oneM2M **Technical Specification**	**oneM2M** **Technical Specification**
Document Number	TS-0014-V4.0.1
Document Name:	LWM2M Interworking
Date:	2024-11-09
Abstract:	The present document specifies the interworking capabilities of the M2M Service Layer between ASN/IN/MN CSEs and LWM2M Endpoints.

About oneM2M

The purpose and goal of oneM2M is to develop technical specifications which address the need for a common M2M Service Layer that can be readily embedded within various hardware and software, and relied upon to connect the myriad of devices in the field with M2M application servers worldwide.

More information about oneM2M may be found at: http//www.oneM2M.org

Editors's Note: Convert sequence diagrams to PlantUML

Contents

1 Scope
2 References
  2.1 Normative references
  2.2 Informative references
3 Definitions and abbreviations
  3.1 Definitions
  3.2 Abbreviations
4 Conventions
5 Architecture Model
  5.1 Introduction
  5.2 Reference Model

5.3 Types of Interworking
5.4 Composition of the Interworking Proxy Entity
6 Architecture Aspects
6.1 Introduction
6.2 LWM2M Device and Endpoint Lifecycle
6.2.1 Introduction
6.2.2 LWM2M Device and Endpoint Resource Representation
6.2.2.1 Introduction
6.2.2.2 LWM2M Device and Endpoint Resource Identification
6.2.2.3 LWM2M Endpoint Lifecycle
6.2.2.4 Configuration of CMDH Policies
6.2.2.5 Registering a Registered LWM2M Endpoint
6.3 LWM2M Object Discovery
6.3.1 Introduction
6.3.2 LWM2M Object Identifier Representation
6.3.2.1 Introduction
6.3.2.2 Void
6.3.2.3 LWM2M Object Lifecycle
6.4 LWM2M Object Transport and Interworking
6.4.1 Introduction
6.4.2 LWM2M Interworking Mechanisms
6.4.2.1 Introduction
6.4.2.2 Relevant Interworked Resource Settings
6.4.2.3 oneM2M RETRIEVE Procedure
6.4.2.4 oneM2M CREATE Procedure
6.4.2.5 oneM2M UPDATE Procedure
6.4.2.6 oneM2M DELETE Procedure
6.4.3 oneM2M Resource Operation Responses
6.5 LWM2M Object Subscription and Notification
6.5.1 Introduction
6.5.2 LWM2M Subscription Procedure
6.5.3 LWM2M Notification Procedure
6.6 LWM2M Object Security
6.6.1 Introduction
6.6.2 LWM2M Interworking Access Control Policy
6.6.3 IPE and Object Security provisioning
6.7 LWM2M IPE Administration and Maintenance
6.7.1 Introduction
6.7.2 Administration and Maintenance of the LWM2M Server Functionality
6.7.2.1 Introduction
6.7.2.2 LWM2M Server Maintenance
6.7.3 Maintenance of the LWM2M IPE AE Context
6.7.3.1 Introduction
6.7.3.2 LWM2M Endpoint List
6.7.3.3 Configuration of Interworking Functions
6.8 LWM2M Client Provisioning (Bootstrap)

7 Transparent Interworking Function
7.1 Introduction
7.2 Attribute Mapping for the <contentInstance> Resources
8 Semantically Enabled Interworking Function (Informative)
8.1 Introduction
8.2 Organization of Semantically Enabled Content Sharing Resources
8.2.1 Introduction
8.2.2 Lifecycle of Semantically Enabled Content Sharing Resources
8.2.3 Mapping for the Encoding of the <contentInstance> Resource
8.3 Guidelines for Mapping to the Base Ontology
8.3.1 Introduction
8.3.2 Mapping of the LWM2M Client
8.3.4 Mapping of the LWM2M Object, Object Instance. Resource and Resource Instance
8.3.4.1 Introduction
9 oneM2M Management Object-based Interworking Function
9.1 Introduction
9.2 Translation of oneM2M Management Resource Types
9.2.1 Introduction
9.2.2 Translation to <mgmtObj> Resource Types
9.2.2.1 Mapping to <mgmtObj> Resource Types
9.2.2.2 Interworking of <mgmtObj> Resources
9.2.2.2.1 Introduction
9.2.2.2.2 Interworking of <mgmtObj> Resource Settings
9.2.2.2.3 Synchronization *<mgmtObj>* resources
9.2.2.3 Example of creating new specialized <mgmtObj> resources
9.2.2.4 LWM2M Interworking Procedure
9.2.2.5 Use of oneM2M attribute level subscription in LWM2M Interworking
Annex A (Informative): Introduction to OMA LightweightM2M (LWM2M)
A.1 Introduction
A.2 Architecture
A.3 Terminology
A.4 Reference Points
A.4.1 Introduction
A.4.2 Functional Components
A.4.2.1 LWM2M Server
A.4.2.2 LWM2M Client
A.4.3 Interfaces
A.5 Protocols
A.5.1 Protocol Stack
A.5.2 Data Model
A.5.3 Interface Descriptions
A.5.3.1 Bootstrap
A.5.3.2 Client Registration
A.5.3.3 Device Management and Service Enablement

A.5.3.4 Information Reporting
A.6 Functions
History

1 Scope

The present document specifies the interworking capabilities of the M2M Service Layer between ASN/IN/MN CSEs and LWM2M Endpoints using the architecture identified in Annex F of oneM2M TS-0001 [2] for the following interworking scenarios:

- Interworking for transparent transport of encoded LWM2M Objects and commands in Content Sharing Resources between LWM2M Endpoints and M2M Applications.
- Interworking with full mapping of LWM2M Objects in LWM2M Endpoints to semantically enabled Content Sharing Resources that are utilized by M2M Applications.
- Interworking with one-to-one mapping of LWM2M Objects with oneM2M <mgmtObj> resources

  NOTE: The present document limits Content Sharing Resources to <container> and <contentInstance> resources.

