR-015 OSR-018 OSR-019 OSR-021 OSR-027 OSR-032 OSR-035 OSR-038 OSR-039 OSR-040 OSR-048 OSR-049 OSR-050 OSR-053 OSR-062 OSR-063 OSR-064 OSR-065 OSR-066 OSR-067 OSR-068 CRPR-001 CRPR-002 CRPR-003 MGR-016	Overlap w/: DMR for OSR-001, OSR-009, OSR-021, OSR-032 SSM for OSR-009 LOC for OSR-006 GMG for OSR-006 NSSE for OSR-006, OSR-027 SSM for OSR-009
Data Management and Repository (DMR)	• Data storage and management • Semantic support • Data aggregation • Data analytics • Device data backup and recovery	OSR-001 OSR-007 OSR-009 OSR-016 OSR-020 OSR-021 OSR-032 OSR-034 OSR-036 OSR-058 SMR-006 SER-015	Overlap w/: CMDH for OSR-001, OSR-009, OSR-021, OSR-032 SUB for OSR-016 GMG for OSR-020

CSF Name	Supported Sub-Functions	Associated Requirements	Notes
Device Management (DMG)	• Configuration Management • Diagnostics and Monitoring • Firmware management • Software management • Device Area Network topology management	OSR-017 OSR-069 OSR-070 OSR-071 OPR-001 OPR-002 OPR-003 MGR-001 MGR-003 MGR-004 MGR-006 MGR-007 MGR-008 MGR-009 MGR-011 MGR-012 MGR-013 MGR-014 MGR-015 MGR-019 MGR-020 MGR-021 SER-013 SER-014	Overlaps w/: GMG for OSR-017 SEC for SER-013
Discovery (DIS)	• Discover resource • Local discovery (within CSE) • Directed remote discovery	OSR-023 OSR-024 OSR-025 OSR-059 OSR-060 OSR-061 MGR-002 SMR-004	Overlaps w/: AID for OSR-023, OSR-024, OSR-025
Group Management (GMG)	• Management of a group and its membership • CRUD • Use Underlying Network group capabilities • Bulk operations • Access control	OSR-006 OSR-017 OSR-020 OSR-029 OSR-030 OSR-031 OSR-037 OSR-047 MGR-005	Overlaps w/: CMDH for OSR-006 LOC for OSR-006 GMG for OSR-006 NSSE for OSR-006, OSR-037 DMR for OSR-020 DMG for OSR-017
Location (LOC)	• Location management • Network-provided • GPS-provided • Confidentiality enforcement as it relates to location	OSR-006 OSR-051 OSR-052 SER-026	
Network Service Exposure /Service execution and triggering (NSSE)	• Access Underlying Network service • Location • Device triggering • Small data • Policy and charging • Support multiple Underlying Network functions	OSR-006 OSR-011 OSR-027 OSR-037 OSR-054 OSR-055 OSR-056 MGR-017 MGR-018 OPR-004 OPR-005 OPR-006	Overlaps w/: CMDH for OSR-027 GMG for OSR-006, OSR-037 LOC for OSR-006
Registration (REG)	• CSE registration • Application registration • Device registration • ID correlation	MGR-010	Overlaps w/: SEC

CSF Name	Supported Sub-Functions	Associated Requirements	Notes
Security (SEC)	• Sensitive Data Handling • Secure storage • Secure execution • Independent environments • Security administration • Pre-provisioning • Dynamic bootstrap • Network bootstrap • Security association • Link level • Object level • Authorization and access • Identity protection	SER-001 SER-002 SER-003 SER-004 SER-005 SER-006 SER-007 SER-008 SER-009 SER-010 SER-011 SER-012 SER-013 SER-016 SER-017 SER-018 SER-019 SER-020 SER-021 SER-022 SER-023 SER-024 SER-025 MGR-010	Overlap w/: DMG for SER-013 REG for MGR-010 SSM for SER-007
Service Charging and Accounting (SCA)	• Charging enablers • Sending charging information to charging server • Subscription-based charging • Event-based charging • Session-based charging • Service-based charging • Correlation with Underlying Network • Charging management • Offline charging • Online charging	CHG-001, CHG-002a, CHG-002b, CHG-003, CHG-004, CHG-005	
Service Session Management (SSM)	• Service Session Management (CSE to CSE, AE to CSE, and AE to AE) • Session persistence over link outage • Session context handling • Assignment of session ID • Session routing • Multi-hop session management • Session policy management	OSR-003 OSR-004 OSR-009 OSR-045 SER-007	Overlap w/: CMDH and DMR for OSR-009 SEC for SER-007
Subscription/Notification Support (SUB)	• Subscribe (CSE, AE) • Local • Remote • Subscription to a group • Notification • Synchronous • Asynchronous	OSR-010 OSR-016 OSR-033	Overlaps w/: DMR for OSR-016

Annex B:
Void

Annex C (informative): Interworking between oneM2M System and 3GPP2 Underlying Networks

C.1 General Concepts

Interworking between oneM2M System and 3GPP2 Underlying Networks is based on 3GPP2 X.P0068 [i.17].

In order to provide M2M services, interworking between oneM2M System and the 3GPP2 Underlying Network is required. M2M Applications (AEs) in the M2M UEs (M2M Nodes such as the ASNs, MNs, and ADNs) and the M2M Applications in the external network (Infrastructure Domain) use services provided by the 3GPP2 Underlying Network, and optionally the services provided by an M2M Server (IN-CSE). The 3GPP2 Underlying Network provides transport and communication services, including 3GPP2 bearer services, IMS and SMS.

3GPP2 Underlying Network supports several interworking models, such as the following:

- Direct Model - Direct Communication provided by the 3GPP2 Network Operator:
  - The M2M Applications in the external network connect directly to the M2M Applications in the UEs used for M2M via the 3GPP2 Underlying Network without the use of any M2M Server.
- Indirect Model - M2M Service Provider controlled communication:
  - Uses an M2M Server that is an entity outside the 3GPP2 Underlying Network operator domain for enabling communications between the Applications in the external network and at the UEs used for M2M. Tsp interface or SMS interface is an external interface that the third party M2M Server supports with the entities that are within the 3GPP2 Underlying Network domain.
- Indirect Model - 3GPP2 Operator controlled communication:
  - Uses an M2M Server that is an entity inside the 3GPP2 Underlying Network operator domain for enabling communications between the Applications in the external network and at the UEs used for M2M. Tsp interface or SMS interface is an internal interface that the 3GPP2 Underlying Network operator controlled M2M Server supports with other entities within the 3GPP2 Underlying Network domain.
- Hybrid Model:
  - Direct and Indirect models are used simultaneously in the hybrid model i.e. performing Control Plane signalling using the Indirect Model and connecting the M2M Applications in the external network and at the UEs used for M2M over User Plane using the Direct Model.

C.2 M2M Communication Models

In the indirect and hybrid models, the deployment of an M2M Server (IN-CSE) may be inside or outside the 3GPP2 Underlying Network operator domain as illustrated in figures C.2-1 and C.2-2. When the M2M Server is part of the 3GPP2 Underlying Network operator domain (figures C.2-1(C) and C.2-2), the M2M Server is considered a 3GPP2 Underlying Network operator internal network function, is operator controlled, and may provide operator value-added services. In this case, security and privacy protection for communication between the M2M-IWF and the M2M Server (IN-CSE) is optional. When the M2M Server is deployed outside the 3GPP2 Underlying Network operator domain (figures C.2-1(B) and C.2-2), the M2M Server is M2M Service Provider controlled. In this case, security and privacy protection for communication between the M2M-IWF and the M2M Server (IN-CSE) is needed. In the direct model (figure C.2-1(A)), there is no external or internal M2M Server in the communication path.

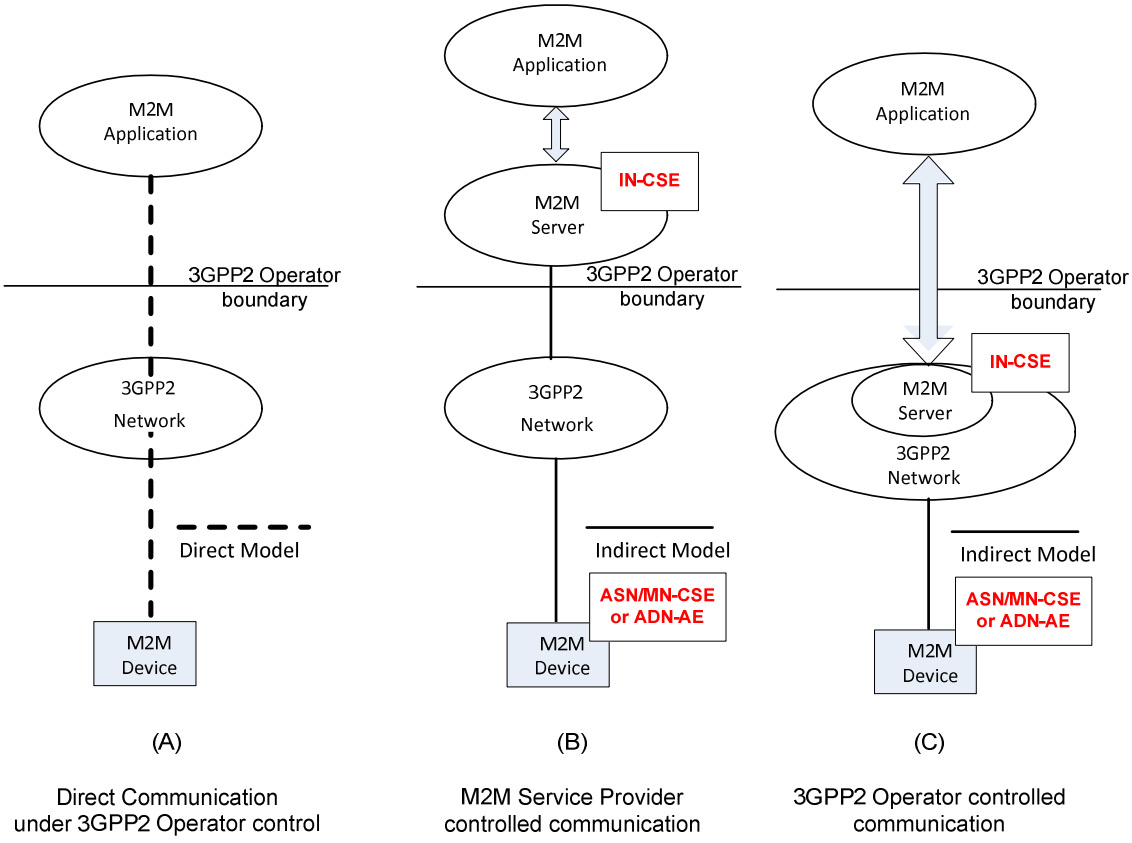


Figure C.2-1: M2M Communication Models

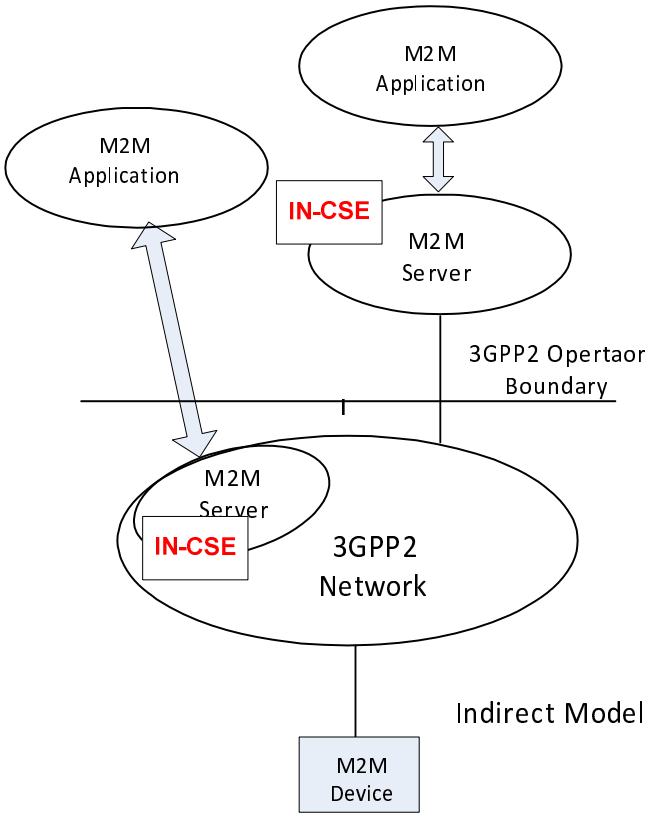


Figure C.2-2: Multiple M2M Applications Using Diverse Communication Models

A 3GPP2 network operator may deploy the hybrid model with a combination of no internal and external M2M Server (as in the Direct Model) and internal and/or external M2M Server (as in the Indirect Model). As shown in figure C.2-2, a UE (an M2M Node such as ASN/MN or ADN-AE) may be in communication with multiple M2M Servers which can be made up of a combination of 3GPP2 Underlying Network operator controlled and M2M Service Provider controlled M2M Servers. In that scenario, the M2M Service Provider controlled M2M Server, and the 3GPP2 Underlying Network operator controlled M2M Server may offer different capabilities to the M2M Applications.

Though not illustrated, it is also possible that in the Indirect Service Model with 3GPP2 network operator controlled M2M Server; the M2M Application may be inside the 3GPP2 network operator domain and under 3GPP network operator control.

C.3 3GPP2 Architectural Reference Model for M2M

Figure C.3-1 shows the architecture for a UE used for M2M connecting to the 3GPP2 Underlying Network. The architecture supports various architectural models described in clause C.2.

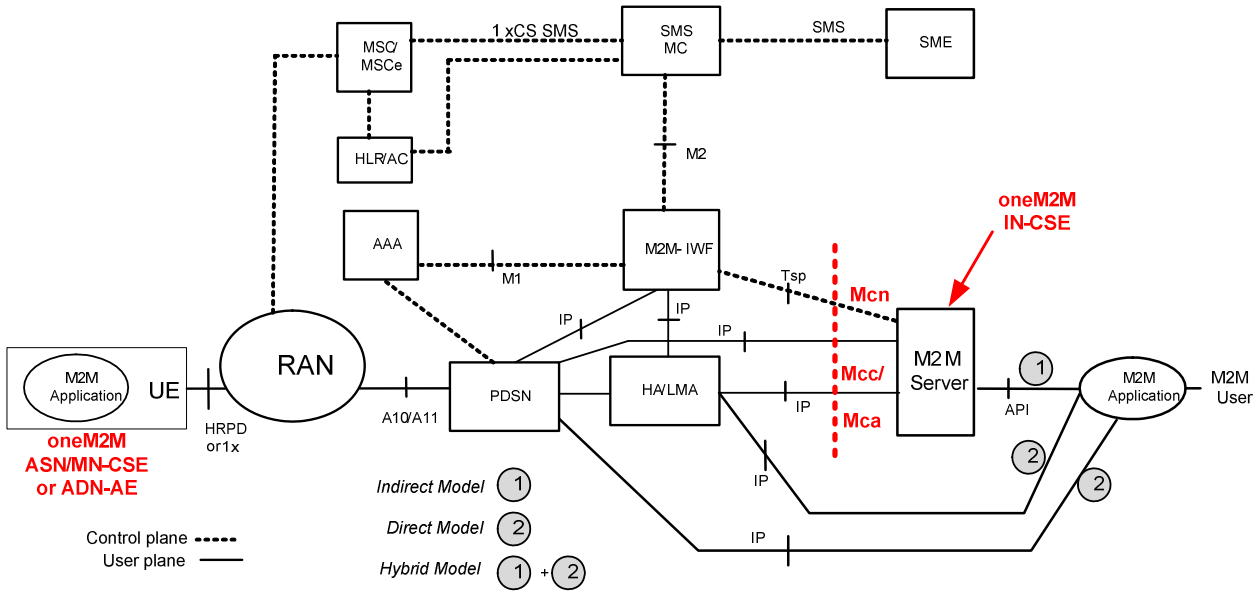


Figure C.3-1: Enhanced 3GPP2 Network Architecture for Supporting M2M

The M2M Server (IN-CSE) is the entity which connects to the 3GPP2 Underlying Network for providing communication with the UEs used for M2M. The M2M Server offers capabilities for use by one or multiple M2M Applications (AEs) hosted by the UE (ASN/MN or ADN-AE). The corresponding M2M Applications in the external network (Infrastructure Domain) are hosted by one or multiple M2M Application platform(s).

The M2M Server interfaces with the 3GPP2 Underlying Network entities located in the home domain of the UE used for M2M via the Tsp and IP interfaces. M2M Server encompasses the IN-CSE entity specified by oneM2M. M2M Server interfaces with the M2M-IWF via Tsp Interface for Control Plane communications. User plane interactions between the M2M Server and 3GPP2 Underlying Network entities such as the PDSN and/or HA/LMA is via native-IP. With this configuration, oneM2M reference points Mcn and Mcc map to 3GPP2 reference points Tsp and IP respectively.

C.4 Communication between oneM2M Service Layer and 3GPP2 Underlying Network

Communication between the M2M Server (IN-CSE) and the entities in the 3GPP2 Underlying Network make use of the User Plane and the Control Plane communication paths, as needed for different 3GPP2 M2M communication models. User Plane communication path uses IP transport between the M2M Server (IN-CSE) and the ADN-AE or the CSE in the UE used for M2M (ASN/MN-CSE). The User Plane maps to oneM2M Mcc reference point. Control Plane communication path is over Tsp interface and maps to oneM2M Mcn reference point.