2 References

2.1 Normative references

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

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

- [1] oneM2M TS-0011: “Common Terminology”.
- [2] oneM2M TS-0001: “Function Architecture”.
- [3] OMA-TS-LightweightM2M-V1_0-20150318-D: “Lightweight Machine to Machine Technical Specification”.
- [4] oneM2M TS-0003: “Security Solutions”.
- [5] oneM2M TS-0005: “Management Enablement (OMA)”.
- [6] 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 nonspecific. For specific references, only the cited

version applies. For non-specific references, the latest version of the reference document (including any amendments) applies.

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] oneM2M Drafting Rules (http://member.onem2m.org/Static_pages/Others/Rules_Pages/oneM2M-Drafting-Rules-V1_0.doc
- [i.2] IETF RFC 7252: "Constrained Application Protocol (CoAP)".
- [i.3] IETF RFC 6347: "Datagram Transport Layer Security Version 1.2".
- [i.4] OMA OMA-RD-LightweightM2M-V1_0: "OMA Lightweight Machine to Machine Requirement".
- [i.5] oneM2M TS-0012: "Base Ontology".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in oneM2M TS-0011 [1], oneM2M TS0002 [2] apply. A term defined in the present document takes precedence over the definition of the same term, if any, in oneM2M TS-0011 [1] and oneM2M TS-0001 [2].

3.2 Abbreviations

For the purposes of the present document, the terms and definitions given in oneM2M TS-0011 [1] and the following apply:

```
LWM2M    Lightweight M2M
OMA      Open Mobile Alliance
```

4 Conventions

The key words "Shall", "Shall not", "May", "Need not", "Should", "Should not" in this document are to be interpreted as described in the oneM2M Drafting Rules [i.1].

5 Architecture Model

5.1 Introduction

The architecture model followed in the present document is based on the architecture model in Annex F of oneM2M TS-0001 [2] that describes how interworking

between CSEs and non-oneM2M solutions and protocol using specialized Interworking Proxy application Entities (IPE). The present document describes the LWM2M IPE that supports the following scenarios.

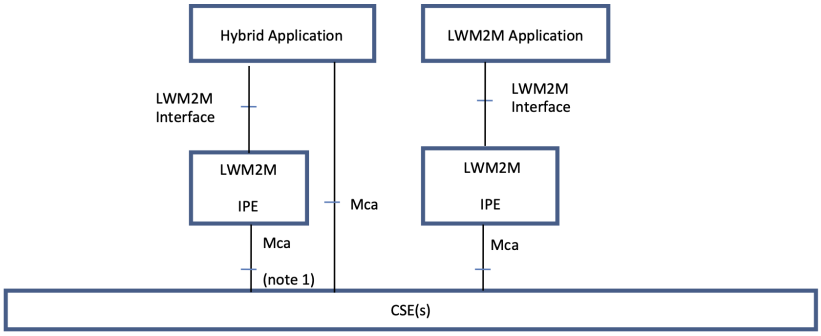

Figure 1: Figure 5.1-1: LWM2M Interworking Scenarios

In the scenarios depicted in Figure 5.1-1, the Hybrid and LWM2M Applications represent applications that implement the LWM2M Client role defined in the LWM2M Protocol [3].

5.2 Reference Model

The LWM2M Interworking reference model utilizes the Functional Architecture's reference model in oneM2M TS0001 [2]; augmenting the oneM2M TS-0001 [2] reference model with capabilities provided by the LWM2M IPE.

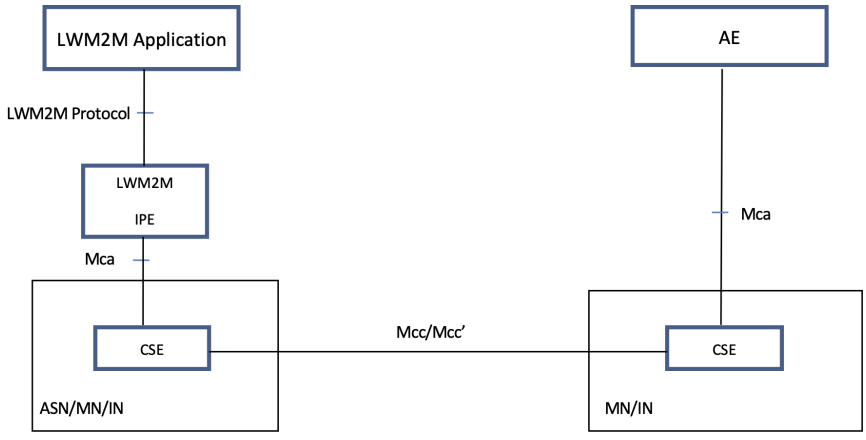

Figure 2: Figure 5.2-1: LWM2M Reference Model

NOTE: The AE in the reference model could be registered with the

same CSE as the LWM2M IPE.