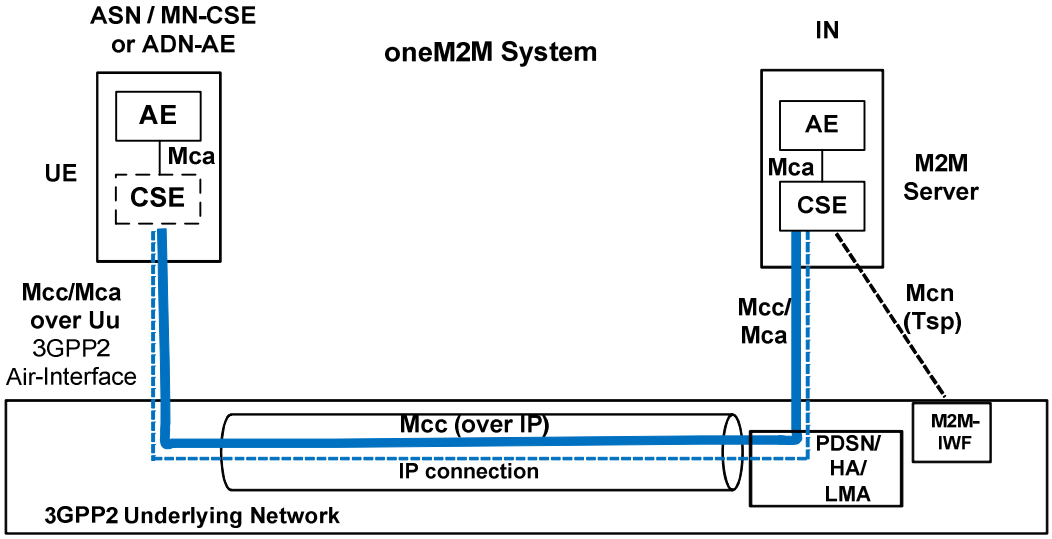


Figure C.4-1: User Plane and Control Plane Communication Paths

C.5 Information Flows

C.5.0 Overview

3GPP2 X.P0068 [i.17] specifies several system optimizations that can be used for M2M applications. Such optimizations include the following:

- Interaction of M2M Server with M2M-IWF for device triggering.
- Device trigger using SMS.
- Device trigger using broadcast SMS.
- Device trigger using IP transport.

C.5.1 Tsp Interface Call Flow

Figure C.5.1-1 is the high level call flow illustrating device triggering using Tsp interface.

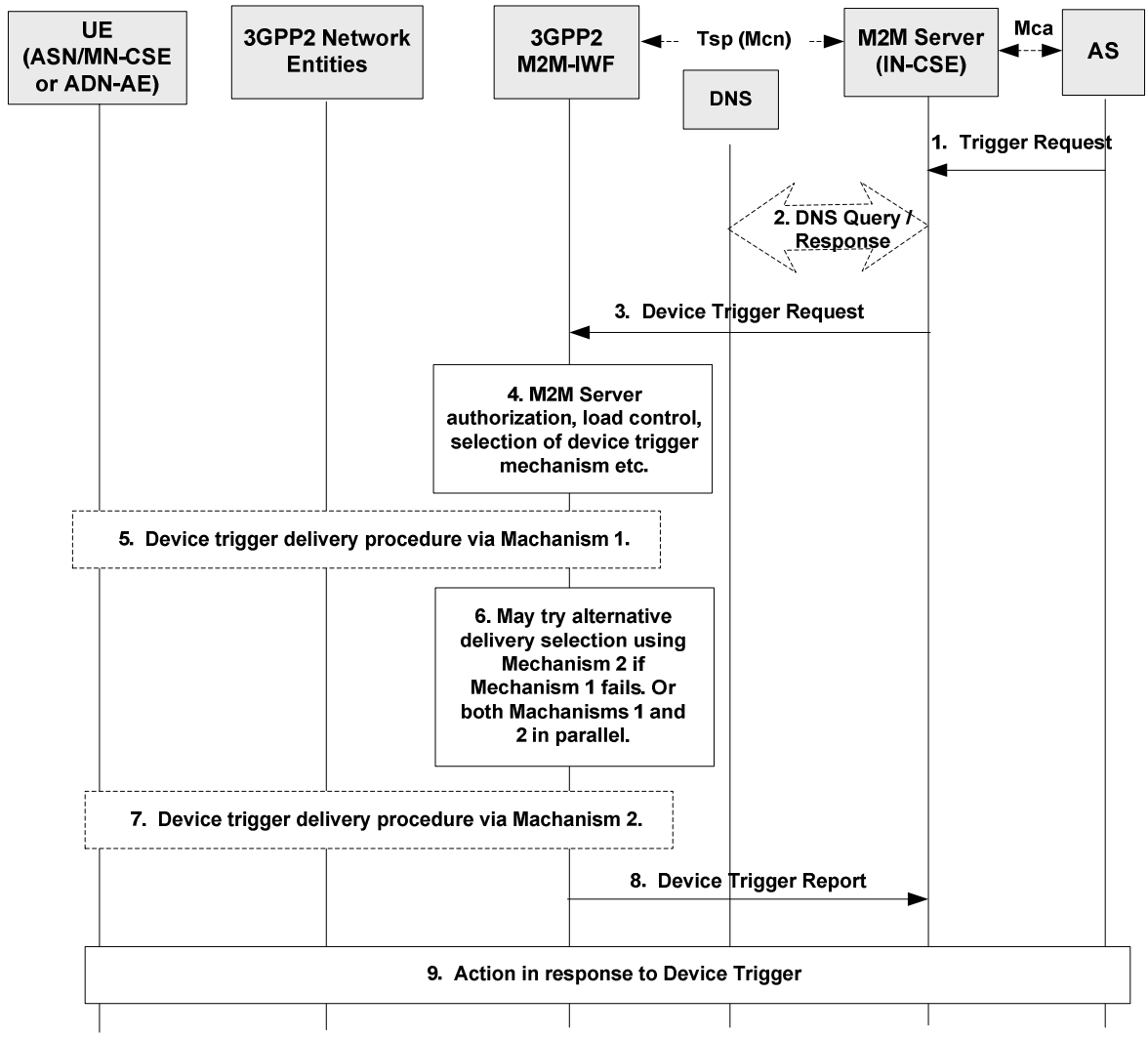


Figure C.5.1-1: Tsp Interface Call Flow

1) M2M Server (IN-CSE) receives a request from an M2M Application Server (AE in Infrastructure Domain) to deliver data to a UE used for M2M (ASN/MN-CSE or ADN-AE) located in the 3GPP2 Underlying Network. Knowing the CSE-ID or AE-ID of the destination M2M Node, IN-CSE deduces its 3GPP2 External Identifier.

2) M2M Server (IN-CSE) may perform DNS query to obtain the IP address of the M2M-IWF for reaching the destination M2M Node.

3) M2M Server sends Device Trigger Request message to the M2M-IWF that includes destination M2M Node External ID and other information.

4) M2M-IWF checks that the M2M Server is authorized to send trigger requests and performs other tasks such as verifying that the M2M Server has not exceeded its quota or rate of trigger submission over Tsp. If such checks fail, the MTC-IWF sends a Device Trigger Confirm message with a cause value indicating the reason for the failure condition and the call flow stops at this step.

Otherwise, the MTC-IWF continues to interact with HAAA/HLR for obtaining 3GPP2 Internal ID for the M2M Node and other information for reaching the M2M Node in the 3GPP2 Underlying Network. M2M-IWF also determines the device trigger mechanisms (e.g. Mechanism 1, Mechanism 2 etc.) supported by the M2M Node. The flow continues with step 5.

5) M2M-IWF decides to deliver device trigger using e.g. Mechanism 1 and performs appropriate 3GPP2 Underlying Network specific procedures.

6) M2M-IWF may try alternative device trigger delivery mechanism (e.g. Mechanism 2) if Mechanism 1 fails. Or both Mechanism 1 and 2 can be performed in parallel.

7) M2M-IWF performs appropriate 3GPP2 Underlying Network specific procedures for delivering device trigger using Mechanism 2.

8) M2M-IWF sends Device Trigger Report to the M2M Server upon receiving the acknowledgment from the M2M Node that it has received M2M device trigger.

9) The M2M Node and the M2M Server/AS take actions in response to the device trigger as needed.

C.5.2 Point to Point Device Triggering

3GPP2 Underlying Network supports the following point-to-point device triggering mechanisms:

- SMS on common channel.
- SMS on 1xCS traffic channel.
- Device trigger using IP interface.

Device trigger using IP interface assumes that PPP sessions has been established and maintained between the M2M Node and the PDSN. An IP address has been assigned to the M2M Node by the IP anchor (PDSN/HA/LMA) and is maintained by the M2M Node and by other entities (e.g. HAAA) in the 3GPP2 Underlying Network. Upon receiving device trigger from the M2M Server, the M2M-IWF obtains the IP address assigned to the M2M Node from the M2M-AAA/HAAA. After that, the M2M-IWF sends device trigger to the M2M Node through IP routing via IP interface to the HA/LMA for MIP and PMIP operation, or to the PDSN for Simple IP operation.

C.5.3 Broadcast Device Triggering

3GPP2 Underlying Network supports the following broadcast device triggering mechanisms:

- SMS broadcast.

Annex D (normative): <*mgmtObj*> Resource Instances Description

D.1 oneM2M Management Functions

This clause describes the management functions supported by oneM2M. These functions are fulfilled by defining specializations of <*mgmtObj*> resources. These specializations can be regarded as "sub-types" of the <*mgmtObj*> resource type with specific designing to support different management capabilities through operations defined by oneM2M. These specializations are service layer information models for the purpose of management. They can be used within the M2M service layer or they can be further mapped to existing management technologies such as OMA DM [i.3], OMA LWM2M [i.4] and BBF TR-069 [i.2] to enable the management of devices with OMA or BBF compliant management clients.

NOTE: The resources defined in this annex represent specializations of the <*mgmtObj*> resource as a result of introducing specializations of the *[objectAttribute]* attribute. The *mgmtDefinition* attribute carries the name of the resource type specialization. The names of instantiations of these resource specializations are not fixed.

D.2 Resource *firmware*

The *[firmware]* resource is used to share information regarding the firmware on the device. The *[firmware]* resource is a specialization of the <*mgmtObj*>resource.

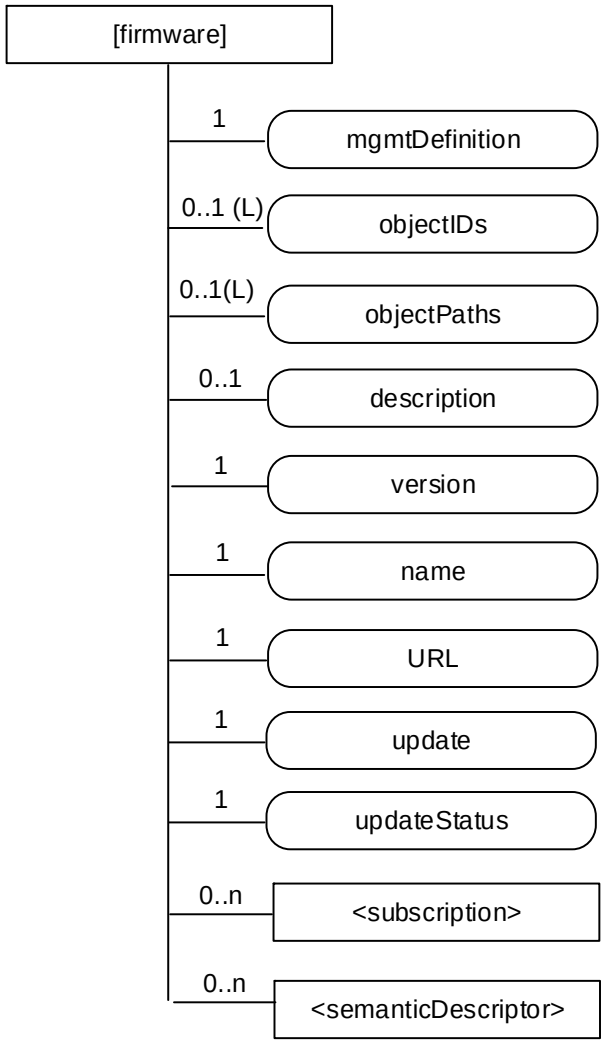


Figure D.2-1: Structure of *[firmware]* resource

The *[firmware]* resource shall contain the child resources specified in table D.2-1.

Table D.2-1: Child resources of *[firmware]* resource

Child Resources of *[firmware]*	Child Resource Type	Multiplicity	Description
[variable]	*<subscription>*	0..n	See clause 9.6.8 where the type of this resource is described.
[variable]	*<semanticDescriptor>*	0..n	See clause 9.6.30

The *[firmware]* resource shall contain the attributes specified in table D.2-2.

Table D.2-2: Attributes of *[firmware]* resource

Attributes of *[firmware]*	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
labels	0..1(L)	RW	See clause 9.6.1.3.
mgmtDefinition	1	WO	See clause 9.6.15. Has fixed value *"firmware"* to indicate the resource is for firmware management.
objectIDs	0..1 (L)	WO	See clause 9.6.15.
objectPaths	0..1 (L)	WO	See clause 9.6.15.
description	0..1	RW	See clause 9.6.15.
version	1	RW	The version of the firmware. This attribute is a specialization of *[objectAttribute]* attribute.
name	1	RW	The name of the firmware to be used on the device. This attribute is a specialization of *[objectAttribute]* attribute.
URL	1	RW	The URL from which the firmware image can be downloaded. This attribute is a specialization of *[objectAttribute]* attribute.
update	1	RW	The action that downloads and installs a new firmware in a single operation. The action is triggered by assigning value "TRUE" to this attribute. This attribute is a specialization of *[objectAttribute]* attribute.
updateStatus	1	RW	Indicates the status of the update. This attribute is a specialization of *[objectAttribute]* attribute.

D.3 Resource *software*

The *[software]* resource is used to share information regarding the software on the device. The *[software]* resource is a specialization of the <*mgmtObj*>resource.

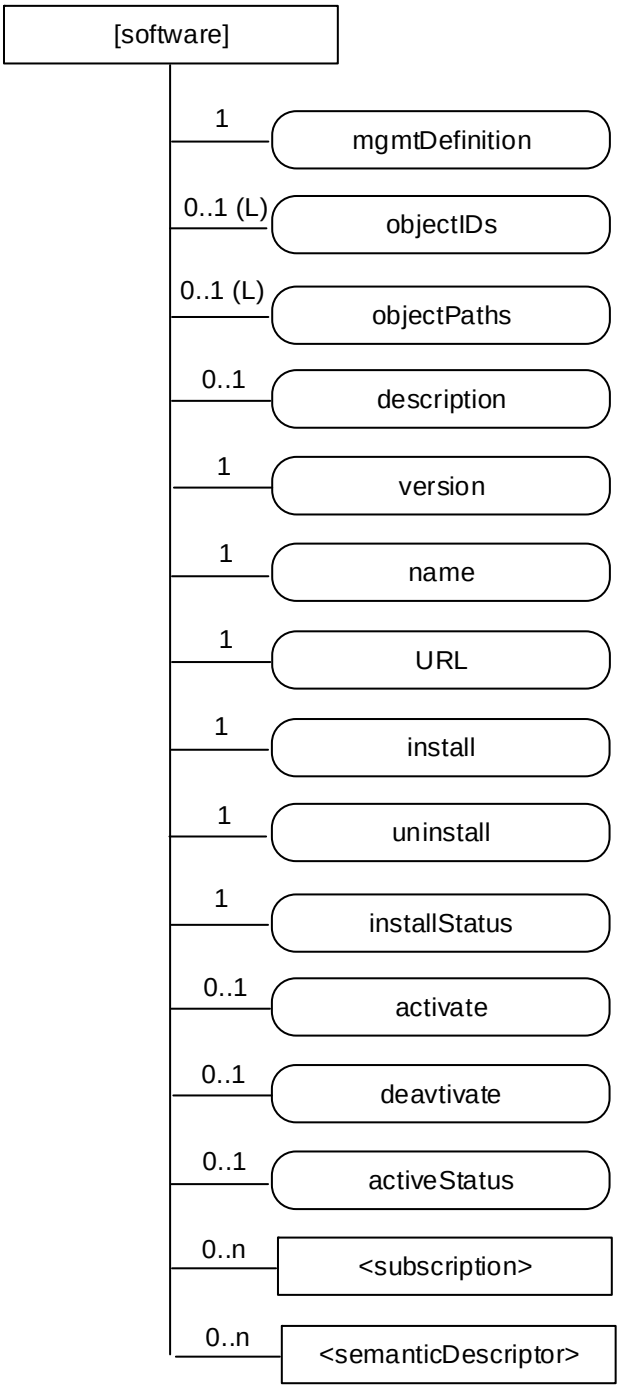


Figure D.3-1: Structure of *[software]* resource

The *[software]* resource shall contain the child resource specified in table D.3-1.

Table D.3-1: Child resources of *[software]* resource

Child Resources of *[software]*	Child Resource Type	Multiplicity	Description
[variable]	*<subscription>*	0..n	See clause 9.6.8 where the type of this resource is described.
[variable]	*<semanticDescriptor>*	0..n	See clause 9.6.30.

The *[software]* resource shall contain the attributes specified in table D.3-2.

Table D.3-2: Attributes of *[software]* resource

Attributes of *[software]*	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
labels	0..1(L)	RW	See clause 9.6.1.3.
mgmtDefinition	1	WO	See clause 9.6.15. Has fixed value *"software"* to indicate the resource is for software management.
objectIDs	0..1 (L)	WO	See clause 9.6.15.
objectPaths	0..1 (L)	WO	See clause 9.6.15.
description	0..1	RW	See clause 9.6.15.
version	1	RW	The version of the software. This attribute is a specialization of *[objectAttribute]* attribute.
name	1	RW	The name of the software to be used on the device. This attribute is a specialization of *[objectAttribute]* attribute.
URL	1	RW	The URL from which the software package can be downloaded. This attribute is a specialization of *[objectAttribute]* attribute.
install	1	RW	The action that downloads and installs new software in a single operation. The action is triggered by assigning value "TRUE" to this attribute. This attribute is a specialization of *[objectAttribute]* attribute.
uninstall	1	RW	The action that un-installs the software. The action is triggered by assigning value "TRUE" to this attribute. This attribute is a specialization of *[objectAttribute]* attribute.
installStatus	1	RW	Indicates the status of the install. This attribute is a specialization of *[objectAttribute]* attribute.
activate	0..1	RW	The action that activates software previously installed. The action is triggered by assigning value "TRUE" to this attribute. This attribute is a specialization of *[objectAttribute]* attribute.
deactivate	0..1	RW	The action that deactivates software. The action is triggered by assigning value "TRUE" to this attribute. This attribute is a specialization of *[objectAttribute]* attribute.
activeStatus	0..1	RW	The status of active or deactivate action. This attribute is a specialization of *[objectAttribute]* attribute.