5.3 Types of Interworking

LWM2M IPEs provide the following types of interworking:

1. Interworking using the Content Sharing Resource for transparent transport of encoded LWM2M Objects that are available to AEs as depicted in Figure 5.3-1.
2. Interworking with full mapping of the semantics of LWM2M Objects to semantically enabled resources that are available to AEs as depicted in Figure 5.3-2.
3. Interworking with one-to-one mapping of LWM2M Objects to oneM2M <mgmtObj> resources as depicted in Figure 5.3-3.

While depicted outside the hosting CSE, the Content Sharing Resources are hosted in a CSE (e.g. CSE1).

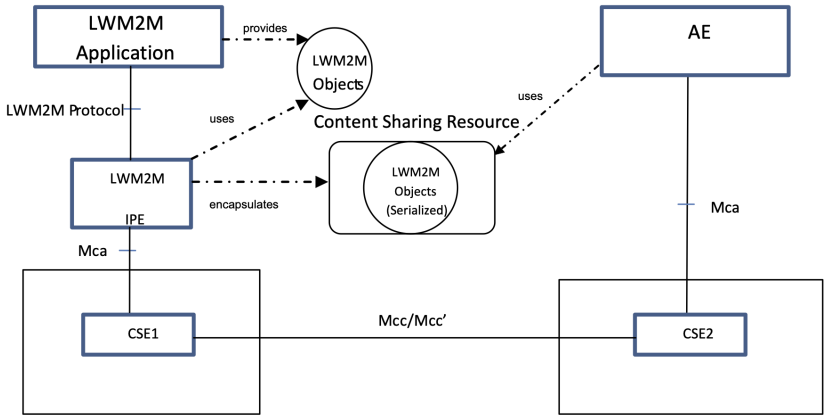

Figure 3: Figure 5.3-1: LWM2M Transparent Interworking Function

In Figure 5.3-1, the LWM2M Objects are provided by the LWM2M Application to the LWM2M IPE using the LWM2M Protocol. The LWM2M IPE then encapsulates the LWM2M Objects in Content Sharing Resources and then hosts the Content Sharing Resources in a CSE using the Mca reference points for use by AEs. The AE accesses the Content Sharing Resource from the CSE that hosts the resource using the Mca reference point. Once the AE receives the Content Sharing Resource, the AE extracts the LWM2M Object from the Content Sharing Resource for the AE's purpose. Clause 7 describes this type of interworking in greater detail.

In Figure 5.3-2, the LWM2M Objects are provided by the LWM2M Application to the LWM2M IPE using the LWM2M Protocol. The LWM2M IPE then interworks the LWM2M Objects into Content Sharing Resources. The Content

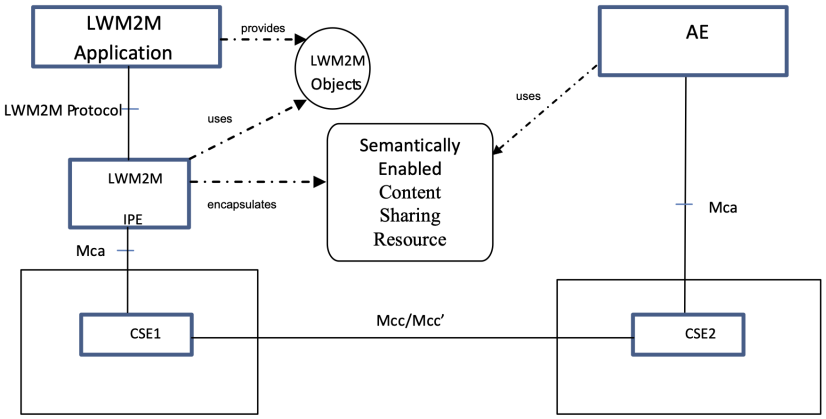

Figure 4: Figure 5.3-2: LWM2M Semantically Enabled Interworking Function

Sharing Resources are based on the oneM2M defined Semantic Ontology. The LWM2M IPE hosts the Content Sharing Resource in a CSE across the Mca reference for use by other AEs. The AE accesses the Content Sharing Resource from the CSE that hosts the resource using the Mca reference point. Once the AE receives the Content Sharing Resource, the AE encodes the information using the Semantic Ontology for the AE's purpose. Clause 8 describes this type of interworking in greater detail.

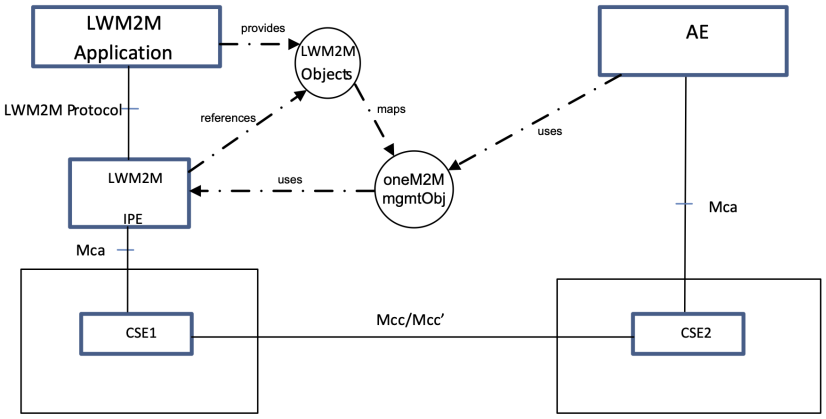

Figure 5: Figure 5.3-3: LWM2M Translation Interworking Function

In Figure 5.3-3, the IPE is provisioned with knowledge of the resource definition for a LWM2M Object and the corresponding oneM2M <mgmtObj> XSD, which shall be generated as described in TS-0005 [5]. The XSD shall map each LWM2M resource of a particular LWM2M Object ID to an [objectAttribute] in