The state machine for managing the software in oneM2M is shown in figure D.3-2.

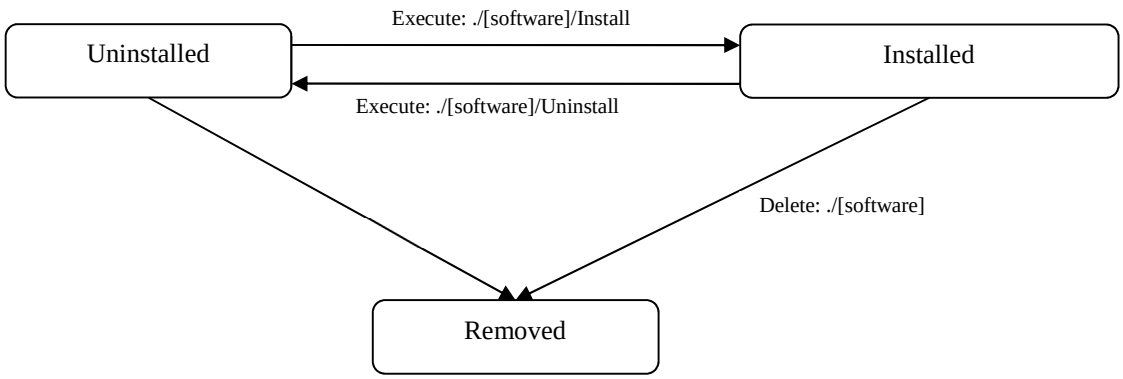


Figure D.3-2: State machine for *[software]* management

Figure D.3-3 is the state machine after install starts from the deactivated state.

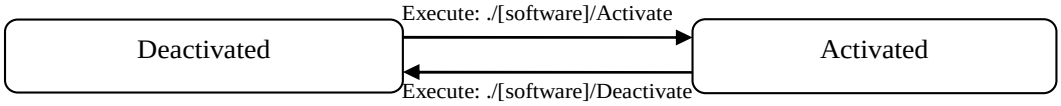


Figure D.3-3: State machine for *[software]* management after install

D.4 Resource *memory*

The *[memory]* resource is used to share information regarding the memory on the device. The *[memory]* resource is a specialization of the <*mgmtObj*>resource.

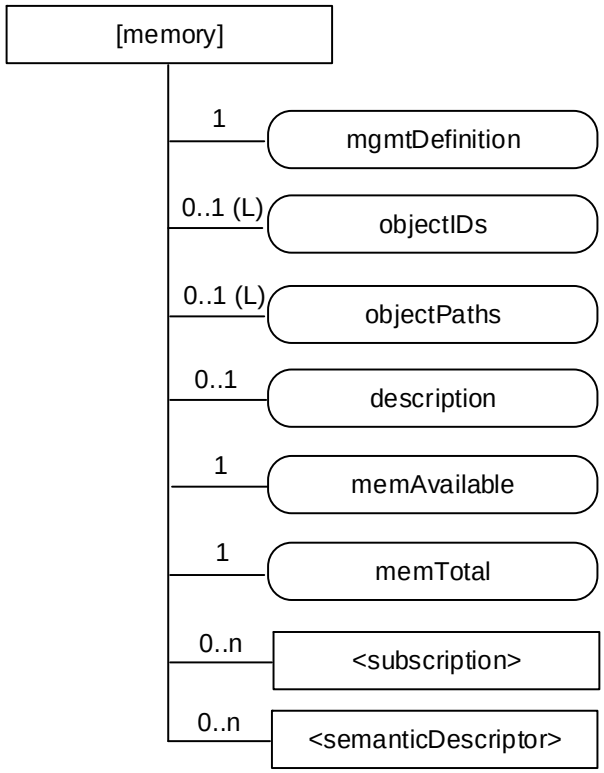


Figure D.4-1: Structure of *[memory]* resource

The *[memory]* resource shall contain the child resources specified in table D.4-1.

Table D.4-1: Child resources of *[memory]* resource

Child Resources of *[memory]*	Child Resource Type	Multiplicity	Description
[variable]	*<subscription>*	0..n	See clause 9.6.8 where the type of this resource is described.
[variable]	*<semanticDescriptor>*	0..n	See clause 9.6.30.

The *[memory]* resource shall contain the attributes specified in table D.4-2.

Table D.4-2: Attributes of *[memory]* resource

Attributes of *[memory]*	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
labels	0..1(L)	RW	See clause 9.6.1.3.
mgmtDefinition	1	WO	See clause 9.6.15. Has fixed value *"memory"* to indicate the resource is for memory management.
objectIDs	0..1 (L)	WO	See clause 9.6.15.
objectPaths	0..1 (L)	WO	See clause 9.6.15.
description	0..1	RW	See clause 9.6.15.
memAvailable	1	RW	The current available amount of memory. This attribute is a specialization of *[objectAttribute]* attribute.
memTotal	1	RW	The total amount of memory. This attribute is a specialization of *[objectAttribute]* attribute.

D.5 Resource *areaNwkInfo*

The *[areaNwkInfo]* resource is a specialization of the <*mgmtObj*>resource.

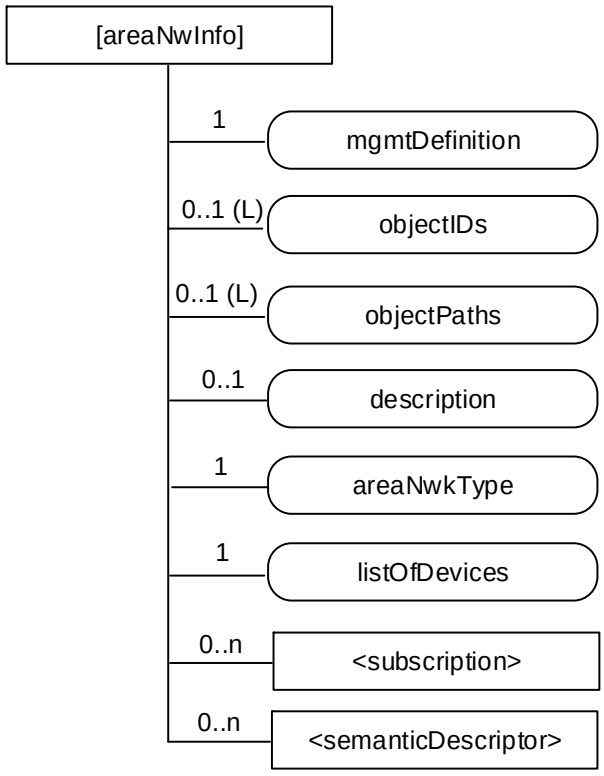


Figure D.5-1: Structure of *[areaNwkInfo]* resource

The *[areaNwkInfo]* resource shall contain the child resource specified in table D.5-1.

Table D.5-1: Child resources of *[areaNwInfo]* resource

Child Resources of *[areaNwkInfo]*	Child Resource Type	Multiplicity	Description
[variable]	*<subscription>*	0..n	See clause 9.6.8 where the type of this resource is described.
[variable]	*<semanticDescriptor>*	0..n	See clause 9.6.30.

The *[areaNwkInfo]* resource shall contain the attributes specified in table D.5-2.

Table D.5-2: Attributes of *[areaNwkInfo]* resource

Attributes of *[areaNwkInfo]*	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
labels	0..1(L)	RW	See clause 9.6.1.3.
mgmtDefinition	1	WO	See clause 9.6.15. Has fixed value *"areaNwkInfo"* to indicate the resource is for area network information.
objectIDs	0..1 (L)	WO	See clause 9.6.15.
objectPaths	0..1 (L)	WO	See clause 9.6.15.
description	0..1	RW	See clause 9.6.15.
areaNwkType	1	RW	The *areaNwkType* is a value that indicates the type of M2M Area Network. This attribute is a specialization of *[objectAttribute]* attribute.
listOfDevices	0..1 (L)	RW	Indicates the list of devices in the M2M Area Network. The attribute contains references to *[areaNwkDeviceInfo]* resource. From *listOfDevices*, the topology of the area network can be discovered and retrieved. This attribute is a specialization of *[objectAttribute]* attribute.

D.6 Resource areaNwkDeviceInfo

The *[areaNwkDeviceInfo]* resource is a specialization of the <*mgmtObj*>resource.

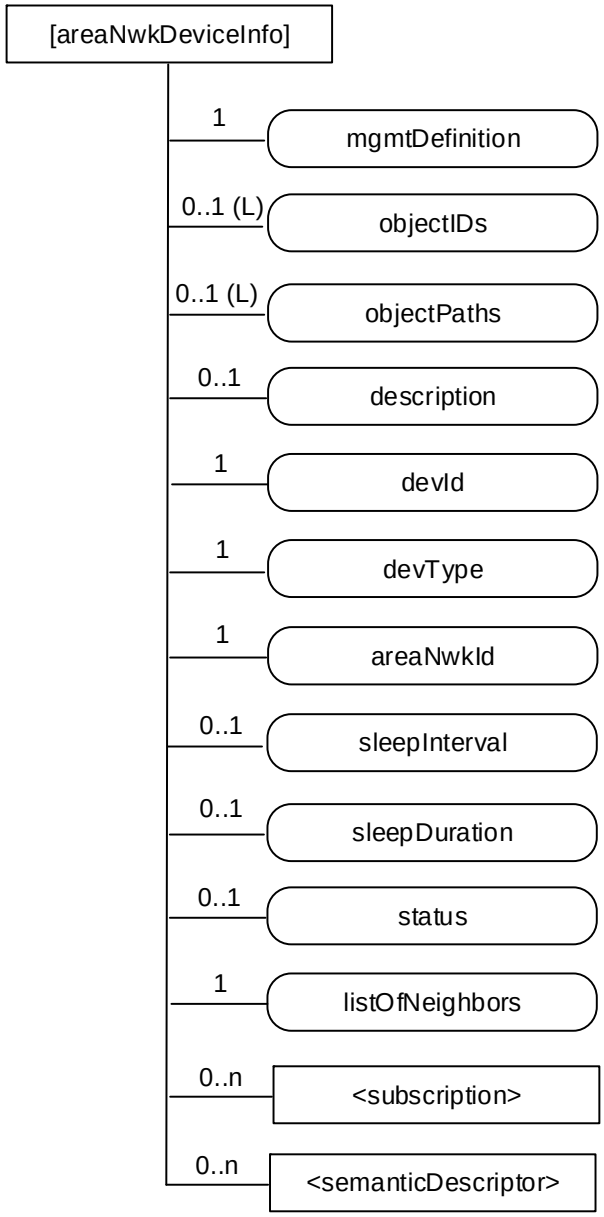


Figure D.6-1: Structure of *[areaNwkDeviceInfo]* resource

The *[areaNwkDeviceInfo]* resource shall contain the child resources specified in table D.6-1.

Table D.6-1: Child resources of *[areaNwkDeviceInfo]* resource

Child Resources of *[areaNwkDeviceInfo]*	Child Resource Type	Multiplicity	Description
[variable]	*<subscription>*	0..n	See clause 9.6.8 where the type of this resource is described.
[variable]	*<semanticDescriptor>*	0..n	See clause 9.6.30.

The *[areaNwkDeviceInfo]* resource shall contain the attributes specified in table D.6-2.

Table D.6-2: Attributes of *[areaNwkDeviceInfo]* resource

Attributes of *[areaNwkDeviceInfo]*	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
labels	0..1(L)	RW	See clause 9.6.1.3.
mgmtDefinition	1	WO	See clause 9.6.15. Has fixed value *"areaNwkDeviceInfo"* to indicate the resource is for area network device information.
objectIDs	0..1 (L)	WO	See clause 9.6.15.
objectPaths	0..1 (L)	WO	See clause 9.6.15.
description	0..1	RW	See clause 9.6.15.
devId	1	RW	Indicates the id of the device. It could be the id of the hardware or *nodeId*. This attribute is a specialization of *[objectAttribute]* attribute.
devType	1	RW	Indicates the type of the device. The attribute also indicates the functions or services that are provided by the device. Examples include temperature sensor, actuator, Zigbee coordinator or Zigbee router. This attribute is a specialization of *[objectAttribute]* attribute.
areaNwkId	1	RW	The reference to an *areaNwkInfo* resource which this device associates with. This attribute is a specialization of *[objectAttribute]* attribute.
sleepInterval	0..1	RW	The interval between two sleeps. This attribute is a specialization of *[objectAttribute]* attribute.
sleepDuration	0..1	RW	The time duration of each sleep. This attribute is a specialization of *[objectAttribute]* attribute.
status	0..1	RW	The status of the device (sleeping or waked up).
listOfNeighbors	0..1 (L)	RW	Indicates the neighbour devices of the same area network. When modified, the connection relationship of the devices shall be modified accordingly. This attribute is a specialization of *[objectAttribute]* attribute.

D.7 Resource *battery*

The *[battery]* resource is used to share information regarding the battery. The *[battery]* resource is a specialization of the <*mgmtObj*> resource.

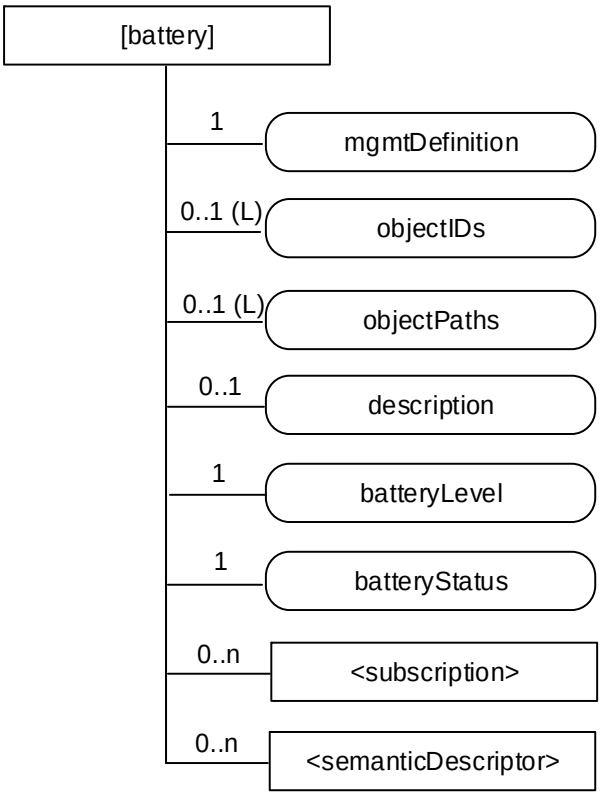


Figure D.7-1: Structure of *[battery]* resource

The *[battery]* resource shall contain the child resources specified in table D.7-1.

Table D.7-1: Child resources of *[battery]* resource

Child Resources of *[battery]*	Child Resource Type	Multiplicity	Description
[variable]	<*subscription*>	0..n	See clause 9.6.8 where the type of this resource is described.
[variable]	<*semanticDescriptor*>	0..n	See clause 9.6.30.

The *[battery]* resource shall contain the attributes specified in table D.7-2.

Table D.7-2: Attributes of *[battery]* resource

Attributes of *[battery]*	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
labels	0..1(L)	RW	See clause 9.6.1.3.
mgmtDefinition	1	WO	See clause 9.6.15. This attribute shall have the fixed value *"battery"*.
objectIDs	0..1 (L)	WO	See clause 9.6.15.
objectPaths	0..1 (L)	WO	See clause 9.6.15.
description	0..1	RW	See clause 9.6.15.
batteryLevel	1	RW	The current battery level. This attribute is a specialization of *[objectAttribute]* attribute.
batteryStatus	1	RW	Indicates the status of the battery. This attribute is a specialization of *[objectAttribute]* attribute.

D.8 Resource *deviceInfo*

The *[deviceInfo]* resource is used to share information regarding the device. The *[deviceInfo]* resource is a specialization of the <*mgmtObj*> resource.

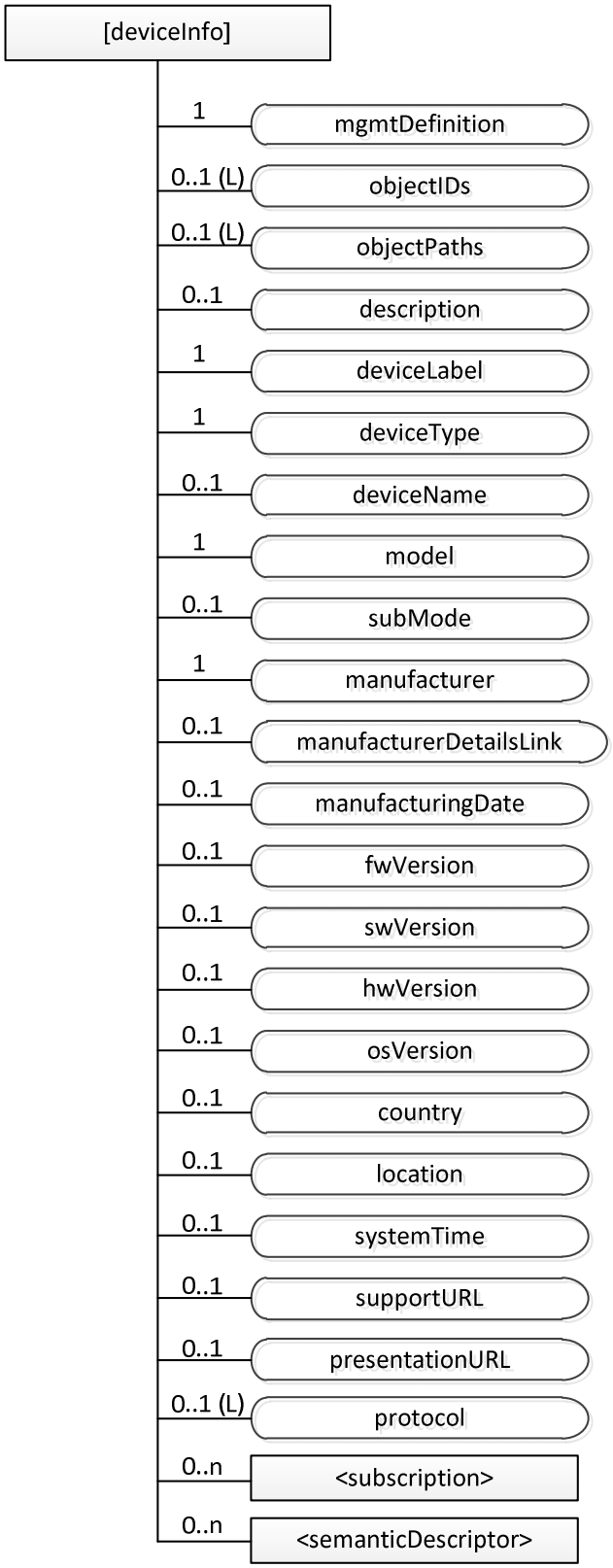


Figure D.8-1: Structure of *[deviceInfo]* resource

The *[deviceInfo]* resource shall contain the child resources specified in table D.8-1.

Table D.8-1: Child resources of *[deviceInfo]* resource

Child Resources of *[deviceInfo]*	Child Resource Type	Multiplicity	Description
[variable]	*<subscription>*	0..n	See clause 9.6.8 where the type of this resource is described.
[variable]	*<semanticDescriptor>*	0..n	See clause 9.6.30.

The *[deviceInfo]* resource shall contain the attributes specified in table D.8-2.

Table D.8-2: Attributes of *[deviceInfo]* resource

Attributes of *[deviceInfo]*	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
labels	0..1(L)	RW	See clause 9.6.1.3.
mgmtDefinition	1	WO	See clause 9.6.15. This attribute shall have the fixed value *"deviceInfo"*.
objectIDs	0..1 (L)	WO	See clause 9.6.15.
objectPaths	0..1 (L)	WO	See clause 9.6.15.
description	0..1	RW	See clause 9.6.15.
deviceLabel	1	RW	Unique device label assigned by the manufacturer. This attribute is a specialization of *[objectAttribute]* attribute. The value of the attribute typically exposes the device's serial number that is specific to a manufacturer and possibly further restricted within the manufacturer by a *deviceType* or model. This attribute shall be formatted as either single value-only string or a string format that contains a list of key-value pairs. When this attribute contains a list of key-value pairs, the list of key-value pairs is identified by a "|"(vertical line) character as the first character in the string. Within the list of key-value pairs, each key and value shall be separated by ":"(colon) and each pairs shall be separated by "(SPACE(U+0020))". An example for the key-value string about OMA DWAPI is "|systemID:0123 serviceID:xyz". When using reserved characters (e.g. SPACE, ":", "%", or "|") within a key or value element, the reserved characters are escaped by identifying the ascii value of the character with a percent escape character preceding the ascii value. For example if the previous examples systemID key's value included a SPACE character the string is represented as "|systemID:01%2023 serviceID:xyz". It is also possible to use a list of URNs.
manufacturer	1	WO	The name/identifier of the device manufacturer. This attribute is a specialization of *[objectAttribute]* attribute.
manufacturerDetailsLink	0..1	RW	URL to manufacturer's website. This attribute is a specialization of *[objectAttribute]* attribute.
manufacturingDate	0..1	WO	Manufacturing date of device. This attribute is a specialization of *[objectAttribute]* attribute.
model	1	WO	The name/identifier of the device mode assigned by the manufacturer. This attribute is a specialization of *[objectAttribute]* attribute.
subModel	0..1	WO	Device sub-model name. This attribute is a specialization of *[objectAttribute]* attribute.

Attributes of *[deviceInfo]*	Multiplicity	RW/ RO/ WO	Description
deviceType	1	RW	The type (e.g. cell phone, photo frame, smart meter) or product class (e.g. X-series) of the device. This attribute is a specialization of *[objectAttribute]* attribute.
deviceName	0..1	RW	Device name. This attribute is a specialization of *[objectAttribute]* attribute.
fwVersion	0..1	RW	The firmware version of the device (see note).
swVersion	0..1	RW	The software version of the device. This attribute is a specialization of *[objectAttribute]* attribute.
hwVersion	0..1	WO	The hardware version of the device. This attribute is a specialization of *[objectAttribute]* attribute.
osVersion	0..1	RW	Version of the operating system (defined by manufacturer). This attribute is a specialization of *[objectAttribute]* attribute.
country	0..1	WO	Country code of the device. It could be manufacturing country, deployment country or procurement country. This attribute is a specialization of [objectAttribute] attribute.
location	0..1	RW	Location where the device is installed. It may be configured via the user interface provided by the 'presentationURL' property or any other means. This attribute is a specialization of [objectAttribute] attribute.
systemTime	0..1	RW	Reference time for the device. This attribute is a specialization of [objectAttribute] attribute.
supportURL	0..1	RW	URL that points to product support information of the device. This attribute is a specialization of [objectAttribute] attribute.
presentationURL	0..1	RW	To quote UPnP: "the control point can retrieve a page from this URL, load the page into a web browser, and depending on the capabilities of the page, allow a user to control the device and/or view device status. The degree to which each of these can be accomplished depends on the specific capabilities of the presentation page and device". This attribute is a specialization of [objectAttribute] attribute.
protocol	0..1(L)	RW	A list of MIME types for all supported communication protocol(s) of the device. EXAMPLE: application/x-alljoin;version=1.0 application/x-echonet-lite;version=1.0 indicates the device supports both AllJoyn v1.0 and Echonet Lite v1.0. This attribute is a specialization of [objectAttribute] attribute.
NOTE: If the device only supports one kind of Software this is identical to *swVersion*. This attribute is a specialization of *[objectAttribute]* attribute.			

D.9 Resource deviceCapability

The *[deviceCapability]* resource represents each device's capability. The *[deviceCapability]* resource is a specialization of the <*mgmtObj*> resource.

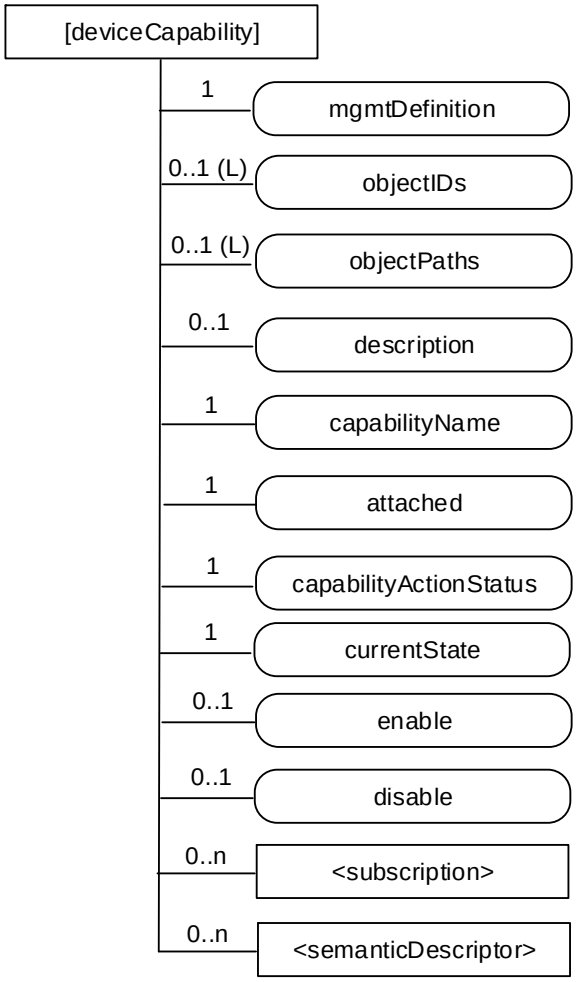


Figure D.9-1: Structure of *[deviceCapability]* resource

The *[deviceCapability]* resource shall contain the child resources specified in table D.9-1.

Table D.9-1: Child resources of *[deviceCapability]* resource

Child Resources of *[deviceCapability]*	Child Resource Type	Multiplicity	Description
[variable]	<*subscription*>	0..n	See clause 9.6.8 where the type of this resource is described.
[variable]	<*semanticDescriptor*>	0..n	See clause 9.6.30.

The *[deviceCapability]* resource shall contain the attributes specified in table D.9-2.

Table D.9-2: Attributes of *[deviceCapability]* resource

Attributes of *[deviceCapability]*	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
labels	0..1(L)	RW	See clause 9.6.1.3.
mgmtDefinition	1	WO	See clause 9.6.15. This attribute shall have the fixed value *"deviceCapability"*.
objectIDs	0..1 (L)	WO	See clause 9.6.15.
objectPaths	0..1 (L)	WO	See clause 9.6.15.
description	0..1	RW	See clause 9.6.15.
capabilityName	1	WO	The name of the capability. This attribute is a specialization of *[objectAttribute]* attribute.
attached	1	RO	Indicates whether the capability is attached to the device or not. This attribute is a specialization of *[objectAttribute]* attribute.
capabilityActionStatus	1	RO	Indicates the status of the Action (including a performed action and the corresponding final state). This attribute is a specialization of *[objectAttribute]* attribute.
currentState	1	RO	Indicates the current state of the capability (e.g. enabled or disabled). This attribute is a specialization of *[objectAttribute]* attribute.
enable	0..1	WO	The action that allows enabling the device capability. This attribute is a specialization of *[objectAttribute]* attribute.
disable	0..1	WO	The action that allows disabling the device capability. This attribute is a specialization of *[objectAttribute]* attribute.

D.10 Resource *reboot*

The *[reboot]* resource is used to reboot a device. The *[reboot]* resource is a specialization of the <*mgmtObj*> resource.

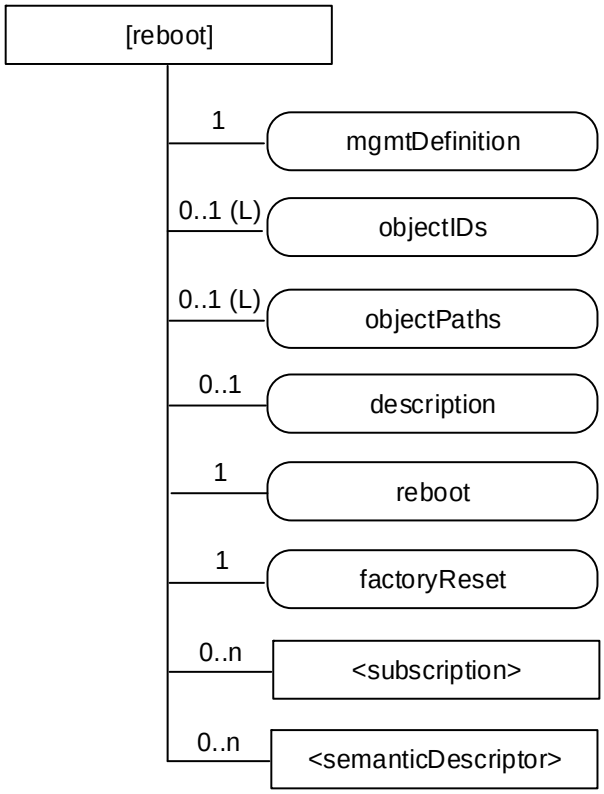


Figure D.10-1: Structure of *[reboot]* resource

The *[reboot]* resource shall contain the child resources specified in table D.10-1.

Table D.10-1: Child resources of *[reboot]* resource

Child Resources of *[reboot]*	Child Resource Type	Multiplicity	Description
[variable]	*<subscription>*	0..n	See clause 9.6.8 where the type of this resource is described.
[variable]	*<semanticDescriptor>*	0..n	See clause 9.6.30.

The *[reboot]* resource shall contain the attributes specified in table D.10-2.

Table D.10-2: Attributes of *[reboot]* resource

Attributes of *[reboot]*	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
labels	0..1(L)	RW	See clause 9.6.1.3.
mgmtDefinition	1	WO	See clause 9.6.15. This attribute shall have the fixed value "reboot".
objectIDs	0..1 (L)	WO	See clause 9.6.15.
objectPaths	0..1 (L)	WO	See clause 9.6.15.
description	0..1	RW	See clause 9.6.15.
reboot	1	RW	The action that allows rebooting the device. The action is triggered by assigning value "TRUE" to this attribute. This attribute is a specialization of *[objectAttribute]* attribute.
factoryReset	1	RW	The action that allows making the device returning to the factory settings. The action is triggered by assigning value "TRUE" to this attribute. This attribute is a specialization of *[objectAttribute]* attribute.

D.11 Resource *eventLog*

The *[eventLog]* resource is used to record the event log for a device. The *[eventLog]* resource is a specialization of the <*mgmtObj*> resource.

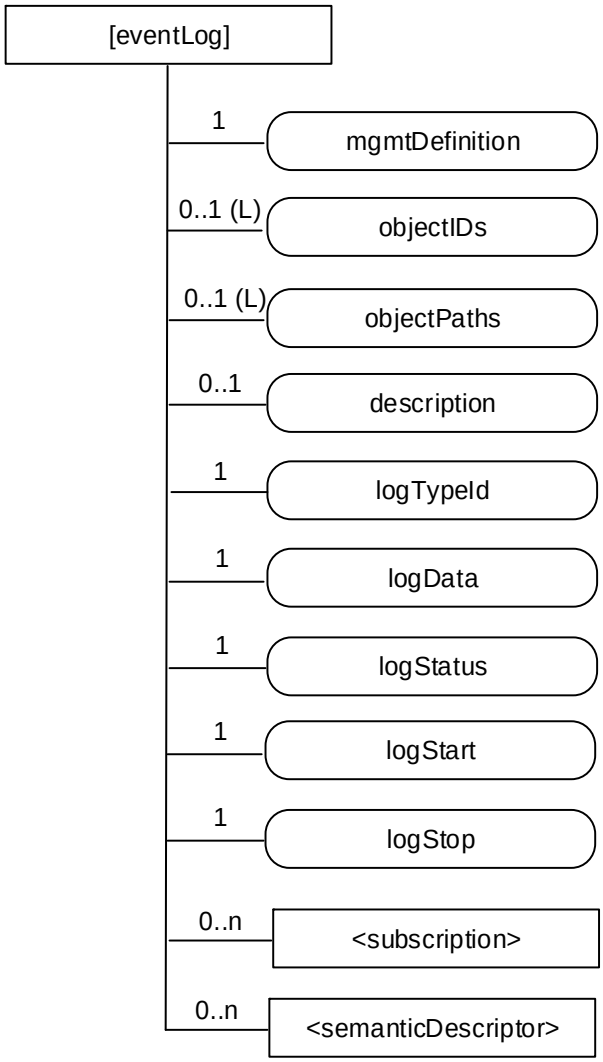


Figure D.11-1: Structure of *[eventLog]* resource

The *[eventLog]* resource shall contain the child resources specified in table D.11-1.

Table D.11-1: Child resources of *[eventLog]* resource

Child Resources of *[eventLog]*	Child Resource Type	Multiplicity	Description
[variable]	<*subscription*>	0..n	See clause 9.6.8 where the type of this resource is described.
[variable]	<*semanticDescriptor*>	0..n	See clause 9.6.30.

The *[eventLog]* resource shall contain the attributes specified in table D.11-2.

Table D.11-2: Attributes of *[eventLog]* resource

Attributes of *[eventLog]*	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
labels	0..1(L)	RW	See clause 9.6.1.3.
mgmtDefinition	1	WO	See clause 9.6.15. This attribute shall have the fixed value *"eventLog"*.
objectIDs	0..1 (L)	WO	See clause 9.6.15.
objectPaths	0..1 (L)	WO	See clause 9.6.15.
description	0..1	RW	See clause 9.6.15.
logTypeId	1	RW	Identifies the types of log to be recorded. E.g. security log, system log. This attribute is a specialization of *[objectAttribute]* attribute.
logData	1	RW	Diagnostic data logged upon event of interests defined by this diagnostic function. This attribute is a specialization of *[objectAttribute]* attribute.
logStatus	1	RW	Indicates the status of the logging process. E.g. Started, Stopped. This attribute is a specialization of *[objectAttribute]* attribute.
logStart	1	RW	The action that allows starting the log corresponding to the mentioned *logTypeId*. The action is triggered by assigning value "TRUE" to this attribute. This attribute is a specialization of *[objectAttribute]* attribute.
logStop	1	RW	The action that allows stopping the log corresponding to the mentioned *logTypeId*. The action is triggered by assigning value "TRUE" to this attribute. This attribute is a specialization of *[objectAttribute]* attribute.

D.12 Resource *cmdhPolicy*

D.12.0 Overview

A *[cmdhPolicy]* resource is defined as a specialization of the <*mgmtObj*> resource type as specified in clause 9.6.15. It includes a number of child resources which are referenced by means of *mgmtLink* attributes. Each of these linked child resources represents itself a specialization of the <*mgmtObj*> resource type. These child resources and their child resources are defined in clauses D.12.1 to D.12.8.

The *[cmdhPolicy]* resource represents a set of rules associated with a node hosting a specific CSE or a specific ADN that govern the behaviour of that CSE communication behaviour of that node regarding rejecting, buffering and sending request or response messages via the Mcc or Mca reference point between a CSE and its Registrar - in case of a node hosting a CSE - or via the Mca reference point - in case of an ADN.

The reference point between an associated CSE and its Registrar CSE - in case the *[cmdhPolicy]* resource is associated with a node hosting a CSE - or the Mca reference point used by the associated ADN - in case the *[cmdhPolicy]* resource is associated with an ADN - is termed "associated reference point" in what follows.

The rules contained in a *[cmdhPolicy]* resource are sub-divided into rules represented by different linked child resources with different purposes as follows:

- **Defaults:** Defines which CMDH related parameters will be used by default when a request or response message issued by a registrar of the associated CSE or the associated CSE itself to be sent across the associated reference point contains the ***Event Category*** parameter but not all other CMDH related parameters and which default ***Event Category*** parameter shall be used when none is given in the request or response.

- **Limits:** Defines the allowed limits for CMDH related parameters in request or response messages to be sent across the associated reference point with a given ***Event Category*** value.

- **Network usage:** Defines the conditions when usage of specific Underlying Networks is allowed for request or response messages to be sent across the associated reference point with a given ***Event Category*** value.