the corresponding oneM2M <mgmtObj>. The ordering of the [objectAttribute] in the XSD shall match the same order of resources defined in the corresponding LWM2M Object. For a given LWM2M Object, the IPE shall create a corresponding <mgmtObj> resource on the CSE and configure the *mgmtSchema* attribute with a URI of the XSD file for that <mgmtObj>. All AEs can then access the <mgmtObj> the same way they access other oneM2M resources. Clause 9 describes this type of interworking in more details.

An instance of a LWM2M IPE shall provide the capability for transparent transport of encapsulated LWM2M Objects as Content Sharing Resources and/or translation of LWM2M Objects as oneM2M semantically enabled Content Sharing Resources or as mapped oneM2M <mgmtObj> resources.

5.4 Composition of the Interworking Proxy Entity

The LWM2M IPE participation in the LWM2M Protocol as described in clause 5 does so in the role of a LWM2M Server to which LWM2M Applications (LWM2M Clients) interact. For each LWM2M Client (Endpoint) that is maintained by the LWM2M Server in the LWM2M IPE, the LWM2M IPE shall instantiate and maintains an instance of a Resource of type <AE>.

6 Architecture Aspects

6.1 Introduction

The LWM2M IPE participation in the LWM2M Protocol as described in clause 5 does so in the role of a LWM2M Server to which LWM2M Applications (LWM2M Clients) interact. As a LWM2M Server, the IPE provides the following Architecture Aspects based on the LWM2M Protocol Aspects described in clause A.2:

- LWM2M Device and Endpoint Lifecycle (Client Registration).
- LWM2M Object Discovery (Client Registration, Device Management and Service Enablement).
- LWM2M Object Transport and Interworking (Device Management and Service Enablement).
- LWM2M Object Subscription and Notification (Information Reporting).
- LWM2M Interworking Proxy Entity Administration.
- LWM2M Client Provisioning (Bootstrap).
- LWM2M Object Security (Device Management and Service Enablement).

6.2 LWM2M Device and Endpoint Lifecycle

6.2.1 Introduction

As the LWM2M IPE discovers LWM2M Endpoints when the LWM2M IPE interacts with the LWM2M Client over the LWM2M protocol's Bootstrap and

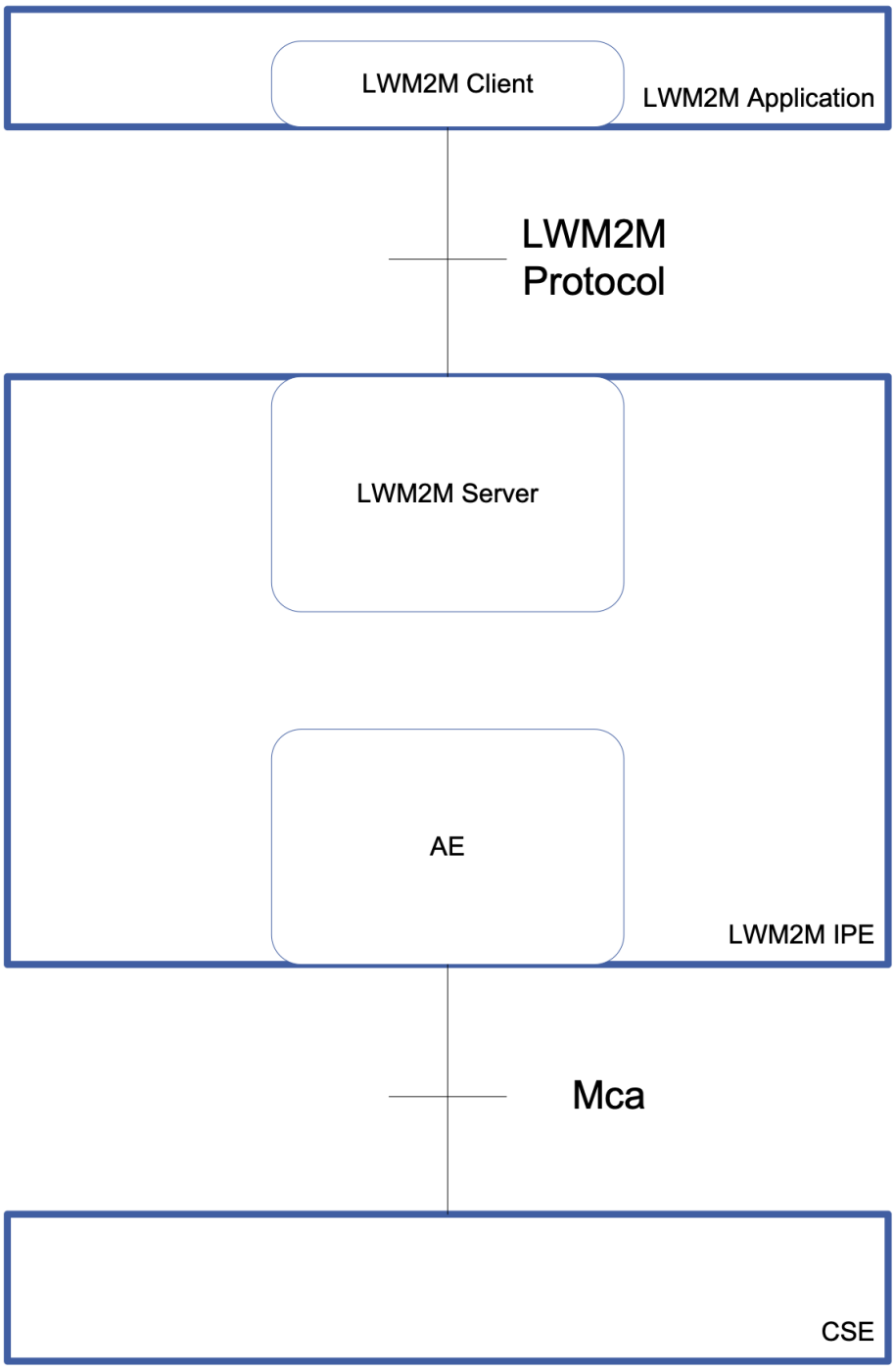

Figure 6: Figure 5.4-1: LWM2M IPE Architecture

Client Registration interfaces, the LWM2M IPE shall maintain the associated resources in the CSE that represents the LWM2M Device and Endpoint.

6.2.2 LWM2M Device and Endpoint Resource Representation

6.2.2.1 Introduction LWM2M Endpoint provides the management and control functions for an M2M Application on a device. As such, the CSE that hosts the M2M Application shall represent the LWM2M Endpoint as a <AE> resource (LWM2M Endpoint <AE> resource). The LWM2M Device that hosts the LWM2M Endpoint shall be represented as a <node> resource.

6.2.2.2 LWM2M Device and Endpoint Resource Identification LWM2M Endpoints are identified by their Endpoint Client Name described in clause 6.2.1 of the LWM2M Technical Specification [3]. The Endpoint Client Name URN without the "urn:" sequence is used as the AE-ID of the associated <AE> resource that represents the LWM2M Client.

In most deployment scenarios, LWM2M Devices host one (1) LWM2M Endpoint. In this scenario the LWM2M Device's <node> resource's M2M-Node-ID should be the same as the LWM2M Endpoint Client Name URN without the "urn:" sequence. When a LWM2M Device host's more than one (>1) LWM2M Endpoint, the determination of the <node> resource's M2M-Node-ID is implementation specific. In all deployment scenarios, the <AE> resource is linked with the <node> resource as described in oneM2M TS-0001 [2].

As the LWM2M Endpoint is represented as an <AE> resource and a LWM2M Object is represented as a Content Sharing Resource in the M2M Service Layer, a reference shall be made between the <AE> resource that represents the LWM2M Endpoint and the Content Sharing Resources which represents the list of LWM2M Objects and Object Instances available in that LWM2M Client.

In order to identify interworked entities hosted in a CSE for the LWM2M technology described in this present document, the <AE> resource that represents the LWM2M Endpoint and the Content Sharing Resources which represent the list of LWM2M Objects and Object Instances available in that LWM2M Client, shall have a Iwked_Technology *labels* attribute set to LWM2M.

For the case where a LWM2M Endpoint is represented as an <AE> resource and LWM2M Objects are represented as <mgmtObj> resources in the M2M Service Layer, each <mgmtObj> resource shall be a child of the <node> resource representing the LWM2M Device that hosts the LWM2M Endpoint. In this case, the *nodeLink* attribute of the <AE> resource representing the LWM2M Endpoint shall reference the <node> resource representing the LWM2M Device.

In addition the <AE> resource uses the Hierarchical and Non-Hierarchical mechanisms for Resource Addressing as defined in clause 9.3.1 of oneM2M TS-0001 [2] where the *resourceName* attribute of the <AE> resource shall be a Endpoint Client Name URN without the "urn:" sequence.

6.2.2.3 LWM2M Endpoint Lifecycle LWM2M Endpoint's are discovered when the LWM2M Client is successfully registers with the LWM2M Server using the LWM2M Register operation. In addition to the LWM2M Register operation, the LWM2M Client can periodically refresh the LWM2M Client's registration with the LWM2M IPE using the LWM2M Update operation. Finally a LWM2M Client can deregister when the LWM2M Client issues a De-register operation to the LWM2M IPE or the LWM2M Client's registration lifetime expires.

The LWM2M Client Registration interface's operations and the registration lifetime expiration event maps to the following operations on the <AE> and <node> resources.

Table 2: Table 6.2.2.3-1: LWM2M Endpoint Lifecycle Translation - Client Registration Interface

LWM2M Operation Client Registration Interface	oneM2M Resource and Operation
Register	create <AE>, create <Node>
Update	update <AE>, update <Node>
De-register	delete <AE>, delete <Node>

Table 3: Table 6.2.2.3-2: LWM2M Endpoint Lifecycle Translation - LWM2M Server Events

LWM2M Server Events	oneM2M Resource and Operation
client lifetime expiration	delete <AE>, delete <Node>, delete <container> resources associated with the <AE> resource, delete <mgmtObj> resources associated with <node> resource.