- **Buffering:** Defines limits of supported buffer size to be used for storing pending messages with a given ***Event Category*** value and their priorities when deletion cannot be avoided. Buffering of messages to be sent across an associated reference point form an ADN supporting CMDH to other nodes is optional.

The relationships of *[cmdhPolicy]* resources with other resources and the position within the overall resource structure are depicted in figure D.12.0-1. One or several *[cmdhPolicy]* resources can be assigned as child resources under a parent of <*node*> resource type. The <*node*> resource carrying CMDH policies is linked by means of a *nodeLink* attribute from an instance of a <*remoteCSE*> resource type or an instance of an <*AE*>resource type representing an Application Entity on an ADN. This *nodeLink* attribute as well as the reverse *hostedCSELink* or *hostedAELinks* attribute in the <*node*> resource define to which AE(s) or node the set of CMDH policies apply whenever this CSE receives requests or response messages that need to be forwarded over Mcc reference point sent across the associated reference point to or from the indicated entities. Since only one particular set of CMDH rules can be active for a given node at any given point in time, an *[activeCMDHPolicy]* child resource for CMDH policies shall be applied is used to point to the active *[cmdhPolicy]* resource that shall be effective for that particular node.

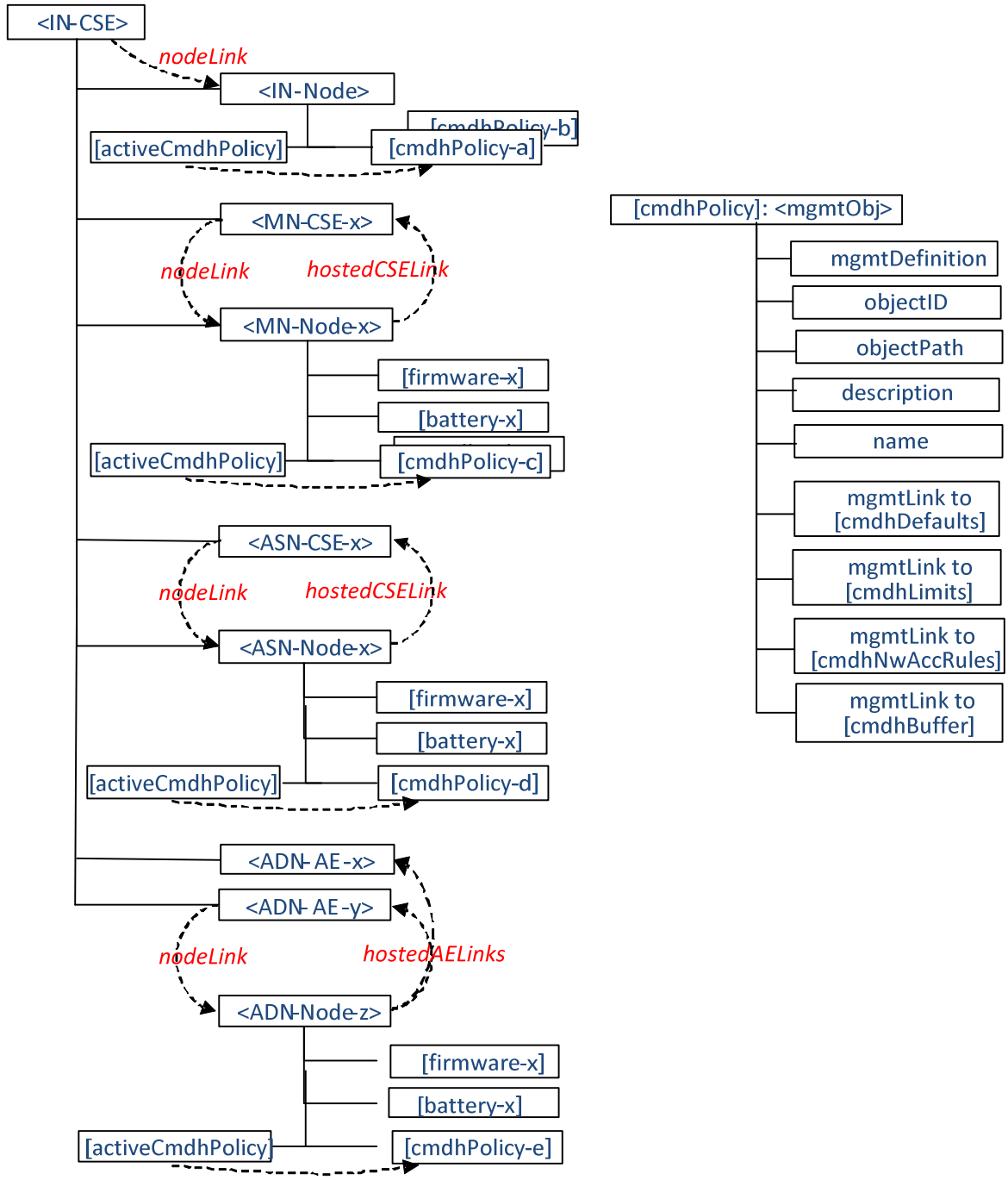


Figure D.12.0-1: Relationships between *[cmdhPolicy]* resource and other resources

When employing external management technology, the *[cmdhPolicy]* resources are assigned under instances of the <*node*> resources that represent the remotely managed field nodes in the IN-CSE performing device management for these nodes. In this scenario, the *[cmdhPolicy]* resources are transferred to the field node by means of the external device management technology applicable for that specific node.

When a field domain node is managed via the Mcc reference point, the *[cmdhPolicy]* resources are provisioned directly to instances of the <*node*> resources in the respective field domain CSE from an IN-CSE responsible for the device/entity management.

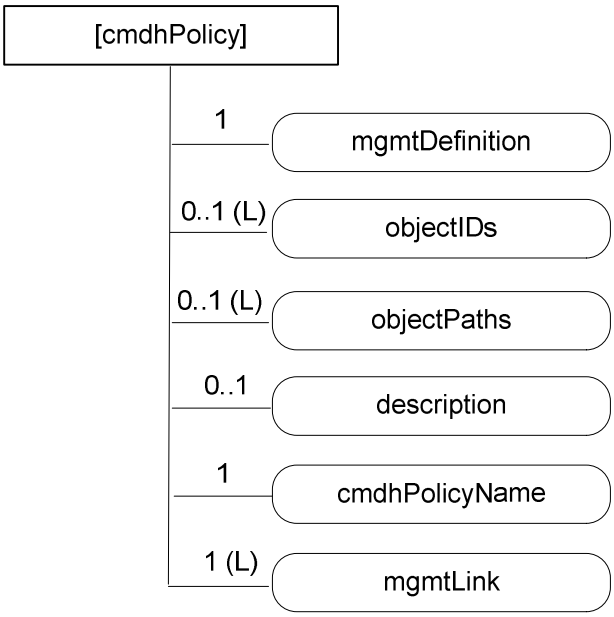


Figure D.12.0-2: Structure of *[cmdhPolicy]* resource

The *[cmdhPolicy]* resource shall contain attributes specified in table D.12.0-1.

Table D.12.0-1: Attributes of *[cmdhPolicy]* resource

Attributes of *[cmdhPolicy]*	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
labels	0..1(L)	RO	See clause 9.6.1.3.
mgmtDefinition	1	WO	See clause 9.6.15. Has fixed value *"cmdhPolicy"* to indicate the resource is for CMDH policy management.
objectIDs	0..1 (L)	WO	See clause 9.6.15.
objectPaths	0..1 (L)	WO	See clause 9.6.15.
description	0..1	RW	See clause 9.6.15.
cmdhPolicyName	1	RW	A name under which the CMDH policy will be referred. This attribute is a specialization of *[objectAttribute]* attribute.
mgmtLink	1 (L)	RW	A list containing at least 4 links: • 1 link to *[cmdhDefaults]* resource. • At least 1 or more link(s) to *[cmdhLimits]* resource(s). • At least 1 or more link(s) to *[cmdhNetworkAccessRules]* resource(s). • At least 1 or more link(s) to *[cmdhBuffer]* resource(s).

D.12.1 Resource *activeCmdhPolicy*

A managed node can have one or more sets of *[cmdhPolicy]* resources assigned as children.

The *[activeCmdhPolicy]* resource is used to provide a link to the currently active set of CMDH policies. This identifies which set of CMDH policies is currently actively in use in the corresponding CSE node. It allows the device management technology to activate a policy set independently of the download of a new set of CMDH policies in order to avoid potential race conditions. The *[activeCmdhPolicy]* and *[cmdhPolicy]* resources are children of the same <*node*> resource to which these policies apply.

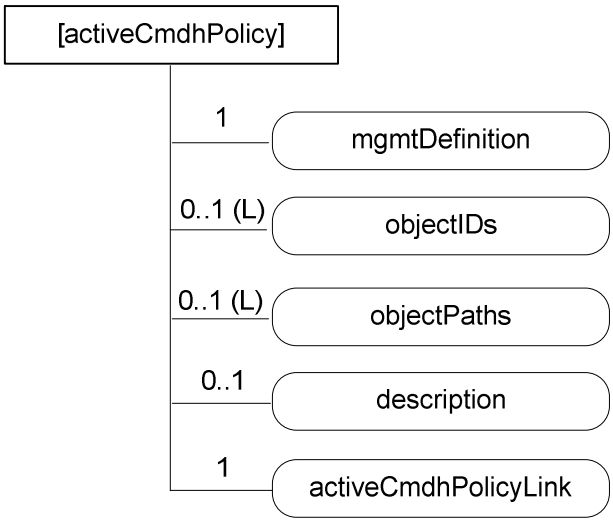


Figure D.12.1-1: Structure of *[activeCmdhPolicy]* resource

The *[activeCmdhPolicy]* resource shall contain attributes specified in table D.12.1-1.

Table D.12.1-1: Attributes of *[activeCmdhPolicy]* resource

Attributes of *[activeCmdhPolicy]*	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
labels	0..1(L)	RO	See clause 9.6.1.3
mgmtDefinition	1	WO	See clause 9.6.15. Has fixed value *"activeCmdhPolicy".*
objectIDs	0..1 (L)	WO	See clause 9.6.15.
objectPaths	0..1 (L)	WO	See clause 9.6.15.
description	0..1	RW	See clause 9.6.15.
activeCmdhPolicyLink	1	RW	The resource ID of the *[cmdhPolicy]* resource instance containing the CMDH policies that are currently active for the associated node that is represented by the parent *<node>* resource.

D.12.2 Resource *cmdhDefaults*

The *[cmdhDefaults]* resource is used to define default values that shall be used for CMDH-related parameters when request or response messages issued by Originators (registered AEs or functions inside the CSE itself) need to be sent across the associated reference point do not contain a value for the parameters ***Event Category***, ***Request Expiration Timestamp***, ***Result Expiration Timestamp***, ***Operation Execution Time***, ***Result Persistence***, and/or ***Delivery Aggregation***.

Upon receiving When a request or response message needs to be sent across the associated reference point, the entity performing the CMDH processing shall first look if the ***Event Category*** parameter is set. If not, the CSE shall use the *[cmdhDefEcValue]* resources (see below) to determine a value that should be used for ***Event Category***.

Then, if any of the parameters ***Request Expiration Timestamp***, ***Result Expiration Timestamp***, ***Operation Execution Time***, ***Result Persistence*** or ***Delivery Aggregation*** is not set, the entity performing the CMDH processing shall use the *[cmdhEcDefParamValues]* resources (see below) to populate the missing parameters supported by the respective message type (and only the missing ones).

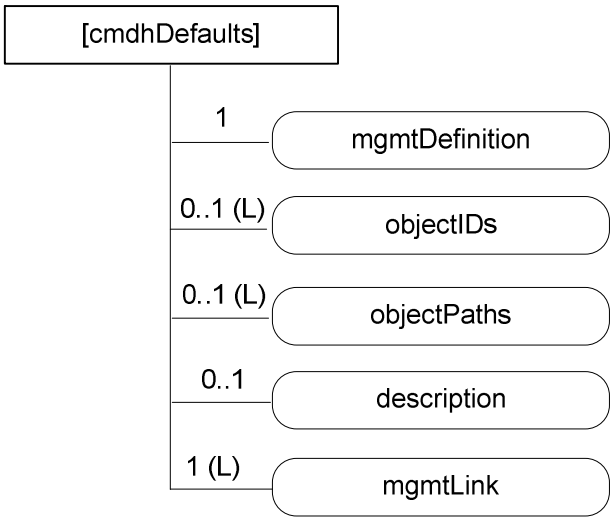


Figure D.12.2-1: Structure of *[cmdhDefaults]* resource

The *[cmdhDefaults]* resource shall contain attributes specified in table D.12-2-1.

Table D.12.2-1: Attributes of *[cmdhDefaults]* resource

Attributes of *[cmdhDefaults]*	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
labels	0..1(L)	RO	See clause 9.6.1.3
mgmtDefinition	1	WO	See clause 9.6.15. Has fixed value *"cmdhDefaults"*.
objectIDs	0..1 (L)	WO	See clause 9.6.15.
objectPaths	0..1 (L)	WO	See clause 9.6.15.
description	0..1	RW	See clause 9.6.15.
mgmtLink	1 (L)	RW	A list containing at least 2 links: • One or more link(s) to *[cmdhDefEcValue]* resource(s); and • One or more link(s) to *[cmdhEcDefParamValues]* resource(s).

D.12.3 Resource *cmdhDefEcValue*

The *[cmdhDefEcValue]* resource is used to define a value for the ***Event Category*** parameter of an incoming request or response message that needs to be sent across the associated reference point when it is not defined.

Upon receiving when a request or response message needs to be sent across the associated reference point, the entity performing the CMDH processing CSE will go through all the *[cmdhDefEcValue]* resources (in the order of their *order* attribute), check the *requestOrigin* and any present *requestContext* and *requestCharacteristics* attributes to see if they match (see description of matching), and if they all do, assign the value stored in the *defEcValue* attribute to the ***Event Category*** parameter.

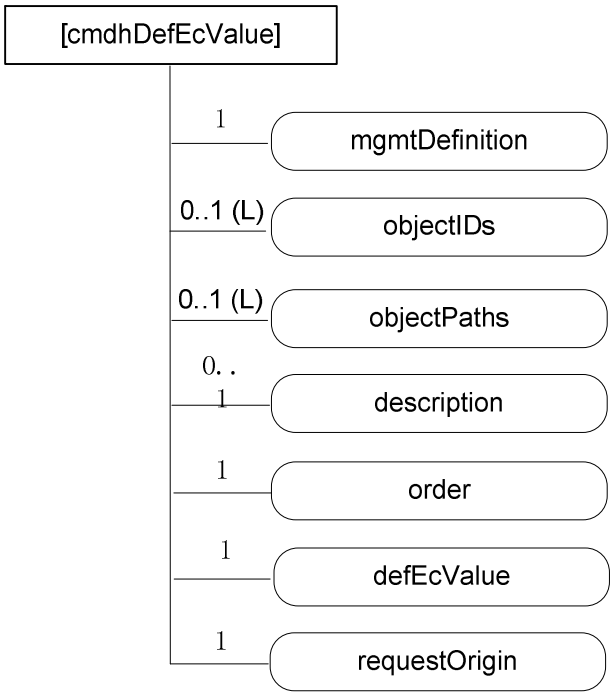


Figure D.12.3-1: Structure of *[cmdhDefEcValue]* resource

The *[cmdhDefEcValue]* resource shall contain attributes specified in table D.12.3-1.

Table D.12.3-1: Attributes of *[cmdhDefEcValue]* resource

Attributes of *[cmdhDefEcValue]*	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
labels	0..1(L)	RO	See clause 9.6.1.3.
mgmtDefinition	1	WO	See clause 9.6.15. Has fixed value *"cmdhDefEcValue"*.
objectIDs	0..1 (L)	WO	See clause 9.6.15.
objectPaths	0..1 (L)	WO	See clause 9.6.15.
description	0..1	RW	See clause 9.6.15.
order	1	RW	The index indicating in which order the *[cmdhDefEcValue]* resource will be treated by the entity performing the CMDH processing to determine a value for the ***Event Category*** parameter. This attribute is a specialization of *[objectAttribute]* attribute.
defEcValue	1	RW	The actual value to use for the ***Event Category*** parameter if the conditions expressed in the *requestOrigin* attribute all match. This attribute is a specialization of *[objectAttribute]* attribute.

Attributes of *[cmdhDefEcValue]*	Multiplicity	RW/ RO/ WO	Description
requestOrigin	1	RW	The *requestOrigin* attribute is a list of zero or more local *AE-IDs*, *App-IDs*, or the strings 'localAE' or 'thisCSE'. When an *AE-ID* appears in the *requestOrigin* attribute, the default ***Event Category*** value defined inside the *defEcValue* attribute is applicable for the ***Event Category*** if a request message was issued by that specific Application Entity or if a response message is targeting that specific Application Entity. When an *App-ID* appears in the *requestOrigin* attribute, the default ***Event Category*** value defined inside the *defEcValue* attribute is applicable for the ***Event Category*** if a request message was issued by an AE with that *App-ID* or if a response message is targeting an AE with that *App-ID* unless covered by another associated *[cmdhDefEcValue]* resource with a *requestOrigin* attribute containing its specific *AE-ID*. When the string 'localAE' appears in the *requestOrigin* attribute, the default ***Event Category*** value defined inside the *defEcValue* attribute is applicable for the ***Event Category*** for request messages issued by any local AEs or for response messages targeting any local AEs hosted on the node associated to this CMDH policy unless covered by another *[cmdhDefEcValue]* resource with a *requestOrigin* attribute containing the specific *AE-ID* or *App-ID* of the Originator of the request. When the string 'thisCSE' appears in the *requestOrigin* attribute, the default ***Event Category*** value defined inside the *defEcValue* attribute is applicable for the ***Event Category*** for request messages that are originating from the CSE hosted on the node associated to this CMDH policy or for response messages targeting that CSE. This is only valid if the associated node is an ASN or MN. The set of CMDH policies associated with a particular node shall contain at least one *[cmdhDefEcValue]* resource that contains 'localAE' in the *requestOrigin* attribute. The set of CMDH policies associated with a particular ASN or MN shall contain at least one *[cmdhDefEcValue]* resource that contains 'thisCSE' in the *requestOrigin* attribute. This attribute is a specialization of *[objectAttribute]* attribute.

D.12.4 Resource cmdhEcDefParamValues

The *[cmdhEcDefParamValues]* resource is used to represent a specific set of default values for the CMDH related parameters ***Request Expiration Timestamp***, ***Result Expiration Timestamp***, ***Operation Execution Time***, ***Result Persistence*** and ***Delivery Aggregation*** that are applicable for a given ***Event Category*** if any of the applicable these parameters are not specified in the request or response.

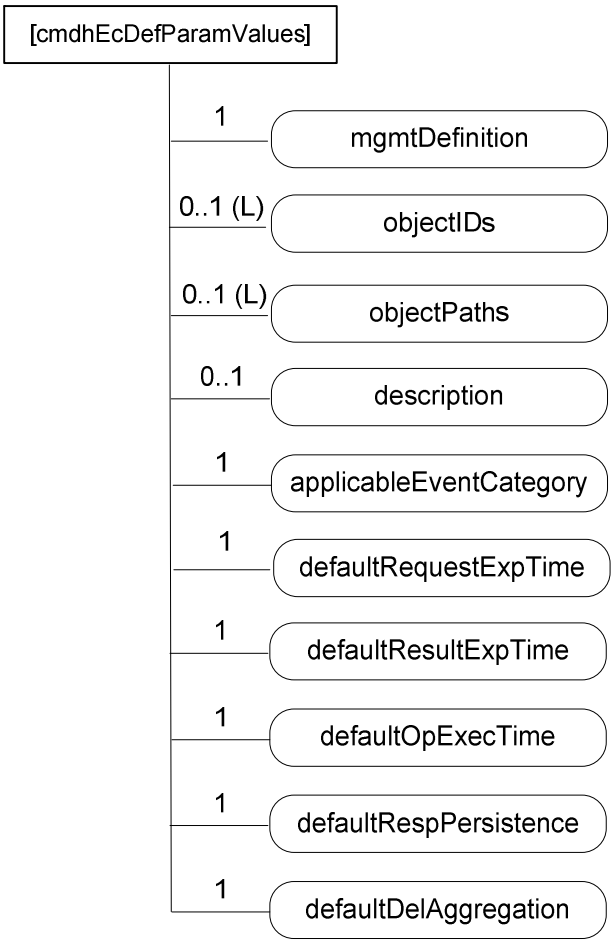


Figure D.12.4-1: Structure of *[cmdhEcDefParamValues]* resource

The *[cmdhEcDefParamValues]* resource shall contain attributes specified in table D.12.4-1.

Table D.12.4-1: Attributes of *[cmdhEcDefParamValues]* resource

Attributes of *[cmdhEcDefParamValues]*	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
labels	0..1(L)	RO	See clause 9.6.1.3.
mgmtDefinition	1	WO	See clause 9.6.15. Has fixed value *"cmdhEcDefParamValues"*.
objectIDs	0..1 (L)	WO	See clause 9.6.15.
objectPaths	0..1 (L)	WO	See clause 9.6.15.
description	0..1	RW	See clause 9.6.15.
applicableEventCategory	1	RW	This attribute defines the event categories for which this set of default parameters defined in this *[cmdhEcDefParamValues]* resource are applicable. This attribute is a list of zero or more ***Event Category*** values, or the string 'default'. When an Event Category value appears in the *applicableEventCategory* attribute, the set of default parameters defined in this *[cmdhEcDefParamValues]* resource are applicable for requests associated with that specific ***Event Category*** value. When the string 'default' appears in the *applicableEventCategory* attribute, the set of default parameters defined in this *[cmdhEcDefParamValues]* resource are applicable for all requests whose associated ***Event Category*** value is not listed in the *applicableEventCategory* attribute of any other provisioned *[cmdhEcDefParamValues]* resource linked to from the same *[cmdhDefaults]* resource. A specific ***Event Category*** value shall appear at most once in any of the *applicableEventCategory* attributes of any of the provisioned *[cmdhEcDefParamValues]* resources linked to from the same *[cmdhDefaults]* resource. The string 'default' shall appear exactly once in any of the *applicableEventCategory* attributes of any of the provisioned *[cmdhEcDefParamValues]* resources linked to from the same *[cmdhDefaults]* resource. This attribute is a specialization of *[objectAttribute]* attribute.
defaultRequestExpTime	1	RW	Default value for the ***Request Expiration Timestamp*** parameter in a request when the ***Request Expiration Timestamp*** parameter of the request is not set. This attribute is a specialization of *[objectAttribute]* attribute.
defaultResultExpTime	1	RW	Default value for the ***Result Expiration Timestamp*** parameter in a request or response when the ***Result Expiration Timestamp*** parameter of the request or response is not set. This attribute is a specialization of *[objectAttribute]* attribute.
defaultOpExecTime	1	RW	Default value for the ***Operation Execution Time*** parameter in a request when the ***Operation Execution Time*** parameter of the request is not set. This attribute is a specialization of *[objectAttribute]* attribute.
defaultRespPersistence	1	RW	Default value for the ***Result Persistence*** parameter in a request when the ***Result Persistence*** parameter of the request is not set. This attribute is a specialization of *[objectAttribute]* attribute.

Attributes of *[cmdhEcDefParamValues]*	Multiplicity	RW/ RO/ WO	Description
defaultDelAggregation	1	RW	Default value for the ***Delivery Aggregation*** parameter in a request when the ***Delivery Aggregation*** parameter of the request is not set. This attribute is a specialization of *[objectAttribute]* attribute.

D.12.5 Resource *cmdhLimits*

The *[cmdhLimits]* resource is used to define limits for CMDH related parameter values used in requests or response messages to be sent across the associated reference points by Originators (registered AEs or functions inside the CSE itself). When an incoming request or response is processed that does not comply with the limits defined by the corresponding *[cmdhLimits]* resource, the request message shall be rejected by the CSE.

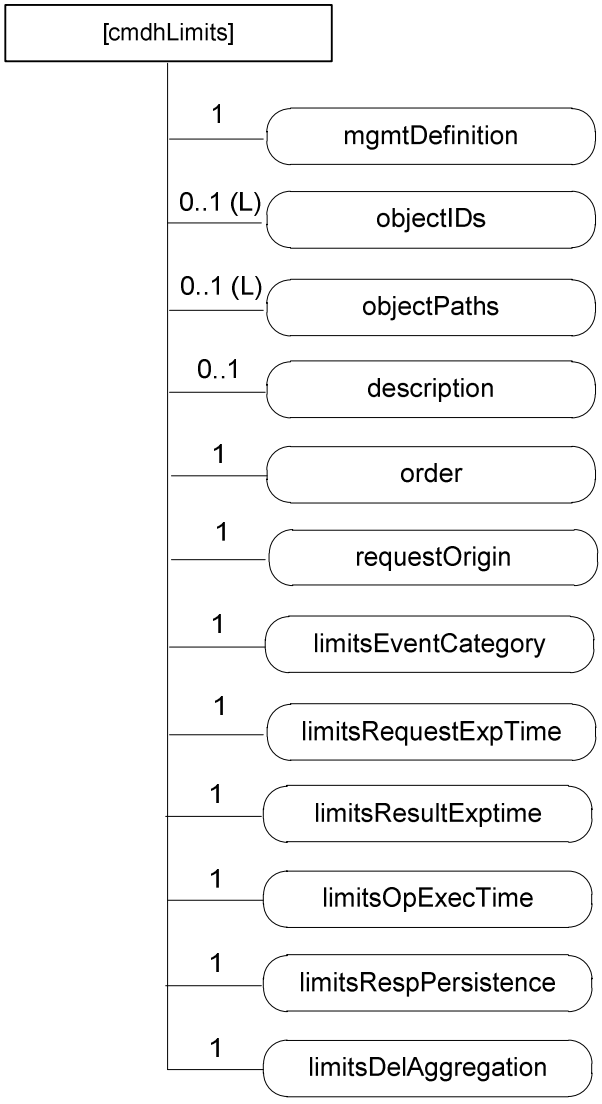


Figure D.12.5-1: Structure of *[cmdhLimits]* resource

The *[cmdhLimits]* resource shall contain attributes specified in table D.12.5-1.

Table D.12.5-1: Attributes of *[cmdhLimits]* resource

Attributes of *[cmdhLimits]*	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
labels	0..1(L)	RO	See clause 9.6.1.3.
mgmtDefinition	1	WO	See clause 9.6.15. Has fixed value *"cmdhLimits"*.
objectIDs	0..1 (L)	WO	See clause 9.6.15.
objectPaths	0..1 (L)	WO	See clause 9.6.15.
description	0..1	RW	See clause 9.6.15.
order	1	RW	The index indicating in which order the *[cmdhLimits]* resource will be treated by the CSE to determine a value for the limit parameters. This attribute is a specialization of *[objectAttribute]* attribute.
requestOrigin	1	RW	The *requestOrigin* attribute is a list of zero or more local *AE-IDs*, *App-IDs*, or the strings 'localAE' or 'thisCSE'. When an *AE-ID* appears in the *requestOrigin* attribute, the CMDH parameter limits defined inside *[cmdhLimits]* resources are applicable for requests issued by that specific Application Entity or for responses targeting that specific Application Entity. When an *App-ID* appears in the *requestOrigin* attribute, the CMDH parameter limits defined inside *[cmdhLimits]* resources are applicable for requests issued by an AE or responses targeting an AE with that *App-ID* unless already covered by another *[cmdhLimits]* resource with a *requestOrigin* attribute containing its specific *AE-ID*. When the string 'localAE' appears in the *requestOrigin* attribute, CMDH parameter limits defined inside *[cmdhLimits]* resources are applicable for all local AEs hosted on the node associated to this CMDH policy unless covered by another *[cmdhLimits]* resource with a *requestOrigin* attribute containing the specific *AE-ID* or *App-ID* of the Originator of the request. When the string 'thisCSE' appears in the *requestOrigin* attribute, CMDH parameter limits defined inside *[cmdhLimits]* resources are applicable for all requests that are originating from the CSE hosted on the node associated to this CMDH policy. This is only valid if the associated node is an ASN or MN. The set of CMDH policies associated with a particular node shall contain at least one *[cmdhLimits]* resource that contains 'localAE' in the *requestOrigin* attribute. The set of CMDH policies associated with a particular ASN or MN shall contain at least one *[cmdhLimits]* resource that contains 'thisCSE' in the *requestOrigin* attribute. This attribute is a specialization of *[objectAttribute]* attribute.
limitsEventCategory	1	RW	Allowed values for the ***Event Category*** parameter) in a request from or response to any of the Originators indicated in the *requestOrigin* attribute. This attribute is a specialization of *[objectAttribute]* attribute.

Attributes of *[cmdhLimits]*	Multiplicity	RW/ RO/ WO	Description
limitsRequestExpTime	1	RW	Range of allowed values for the ***Request Expiration Timestamp*** parameter in a request of any of the Originators indicated in the *requestOrigin* attribute. This attribute is a specialization of *[objectAttribute]* attribute.
limitsResultExpTime	1	RW	Range of allowed values for the ***Result Expiration Timestamp*** parameter in a request from or response to any of the Originators indicated in the *requestOrigin* attribute. This attribute is a specialization of *[objectAttribute]* attribute.
limitsOpExecTime	1	RW	Range of allowed values for the ***Operation Execution Time*** parameter in a request of any of the Originators indicated in the *requestOrigin* attribute. This attribute is a specialization of *[objectAttribute]* attribute.
limitsRespPersistence	1	RW	Range of allowed values for the ***Result Persistence*** parameter in a request of any of the Originators indicated in the *requestOrigin* attribute. This attribute is a specialization of *[objectAttribute]* attribute.
limitsDelAggregation	1	RW	List of allowed values for the ***Delivery Aggregation*** parameter in a request of any of the Originators indicated in the *requestOrigin* attribute. This attribute is a specialization of *[objectAttribute]* attribute.

D.12.6 Resource cmdhNetworkAccessRules

The *[cmdhNetworkAccessRules]* resource is used to define the rules for usage of Underlying Networks for forwarding information across the associated reference point of the node associated to this set of CMDH policies to other CSEs during processing of CMDH-related requests in a CSE. When request or response messages need to be sent across the associated reference point, the associated node, an incoming request is processed by a CSE, it can only use Underlying Networks for forwarding any information to other CSEs in compliance with the rules defined by the corresponding *[cmdhNetworkAccessRules]* resource.

If a pending request cannot be successfully completed in compliance with the rules defined in the corresponding *[cmdhNetworkAccessRules]* resource, that request shall be responded to with an unsuccessful response in case it has not already been accepted by the Receiver CSE or it has to be purged. Error reporting on failed CMDH processing depends on error reporting parameters.

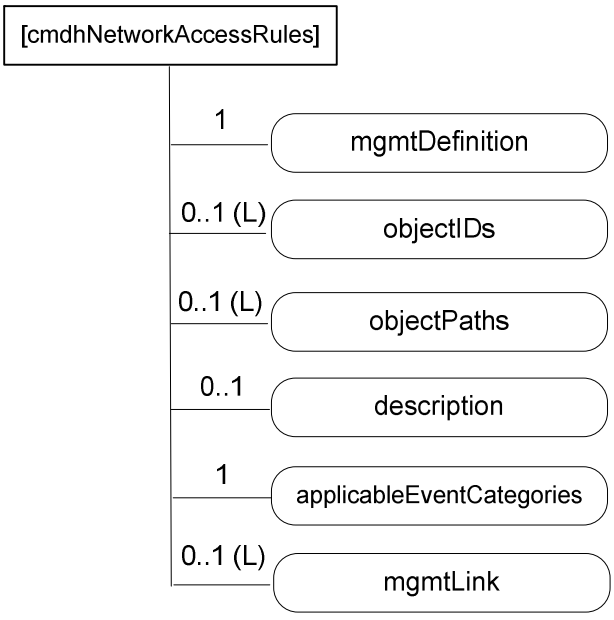


Figure D.12.6-1: Structure of *[cmdhNetworkAccessRules]* resource

If a *[cmdhNetworkAccessRules]* resource has no *mgmtLink* attribute to *[cmdhNwAccessRules]* resources (i.e. multiplicity of 0), request or response messages that match with the *applicableEventCategorie*s attribute (see description of attributes in table D.12.6-1) will not be allowed to use any Underlying Network for forwarding information, i.e. such messages need to be rejected.

The *[cmdhNetworkAccessRules]* resource shall contain attributes specified in table D.12.6-1.

Table D.12.6-1: Attributes of *[cmdhNetworkAccessRules]* resource

Attributes of *[cmdhNetworkAccessRules]*	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
labels	0..1(L)	RO	See clause 9.6.1.3.
mgmtDefinition	1	WO	See clause 9.6.15. Has fixed value *"cmdhNetworkAccessRules*".
objectIDs	0..1 (L)	WO	See clause 9.6.15.
objectPaths	0..1 (L)	WO	See clause 9.6.15.
description	0..1	RW	See clause 9.6.15.
applicableEventCategories	1	RW	This attribute defines for which requests or responses the rules contained in *[cmdhNwAccessRule]* resources linked from this *[cmdhNetworkAccessRules]* resource shall be applied. This attribute is a list of zero or more ***Event Category*** values, or the string 'default'. When an ***Event Category*** value appears in the *applicableEventCategories* attribute, the network usage rules defined inside *[cmdhNwAccessRule]* child resources are applicable for requests or responses associated with that specific ***Event Category*** value. When the string 'default' appears in the *applicableEventCategories* attribute, the network usage rules defined inside *[cmdhNwAccessRule]* child resources are applicable for all requests or responses whose associated ***Event Category*** value is not listed in the *applicableEventCategories* attribute of any other provisioned *[cmdhNetworkAccessRules]* resource linked to from the same *[cmdhPolicy]* resource. A specific ***Event Category*** value shall appear at most once in any of the *applicableEventCategories* attributes of any of the provisioned *[cmdhNetworkAccessRules]* resources linked to from the same *[cmdhPolicy]* resource. The string 'default' shall appear exactly once in any of the *applicableEventCategories* attributes of any of the provisioned *[cmdhNetworkAccessRules]* resources linked to from the same *[cmdhPolicy]* resource. This attribute is a specialization of *[objectAttribute]* attribute.
mgmtLink	0..1 (L)	RW	List of link(s) to *[cmdhNwAccessRule]* resource(s)

D.12.7 Resource *cmdhNwAccessRule*

The *[cmdhNwAccessRule]* resource is used define limits in usage of specific Underlying Networks for forwarding information to other CSEs during processing of CMDH-related requests across the associated reference point.

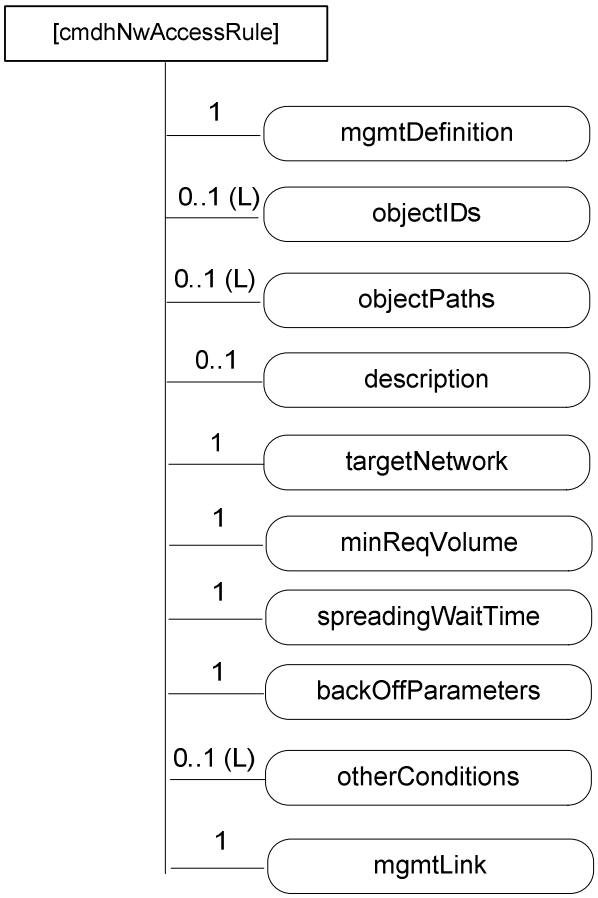


Figure D.12.7-1: Structure of *[cmdhNwAccessRule]* resource

Requests or responses matching the *applicableEventCategories* attribute of the parent *[cmdhNetworkAccessRules]* resource of this *[cmdhNwAccessRule]* resource are processed for forwarding via the associated reference points to other CSEs. The Underlying Network(s) subject to the rules represented by an instance of the *[cmdhNwAccessRule]* resource allowed for potential communication of those requests or responses are indicated by the *targetNetwork* attribute. The allowed schedule is indicated by the <*schedule*> resource pointed at by the *mgmtLink* attribute (see description of attributes in table D.12.7-1).