Table 4: Table 6.2.2.3-3: LWM2M Endpoint Lifecycle Attribute Translation

LWM2M Attributes Client Registration Interface	oneM2M Resource Attribute
Endpoint Client Name	<AE>: AE-ID, resourceName <Node>: M2M-Node-ID when the Device only supports one Endpoint; resourceName
Lifetime	<AE>, <Node>: expirationTime

LWM2M Attributes Client Registration Interface	oneM2M Resource Attribute
LWM2M Version	<AE>, <Node>: labels. Value is "Iwked-Entity-Version:"appended with the value of the LWM2M Version.
Binding Mode	Not Applicable
SMS Number	Not Applicable

Table 5: Table 6.2.2.3-4: LWM2M Endpoint Lifecycle Response Code Translation

LWM2M Errors Client Registration Interface	oneM2M Resource Operation Response
Register	create <AE>, create <Node>
2.01 Created:	2001 Created
4.00 Bad Request	All other codes
4.03 Forbidden	4105 Conflict
Update	update <AE>, update <Node>
2.04 Changed	2004 Changed
4.00 Bad Request	All other codes
4.04 Not Found	4000 Not Found
De-register	delete <AE>, delete <Node>
2.02 Deleted	2002 Deleted
4.04 Not Found	4004 Not Found

6.2.2.4 Configuration of CMDH Policies In the present document, the CMDH Policies associated with the <Node> resource for the AE is implementation specific.

6.2.2.5 Registering a Registered LWM2M Endpoint In the scenario where a LWM2M Client issues a Register operation for an AE that is already created, the LWM2M IPE shall treat the operation as if the LWM2M Client requested a De-Register (Delete <AE> resource) prior this Register operation (Create <AE> resource) as described in Table 6.2.2.3-1. The procedure for the LWM2M Server is defined in clause 5.3.1 of the LWM2M Technical Specification [3].

6.3 LWM2M Object Discovery

6.3.1 Introduction

The LWM2M Client uses the Registration Interface to provide the properties required by the LWM2M Server of the IPE to contact the LWM2M Client

(e.g. Endpoint Name) and to maintain the session between these two LWM2M entities (e.g. Lifetime, Queue Mode).In addition, the LWM2M Client also provides the knowledge of the supported Objects and existing Object Instances across the Registration Interface.

The LWM2M IPE uses the information exchange across this interface to synchronize which LWM2M Objects supported by the LWM2M Endpoint and what is defined in the hosting CSE for the M2M Application representing the LWM2M Endpoint. This clause specifies how discovered LWM2M Objects identifiers are translated to discoverable oneM2M resources along with the associated linkages to other oneM2M resources.

6.3.2 LWM2M Object Identifier Representation

6.3.2.1 Introduction Through the Registration Interface, the LWM2M Client provides the list of supported LWM2M Objects and existing LWM2M Object Instances. Each element of the list is described by its path, which can be the path of an Object or an Object Instance.

For example the LWM2M Client could provide the following list of paths: </1/0>, </1/1>, </2/0>, </2/1>, </3/0>, </4/0>, </5>. This list of paths is a valid list of LWM2M Objects and LWM2M Object Instances in the CoRE Link Format [RFC6690], specifying that LWM2M Objects with OMNA Identifiers 1, 2, 3,4, and 5 are supported. The respective OMNA references are : urn:oma:lwm2m:oma:1, urn:oma:lwm2m:oma:2, urn:oma:lwm2m:oma:3, urn:oma:lwm2m:oma:4, urn:oma:lwm2m:oma:5.

Additionally, information is provided that LWM2M Object 1 (Server Object) and LWM2M Object 2 (Access Control Object) have 2 instances (Instance Identifiers 0 and 1); LWM2M Object 3 (Device Object) and LWM2M Object 4 (Connectivity Monitoring Object) have 1 instance each (0); LWM2M Object 5 is supported but no instance has been created yet for that LWM2M Object.

Optionally other information can be carried by that list as the capability for all the Objects in the LWM2M Client to support:

- an alternate root path (default root path is '/');
- a specific Content-Format (e.g. LWM2M JSON Content-Format).

For discovery of LWM2M Objects by M2M Applications, the properties carried by LWM2M Objects list (i.e. technology, Objects and LWM2M Object Instances Identifiers, optional alternate rootpath, supported Content-Format) shall be translated into the labels attribute of the Content Sharing Resource as separate entries with the following format:

- Iwked-Technology:LWM2M
- Iwked-Entity-Type:Resource Type
- LWM2M-PATH:Resource Root Path (for LWM2M default rootpath is "/").
- Iwked-Entity-ID:Resource Path Object Identifier and Instance Identifier.

- Iwked-Content-Type: Supported Content Format (LWM2M default Supported ContentFormat is LWM2M TLV other can be LWM2M JSON).

For the case where LWM2M Objects are represented as <mgmtObj> resources in the M2M Service Layer, the properties carried by the LWM2M Objects list shall be translated into the labels attribute of the <node> resource using the above format.

For example if the LWM2M Endpoint provided the following LWM2M Objects as part of the Client Registration Interface: </lwm2m>;rt="oma.lwm2m";ct=LWM2M+JSON,</1/0> would translate into a <container> resource with the following entries in the labels attribute: Iwked-Technology:LWM2M Iwked-Entity-Type:"urn:oma:lwm2m:oma:1" LWM2MPTH:"/lwm2m" Iwked-Entity-ID: "/1/0" Iwked-Content-Type:LWM2M+JSON (see note).

> NOTE: LWM2M+JSON is an entry (numerical ID) in the CoAP Content-Format Registry representing the media-type "application/vnd.oma.lwm2m+json" used in LWM2M TS 1.0 enabler and currently engaged in the IANA registration process.

The CoAP Resource Type may also be used as the semantic ontology of the <container> resource by inserting this value in the ontologyRef attribute of the <container> or other translated oneM2M resource.

For the case where LWM2M Objects are represented as <mgmtObj> resources in the M2M Service Layer, the IPE shall use information carried in the LWM2M Objects list to configure not only the *labels* and *description* attributes but also the *objectID* and *objectPath* attributes of the <mgmtObj> resources since *objectID* and *objectPath* can also be helpful for discovery of the supported LWM2M Objects. For the case that 1:1 mapping of LWM2M Object to oneM2M <mgmtObj> is desired, the *objectIDs* attribute shall contain the URN of the corresponding LWM2M Object and the *mgmtSchema* attribute shall contain a URI of the schema file for the new <mgmtObj> specialization as outlined in Clause 6.7 of TS-0005 [4].

LWM2M Object Resources are identified by their URI within the context of the LWM2M Endpoint described in clause 6.2.1 of the LWM2M Technical Specification [3].

As the LWM2M Endpoint is represented as an <AE> resource and a LWM2M Object is represented as a oneM2M resource in the M2M Service Layer, a reference shall be made between the <AE> resource that represents the LWM2M Endpoint and the oneM2M resource which represent the list of LWM2M Objects and Object Instances available in the LWM2M Client.

For the case where a LWM2M Object is represented as a <mgmtObj> resource, this reference is already provided by the AE's nodeLink attribute.

In addition, oneM2M resources that represents the LWM2M Object or LWM2M Object Instance uses the Hierarchical and Non-Hierarchical mechanisms for

Resource Addressing as defined in clause 9.3.1 of oneM2M TS0001 [2] where the *resourceName* attribute of the Content Sharing or oneM2M resource shall be the value of the LWM2MURI. All characters that are not in the unreserved character set defined in clause 2.3 of the of IETF RFC 3986 [6] shall be percent encoded as defined in clause 2.1 of the same IETF RFC, specifically the forward slash (/) character.

For example if the LWM2MURI is "/1/0 and the LWM2MPTH is"/" then the *resourceName* attribute of the oneM2M resource could be "%2F1%2F0".

For the case where <mgmtObj> resources are used, the "/1/0" LWM2MURI is mapped to the <mgmtObj>'s objectPath attribute. All characters that are not in the unreserved character set shall be percent encoded as well.

6.3.2.2 Void

6.3.2.3 LWM2M Object Lifecycle LWM2M Endpoint's are discovered when the LWM2M Client is successfully registers with the LWM2M Server using the LWM2M Register operation. In addition to the LWM2M Register operation, the LWM2M Client can periodically refresh the LWM2M Client's registration with the LWM2M IPE using the LWM2M Update operation. Finally a LWM2M Client can deregister when the LWM2M Client issues a De-register operation to the LWM2M IPE or the LWM2M Client's registration lifetime expires.

The LWM2M Client Registration interface's operations and the registration lifetime expiration event maps to the following operations on the resource.

Table 6: Table 6.3.2.3-1: LWM2M Object Lifecycle Translation - Client Registration Interface

LWM2M Operation Client Registration Interface	oneM2M Resource and Operation
Register	create <container>, oneM2M resource
Update	update <container> , delete <container>, oneM2M resource
De-register	delete <container>, oneM2M resource

Table 7: Table 6.3.2.3-2: LWM2M Object Lifecycle Translation - LWM2M Server Events

LWM2M Server Events	oneM2M Resource and Operation
Client lifetime expiration	delete <container>, oneM2M resource

Table 8: Table 6.3.2.3-3: LWM2M Object Lifecycle Attribute Translation

LWM2M Attributes Client Registration Interface	oneM2M Resource Attribute
Endpoint Client Name	Not Applicable
Resource Links	<container>, oneM2M resource resourceName
Lifetime	<container> , oneM2M resource expirationTime
LWM2M Version	Not Applicable
Binding Mode	Not Applicable
SMS Number	Not Applicable

Table 9: Table 6.2.2.3-4: LWM2M Object Lifecycle Response Code Translation

LWM2M Errors Client Registration Interface	oneM2M Resource Operation Response
Register	create <container>
2.01 Created:	2001 Created
4.00 Bad Request	All other codes
4.03 Forbidden	4105 Conflict
Update	update <container>
2.04 Changed	2004 Changed
4.00 Bad Request	All other codes
4.04 Not Found	4000 Not Found
De-register	delete <container>
2.02 Deleted	2002 Deleted
4.04 Not Found	4004 Not Found

6.4 LWM2M Object Transport and Interworking

6.4.1 Introduction

When an oneM2M request is addressed from a CSE/AE to a hosting CSE containing the representation of a LWM2M Client, the oneM2M response to the Originator of the request is returned through the cooperation of the hosting CSE and the IPE.