The *[cmdhNwAccessRule]* resource shall contain attributes specified in table D.12.7-1.

Table D.12.7-1: Attributes of *[cmdhNwAccessRule]* resource

Attributes of *[cmdhNwAccessRule]*	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
labels	0..1(L)	RO	See clause 9.6.1.3.
mgmtDefinition	1	WO	See clause 9.6.15. Has fixed value *"cmdhNwAccessRules"*.
objectIDs	0..1 (L)	WO	See clause 9.6.15.
objectPaths	0..1 (L)	WO	See clause 9.6.15.
description	0..1	RW	See clause 9.6.15.
targetNetwork	1	RW	The *targetNetwork* attribute defines for which Underlying Networks the usage limits contained in this *[cmdhNwAccessRule]* resource shall be applied. The *targetNetwork* attribute is a list of one or more strings identifying identifiers of Underlying Networks or response messages or the string 'default'. When an identifier of an Underlying Network appears in the *targetNetwork* attribute, the usage limits contained in this *[cmdhNwAccessRule]* resource shall be applied for usage of that specific Underlying Network when processing request or response messages matching with the parent *[cmdhNetworkAccessRules]* resource's *applicableEventCategories* attribute. When the string 'default' appears in the *targetNetwork* attribute, the usage limits contained in this *[cmdhNwAccessRule]* resource shall be applied for usage of all Underlying Networks that are not listed with their specific identifiers in the *targetNetwork* attribute of any other *[cmdhNwAccessRule]* linked to from the same parent *[cmdhNetworkAccessRules]* resource when processing request or response messages matching with the associated *[cmdhNetworkAccessRules]* resource's *targetNetwork* attribute. Each Underlying Network identifier or the string 'default' shall appear at most once in any of the *targetNetwork* attributes of any of the provisioned *[cmdhNwAccessRule]* child resources linked to by the same parent *[cmdhNetworkAccessRules]* resource. This attribute is a specialization of *[objectAttribute]* attribute.
minReqVolume	1	RW	Minimum amount of data that needs to be aggregated before any of the Underlying Networks matching with the *targetNetwork* attribute of this *[cmdhNwAccessRule]* resource can be used for forwarding information.
spreadingWaitTime	1	RW	This parameter consists of a number SWT such that before accessing the underlying network (typically to forward an incoming request), the CSE will wait for an additional amount of time randomly chosen between 0 and SWT. This attribute is a specialization of *[objectAttribute]* attribute.

Attributes of *[cmdhNwAccessRule]*	Multiplicity	RW/ RO/ WO	Description
backOffParameters	1	RW	Parameters that define how usage of any of the Underlying Networks matching with the *targetNetwork* attribute of this *[cmdhNwAccessRule]* resource shall be handled by field nodes when attempts to use such networks have failed. These parameters only apply to communication attempts by field nodes. The *backOffParameters* attribute can either: - Consist of the following values: • An initial back-off time IBT that defines how long a CSE needs to wait before attempting to use a specific Underlying Network again after a first failed attempt. • An additional back-off time ABT increment that defines by how much the back-off time shall be increased after each additional consecutive failed attempt to use the same Underlying Network without success. • A maximum back-off time MBT that defines the maximum wait time before attempting to use an Underlying Network again after previous failures. • An optional random back-off time RBT that will make the network access actually occur randomly in a time window starting at IBT+n.ABT and ending at IBT+n.ABT+RBT (if RBT is not present, then no randomization occurs and the access takes place at IBT+n.ABT). In which case the back-off timers apply for any action attempted onto the network (registration to the network, opening of data session, etc.). - Or consist of an array of several elements, each composed like this [NWA, IBT, ABT, MBT, (optional RBT)] where IBT, ABT, MBT and RBT are defined above, and where NWA is the name of a specific action that is actually attempted on the network. The present document defines the following network action names, that can be used when the CSE knows that it uses an underlying network where these actions are valid: • "cellular-registration" for an IMSI CS-Registration onto 3GPP-compliant cellular networks. • "cellular-attach" for a GPRS Attach onto 3GPP-compliant cellular networks. • "cellular-pdpctxact" for a PDP Context Activation onto 3GPP-compliant cellular networks. • "cellular-sms" for SMS originating from this CSE onto 3GPP-compliant cellular networks. • "default" for all other actions not already declared in this backOffParameters attribute (this action will be used by the CSE when it does not know which kind of underlying network it uses). In which case the back-off timers apply only for the specified actions. This attribute is a specialization of *[objectAttribute]* attribute.
otherConditions	0..1 (L)	RW	List of additional conditions that need to be fulfilled before any of the Underlying Networks matching with the *targetNetwork* attribute of this *[cmdhNwAccessRule]* resource can be used for forwarding information to other CSEs. This attribute is a specialization of *[objectAttribute]* attribute.
mgmtLink	1	RW	Link to an instance *allowedSchedule* of a <*schedule*> resource as defined in clause 9.6.9. The linked <schedule> instance shall be a child of the <*node*> resource to which this resource corresponds. This attribute is a specialization of *[objectAttribute]* attribute.

D.12.8 Resource *cmdhBuffer*

The *[cmdhBuffer]* resource is used to define limits in usage of buffers for temporarily storing information that needs to be forwarded to other nodes during processing of CMDH-related requests in a CSE. When an incoming request or response message needs to be sent is processed by a node, it can only use buffers for temporary storage in compliance with the rules defined by the corresponding *[cmdhBuffer]* resource.

If a request cannot be processed in compliance with the rules defined in the corresponding *[cmdhBuffer]* resource, that request shall either be rejected in case it has not already been accepted by the Receiver CSE or it has to be purged. Error reporting on failed CMDH processing depends on error reporting parameters.

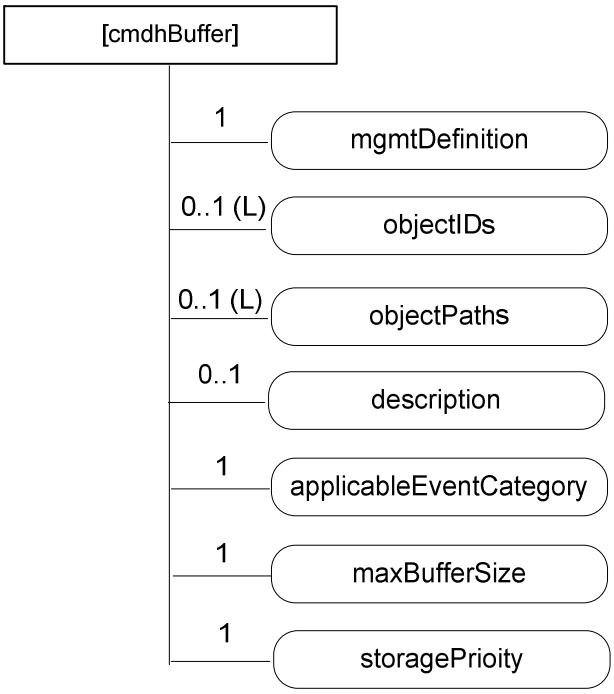


Figure D.12.8-1: Structure of *[cmdhBuffer]* resource

The *[cmdhBuffer]* resource shall contain attributes specified in table D.12.8-1.

Table D.12.8-1: Attributes of *[cmdhBuffer]* resource

Attributes of *[cmdhBuffer]*	Multiplicity	RW/ RO/ WO	Description
resourceType	1	RO	See clause 9.6.1.3.
resourceID	1	RO	See clause 9.6.1.3.
resourceName	1	WO	See clause 9.6.1.3.
parentID	1	RO	See clause 9.6.1.3.
expirationTime	1	RW	See clause 9.6.1.3.
accessControlPolicyIDs	0..1 (L)	RW	See clause 9.6.1.3.
creationTime	1	RO	See clause 9.6.1.3.
lastModifiedTime	1	RO	See clause 9.6.1.3.
labels	0..1(L)	RO	See clause 9.6.1.3.
mgmtDefinition	1	WO	See clause 9.6.15. Has fixed value *"cmdhBuffer"*.
objectIDs	0..1 (L)	WO	See clause 9.6.15.
objectPaths	0..1 (L)	WO	See clause 9.6.15.
description	0..1	RW	See clause 9.6.15.

Attributes of *[cmdhBuffer]*	Multiplicity	RW/ RO/ WO	Description
applicableEventCategory	1	RW	The *applicableEventCategory* attribute defines for which request or response messages the limits contained in this *[cmdhBuffer]* resource shall be applied. The *applicableEventCategory* attribute is a list of zero or more ***Event Category*** values, or the string 'default'. When an Event Category value appears in the *applicableEventCategory* attribute, the buffer usage limits defined inside this *[cmdhBuffer]* resource are applicable for request or response messages associated with that specific ***Event Category*** value. When the string 'default' appears in the *applicableEventCategory* attribute, the buffer usage limits defined inside this *[cmdhBuffer]* resource are applicable for all request or response messages whose associated ***Event Category*** values not listed in the *applicableEventCategory* attribute of any other provisioned *[cmdhBuffer]* resource linked to from the same *[cmdhPolicy]* resource. A specific ***Event Category*** value shall appear at most once in any of the *applicableEventCategory* attributes of any of the provisioned *[cmdhBuffer]* resources linked to from the same *[cmdhPolicy]* resource. The string 'default' shall appear exactly once in any of the *applicableEventCategory* attributes of any of the provisioned *[cmdhBuffer]* resources linked to from the same *[cmdhPolicy]* resource. This attribute is a specialization of *[objectAttribute]* attribute.
maxBufferSize	1	RW	Maximum amount of memory that can be used for buffering requests matching with the *applicableEventCategory* attribute of this *[cmdhBuffer]* resource. This attribute is a specialization of *[objectAttribute]* attribute.
storagePriority	1	RW	Storage priority for data that is stored for buffering request or response messages matching with the *applicableEventCategory* attribute of this *[cmdhBuffer]* resource. The storage priority defines the how to handle purging of buffered data when buffer memory is exhausted and buffered request or response messages need to be purged. Buffered request or response messages associated with a lower storage priority shall be purged before buffered request or response messages with a higher storage priority. The range of storage priority is from 1 to 10. This attribute is a specialization of *[objectAttribute]* attribute.

Annex E (informative): CSE Minimum Provisioning

The present clause defines the minimum set of resources instantiated in a CSE node with the scope to make it ready to provide services to entities that will register to.

For the purpose of the initial configuration two roles are identified:

- **superuser:** this role allows the full CSE control according to infrastructure provider policies. Only one superuser role is allowed per CSE;
- **user:** is the role associated to an AE that will register itself to Registrar CSE. More than one user roles are allowed per CSE. More than one applications can access to CSE with the same role.

Superuser role may be created with the following associated resources:

1) Definition or assignment of CSE-ID name that may be unique in the node hosting the CSE to be instantiated.
2) Creation of <*CSEBase*> resource with name equal to CSE-ID.
3) Creation of following child resources belonging to a tree with <CSEBase> as root:
    a) <*accessControlPolicy*> child resource enabling full access control for superuser's invoked operations to the tree resources. Subsequent created resources may have *accessControlPolicyIDs* attribute addressing this <*accessControlPolicy*> resource.
    b) <*AE*> child resource to be used as registered AE dedicated to superuser related activities.

Each user role may be created with the following associated resources:

1) Definition or assignment of an AE name that may be unique in the CSE.
2) Creation of <*AE*> child resource of <*CSEBase*> resource named as described in step 1, to be used as registered application dedicated to user related activities.
3) Creation of following child resources belonging to a tree with <*AE*> as root:
    a) <*accessControlPolicy*> resource enabling partial access control (e.g. these resources cannot be deleted be the user, superuser's resources can only be read by user) for user's invoked operations to the tree resources. <*AE*> resource can be updated with *accessControlPolicyIDs* attribute addressing <*accessControlPolicy*> resource.

The above described operations may be executed in the node in order provide the elements and the access control privileges required to provide the initial access to resource operations.

Same user can create more than one <*AE*> resources and other child resource types.

Once user role resource trees have been created the registered AE associated to <*AE*> resource (defined for a user role in step 2) is able to create its own <*container*> resource to store business logic application data that can be shared to other registered AEs in a controlled way acting on its own <*accessControlPolicy*> resource.

Annex F (informative): Interworking/Integration of non-oneM2M solutions and protocols

F.1 Introduction

Non-oneM2M solutions are currently installed and will continue to evolve and to be adopted in future for specific deployments. Some of these solution are the evolution of M2M that have a long history and significant mass installations (e.g. the PLC-related protocols commonly used in building and industrial automation), and are also significantly represented by proprietary solutions, especially in terms of semantic of the data model. The non-oneM2M solutions are potentially used for:

- Legacy deployment: such solutions can make use of both, proprietary or standard protocols; often proprietary data models and functionality are combined with the use of standard protocol.
- New system deployment that privilege the vertical optimization rather the horizontal aspects.
- Area network deployment for which native IP based oneM2M is perceived as not optimized respect to the used technology.

For those non-oneM2M solutions oneM2M needs to provide a means to enable:

- Mixed deployment that are partially oneM2M compliant and partially not, where the oneM2M System provides the solution to integrate multiple technologies (e.g. to add new technologies on top of old installations).
- Hybrid deployment that are still using non-oneM2M protocol (proprietary/standard) and want to use at the same time some of the oneM2M functionalities. A typical case is the exchange of heavy data traffic outside the CSE (e.g. for video surveillance), together with the use of CSE services for control and light traffic exchange.

Behaviour of such non-oneM2M solution is out of scope in present document, but there is some market need to communicate with devices in non-oneM2M domain (so called 'NoDN').

Since NoDN does not have any knowledge about the oneM2M system, AE will take responsibility to bridge those two worlds which are Interworking Proxy Entities (IPEs).

The present annex provides oneM2M guidance regarding how to implement interworking between the oneM2M solution and external non-oneM2M systems.

F.2 Interworking with non-oneM2M solutions through specialized interworking applications

The solution is based on the use of specialized interworking Application Entities that are interfaced to the CSE via standard Mca reference points.

Such specialized applications are named Inter-working Proxy and are described in figure F.2-1.

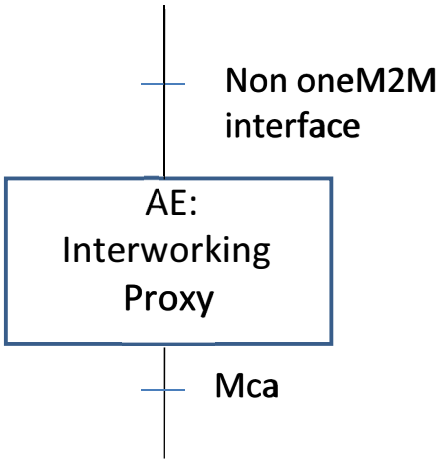


Figure F.2-1: Interworking Proxy

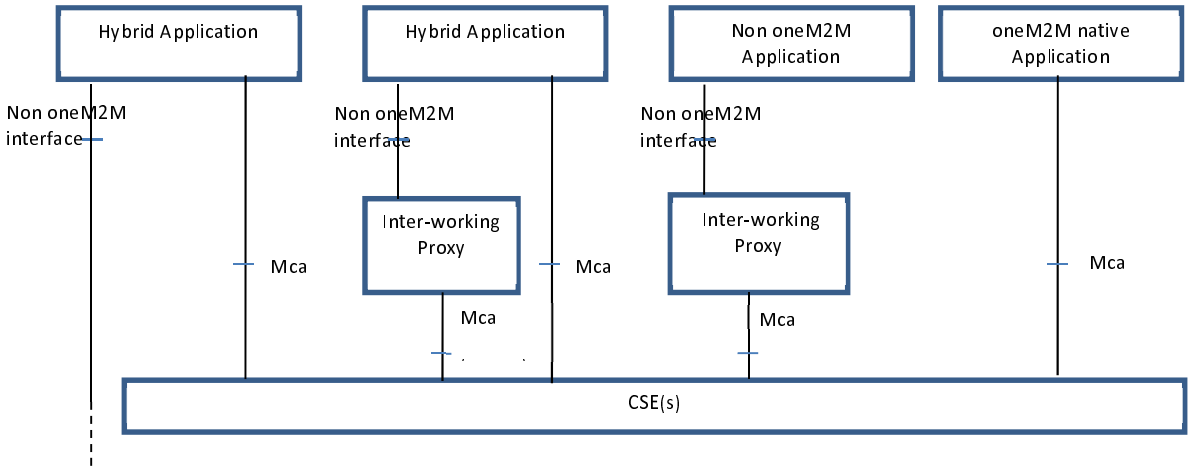
The Inter-working Proxy Application Entity (IPE) is characterized by the support of a non-oneM2M reference point, and by the capability of remapping the related data model to the oneM2M resources exposed via the Mca reference point.

This is typically supported via a full semantic inter-working of the data model used by the non oneM2M and a related protocol inter-working logic, and, depending on the complexity of the non oneM2M data model, can imply the definition of a complex set of resources built via the basic oneM2M ones, or a simple direct mapping of the communication via the containers and its variants (e.g. <container>, <flexContainer>, and <timeSeries>).

The approach enable a unique solution for enabling communications among different protocols, catering for different level of inter-working including protocol inter-working, semantic information exchange, data sharing among the different solution and deployments.

And enables the offering additional values respect to what is today available via existing protocols and proprietary service exposures.

Figure F.2-2 shows the typical scenarios supported by the oneM2M architecture in the context of inter-working. The combination of the different scenarios allows mixed deployments.



NOTE: The additional option of an inter-working proxy embedded in the CSE as a module with an internal specified interface is under consideration.

Figure F.2-2: Scenarios Supported by oneM2M Architecture

These scenarios are applicable to the CSE with the AE as application dedicated node, in the application Service Node, in the Middle Node and in the infrastructure Node.

The following picture provides an example of the use of such capabilities an area network adopting specific protocols, e.g. Zigbee Telco Profile and Mbus using COSEM Data model.

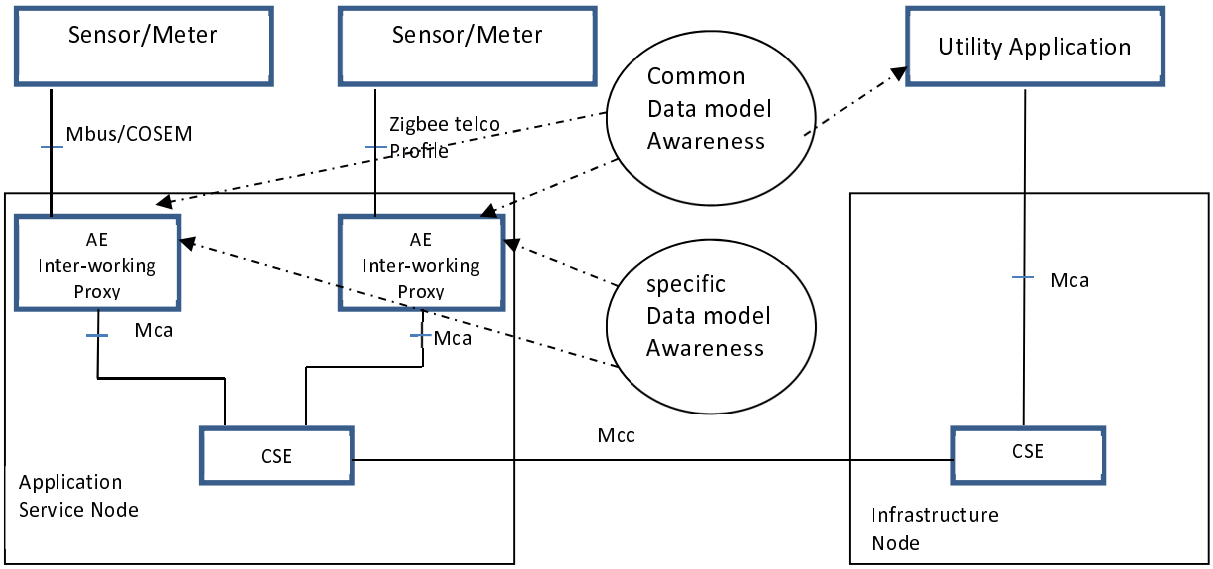


Figure F.2-3: Translation of non-oneM2M Data Model to oneM2M Specific Data Model

There exist three variants of how interworking through an Inter-working Proxy Application Entity over Mca can be supported:

1) Interworking with full mapping of the semantic of the non-oneM2M data model to Mca.

   This is typically supported via a full semantic inter-working of the data model used by the non-oneM2M solution and the generic data model used in oneM2M (based on usage of containers and its variants) for exchanging application data. The IPE includes the related protocol inter-working logic.

   Depending on the complexity of the non-oneM2M data model, this can imply that the Inter-working Proxy Application Entity constructs a complex set of resources (built from the basic oneM2M resources) in the CSE. These resources are oneM2M representations of the non-oneM2M data model and are exposed by the IPE on Mca. They enable CSEs and AEs to access the entities in the non-oneM2M via the IPE.

   The benefit of this level of interworking is that it offers a unique solution for enabling communications among different protocols. The data model of the non-oneM2M solution determines its representation (the names, data types and structure of the oneM2M sub resources) in the M2M System. It caters for different levels of inter-working including protocol inter-working, semantic information exchange, data sharing among the different solution and deployments. It enables offering additional values with respect to what is today available via existing protocols and proprietary service exposures.

NOTE: With this level of interworking an M2M Application can access non-oneM2M solutions without the need to know the specific protocol encoding for these solutions. A drawback is that the IPE also potentially needs to interwork between a non-oneM2M security solution and oneM2M security. E.g. it needs to be the termination point of any non-oneM2M specific encryption.

2) Interworking using containers for transparent transport of encoded non-oneM2M data and commands via Mca.

   In this variant non-oneM2M data and commands are transparently packed by the Inter-working Proxy Application Entity into containers for usage by the CSEs and AEs.

   In this case the CSE or AE needs to know the specific protocol encoding rules of the non-oneM2M Solution to be able to en/de-code the content of the containers.

3) Interworking using a retargeting mechanism.

   This is typically supported via gateway system which is capable to map operations on oneM2M world into non-oneM2M world.

   Either CSE or AE provided mapped interface as oneM2M resource structure, and when the operation is executed on the resource structure, the operations will be retargeted to the IPE.

This mapping may be provided for reverse direction, like status change of the non-oneM2M device will be reflected as the UPDATE on the oneM2M container or its variants.

F.3 Interworking versus integration of non-oneM2M solutions

Interworking:

With the approach given above - where specialized interworking applications (IPEs) allow to interact with any non-oneM2M system via the Mca interface - proprietary non-oneM2M solutions as well as non-oneM2M solutions that follow open standards can be interworked with the oneM2M System.

Integration:

When it is desired to make a certain type of non-oneM2M solution (e.g. some type of non-IP based Area Network) a permanent part of the deployed oneM2M Solution then the functionality of the Inter-working Proxy Application Entity can be integrated into the CSE of an Application Node. This is called "Integration" non-oneM2M solutions.

F.4 Entity-relation representation of non-IP based M2M Area Network

F.4.0 Overview

Figure F.4.0-1 provides an entity-relation model that represents a non-IP based M2M area network as well as its relationship to an Interworking Proxy Application Entity (IPE).

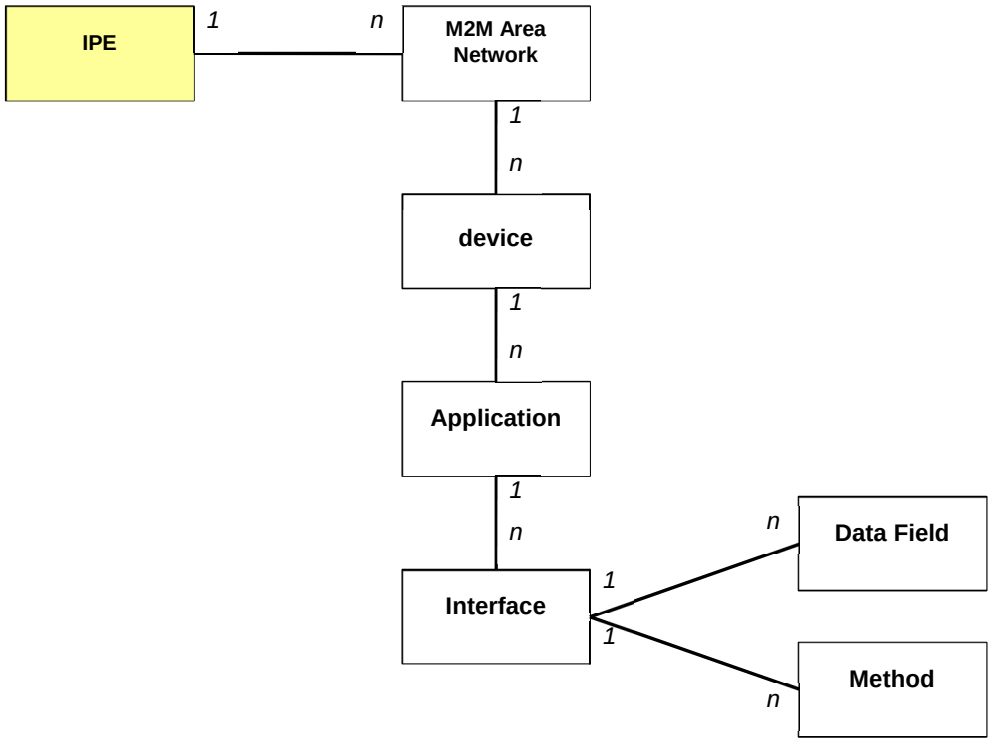


Figure F.4.0-1: Generic entity-relation diagram for an IPE and an M2M Area Network running legacy devices

This entity-relation diagram is e.g. applicable to the following M2M Area Networks:

- ZigBee area network.
- DLMS/COSEM area network.
- Zwave area network.
- BACnet area network.
- ANSI C12 area network.
- Mbus area network.

F.4.1 Responsibilities of Interworking Proxy application Entity (IPE)

More specifically, the IPE is responsible to:

- create oneM2M resources representing the M2M Area Network structure (devices, their applications and interfaces) in the oneM2M Service Capability Layer, accessible via Mca;
- manage the oneM2M resources in case the M2M Area Network structure changes;
- discover the M2M Area Network structure and its changes automatically if this is supported by the technology of the M2M Area Network.

NOTE: Mapping principles of the none-oneM2M information model into oneM2M resources are not specified in this version of the specification.

Annex G:
Void

Annex H (informative): Object Identifier Based M2M Device Identifier

H.1 Overview of Object Identifier

In M2M systems, it is required for devices to be distinguishable from one another through some kind of ID system. In other words, the ID which is allocated to the device is globally unique to ensure the proper operation of M2M systems, such as finding and connecting devices.

In relation to this requirement, the use of Object Identifiers may provide a convenient method to ensure the global uniqueness of M2M devices. The Object Identifier (OID) is an identification mechanism jointly developed by ITU-T and ISO/IEC which can be applied to objects, concepts, and all kinds of tangible or intangible things.

OID uses a hierarchical tree structure and is represented as a sequence of integer values, as shown in figure H.1-1. OID consists of several segments called arcs which provide placeholders for identification and description in the hierarchal tree. The first arc can take the following values:

- itu-t (0);
- iso (1); and
- joint-iso-itu-t (2).

An OID is hierarchically allocated to an entity (e.g. an organization, a country, etc.) which has the authority to define lower arcs. For example, ITU-T can manage and allocate lower arcs below itu-t (0), and ISO can allocate lower arcs below iso (1). The general procedure regarding the use of OID is described in Recommendation ITU-T X.660 | ISO/IEC 9834-1 [i.24].

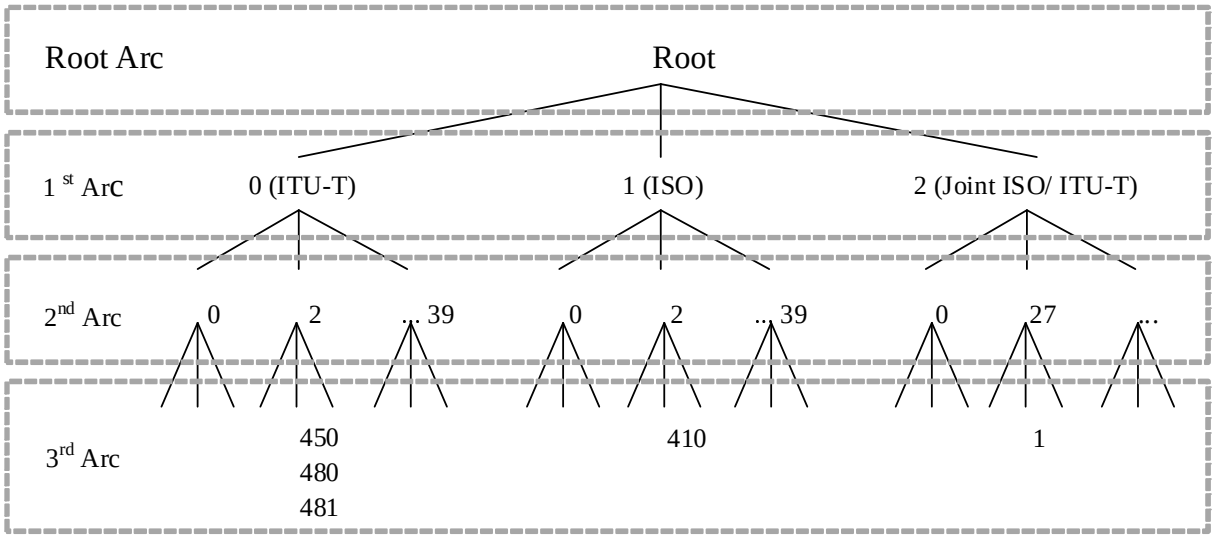


Figure H.1-1: International OID Tree

H.2 OID Based M2M Device Identifier

H.2.0 Overview

An M2M device will be identified individually through a globally unique ID system. This clause explains how to allocate a globally unique ID to each M2M device by using the OID scheme. M2M device ID is an example which shows that OID can be applied to any M2M identifiers which need globally unique IDs.

The M2M device ID consists of a higher arc and a sequence of four arcs. It takes the form of {(higher arc) (x) (y) (z) (a)} as illustrated in figure H.2.0-1. The higher arc is defined and managed according to the OID procedure. Each arc in the remaining sequence of four arcs represents the manufacturer ID, product model ID, serial number ID, and expanded ID, respectively.

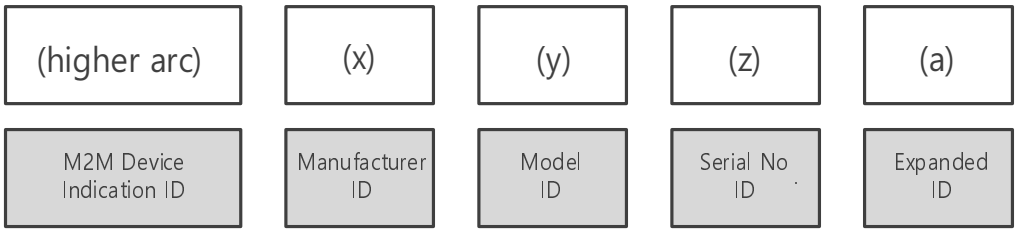


Figure H.2.0-1: M2M Device ID

H.2.1 M2M Device Indication ID - (higher arc)

The M2M Device Indication ID (higher arc) represents a globally unique identifier for the M2M device. The composition of the highest arc is variable and may be composed of several sub-arcs. The higher arc is assigned and managed by ITU-T or ISO.

H.2.2 Manufacturer ID - (x)

The 1st arc (x) among the sequential 4 arcs is used to identify the manufacturer which produces the M2M device. The first arc (x) is managed and allocated by the authority related with (higher arc).

H.2.3 Model ID - (y)

The 2nd arc (y) among the sequential 4 arcs identifies the device model produced by the manufacturer x. The second arc is managed and allocated by the manufacturer represented by the (x) arc.

H.2.4 Serial Number ID - (z)

The 3rd arc (z) among the sequential 4 arcs is for identifying the serial number of the device model y. The third arc is managed and allocated by the manufacturer represented by the (x) arc.

H.2.5 Expanded ID - (a)

The 4th arc (a) among the sequential 4 arcs is for identifying the legacy device which operates under the M2M device. The 4th arc for Expanded ID is allocated by the M2M device by adding a 4th arc to its device ID {(higher arc) (x) (y) (z)}. Therefore, the ID of legacy device which operates under the M2M device takes the form of {(higher arc) (x) (y) (z) (a)}. The fourth arc is managed and allocated by the M2M device.

H.3 Example of M2M device ID based on OID

Assume an M2M Device ID of {0 2 481 1 100 3030 10011}. The M2M device ID can be interpreted as follows:

- (0 2 481 1) in {0 2 481 1 100 3030 10011} - represents the M2M Device Indication ID (higher arc):
  - (0) in {0 2 481 1 100 3030 10011} - identifies the managing organization ITU-T.
  - (2) in {0 2 481 1 100 3030 10011} - identifies "Administration".
  - (481) in {0 2 481 1 100 3030 10011} - identifies the data country code for Korea.
  - (1) in {0 2 481 1 100 3030 10011} - identifies an M2M device.
- (100) in {0 2 481 1 100 3030 10011} - identifies the device Manufacturer.
- (3030) in {0 2 481 1 100 3030 10011} - identifies the device Model.
- (10011) in {0 2 481 1 100 3030 10011} - identifies the device Serial number.

Annex I:
Void

Annex J (normative): Syntaxes for content based discovery of <*contentInstance*>

J.1 Introduction

This annex specifies the syntax for contentFilterQuery filterCriteria (see clause 8.1.2).

The syntax of string for contentFilterQuery parameter shall be chosen by contentFilterSyntax parameter.

J.2 'jsonpath' query syntax

This syntax of query is applicable in the case of stored data in the <*contentInstance*> resource which is indicated as JSON based according to the *contentInfo* attribute value.

The target of evaluation shall be located by JSON path like addressing, which is constructed following rules:

- The entire data shall be referred by '$(dollar sign)' character.
- The notation '[n]' shall refer n-th member of JSON Array.
- The operator '.(dot)' followed by name shall refer member of JSON Objects.
- The name shall be surrounded with '"(quote)' characters when the name contains special characters, such as '$', '.', ' (space)', '[', ']', '{', and '}'.
- The ' (space)' character shall be inserted between reserved keyword and other component of query string.

The following keywords shall be used to construct query string when ***contentFilterSyntax*** parameter was 'JSON-path'.

Table J.2-1: Reserved keywords for JSON-Path query syntax

Keyword	Condition	Applicability
EQ (Equals)	When the target value equals with query.	String or number
NE (Not Equals)	When the target value does not equal with query.	String or number
GT (Greater Than)	When the target value was greater than number given as query.	Number only
LT (Less Than)	When the target value was less than number given as query.	Number only
GE (Greater or Equals)	When the targeted value was greater or equals with number given as query.	Number only
LE (Less or Equals)	When the targeted value was less or equals with number given as query.	Number only
MATCH	When the targeted value contains given string.	String only
AND	Concatenation of query which evaluated as AND combination logic.	Query-string
OR	Concatenation of query-string which evaluated as OR combination logic.	Query-string

Annex K (informative): Bibliography

IETF RFC 6874: "Representing IPV6 Zone Identifiers in Address Literals and Uniform Resources Identifiers".

History

Document history		
V3.22.0	February 2021	Publication