The LWM2M Client uses the Device Management & Service Enablement interface to provide the capabilities for the LWM2M Server of the IPE to access LWM2M Objects, Objects Instances and Resources available from the LWM2M Client.

A hosting CSE maintains a representation of the LWM2M Data Model of LWM2M Object, Object Instance or Resources as instances of oneM2M resource types.

These Content Sharing Resources are instantiated and registered as described in clause 6.3 allowing oneM2M AEs and CSEs to exchange data with LWM2M Clients.

In reference to clause 6.3, at the end of the registration phase all declared LWM2M Object Instances and LWM2M Objects are associated to a Content Sharing Resource created with the resourceName attribute set accordingly to the proper LWM2M Object Instance path (e.g. /9/1) or to the LWM2M Object path (e.g. /9).

6.4.2 LWM2M Interworking Mechanisms

6.4.2.1 Introduction Cooperation between IPE and the oneM2M hosting CSE requires efficient mechanisms to maintain the latest state of the targeted LWM2M Objects, Object Instances and Resources. These mechanisms include data synchronization between the IPE and hosting CSE.

Data synchronization relies on the oneM2M Subscription/Notification and LWM2M Observation/Notification mechanisms. For automated data synchronization between the IPE and hosting CSE to be achieved, the solution shall be granular enough to allow data synchronization for each LWM2M Object Instance.

Access Control mechanisms relies on an interworking between oneM2M and LWM2M Access Control Policies.

LWM2M and oneM2M mechanisms used to achieve Data Synchronization and Access Control is specified in more details in clauses 6.5 and 6.6.

These following sub-clauses specify the sequences of operations involved for each type of supported oneM2M requests following the general procedures specified in clause 10 (CREATE,RETRIEVE, DELETE) as used within the context of the interworking for the present document

6.4.2.2 Relevant Interworked Resource Settings A LWM2M Object Instance that is represented in oneM2M as a <container> resource has 2 direct children resource types: a <subscription> resource and a <contentInstance> resource when used with a <container> resource.

A LWM2M Object Instance that is represented in oneM2M as a <mgmtObj> resource may have 1 direct child resource of type <subscription>.

For supporting the LWM2M interworking process, a few attributes for the <container> resource and the <subscription> resource shall have a specified set of parameters.

a) Attributes of <container> resource

Table 10: Table 6.4.2.2-1: <container> resource - Relevant Interworked Attributes

Attributes of *<container>* resource	Value
accessControlPolicyIDs	ACP set (see Clause 6.6)
maxNrOfInstances	1

b) Child resource types of *<container>* resource

Table 11: Table 6.4.2.2-2: <container> resource - Relevant Child resource types

Child resources of *<container>* resource
<contentInstance> resource
<subscription> resource

c) Attributes of *<subscription>* resource

Table 12: Table 6.4.2.2-3: <subscription> resource - Relevant Interworked Attributes

Attributes of *<subscription>* *resource*	Description / Value
notificationURI	IPE URI
eventType	B. Deletion of the subscribed-to resource; C. Creation of a direct child of the subscribed-to resource; E. An attempt to retrieve a <contentInstance> direct-child-resource of a subscribed-to <container> resource is performed while this child 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".

6.4.2.3 oneM2M RETRIEVE Procedure This procedure describes the retrieval of a resource using the oneM2M RETRIEVE request. The information contained within the resource is related to the LWM2M Objects Instances that are interworked through the IPE. This clause shall be treated in conformance

with the RETRIEVE Procedure specified in oneM2M TS-0001 [2], clauses 10.1.2 and 10.2.4.2.

The 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 according to the procedures for the type of interworking (e.g. Transparent, Semantically Enabled, or Management Objects) as described in clause 7, 8, and 9, otherwise an error response shall be returned to the Originator.

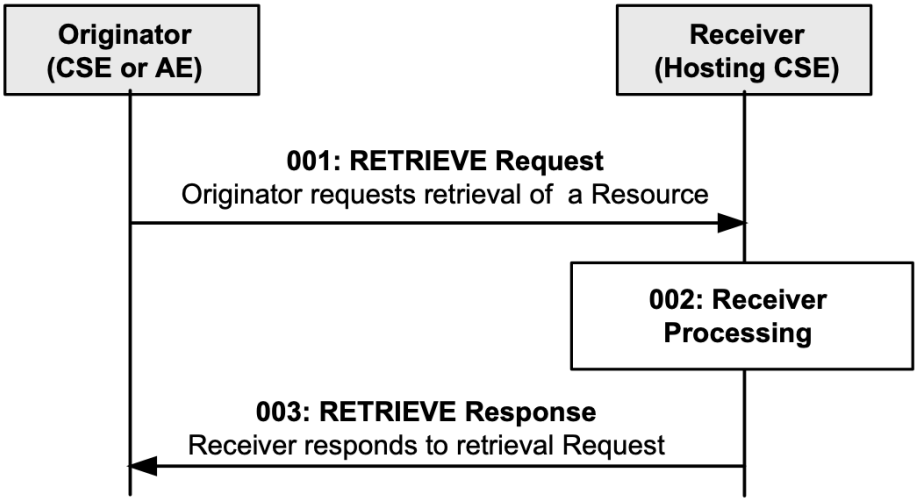

Figure 7: Figure 6.4.2.3-1: Procedure for Retrieving a Resource (oneM2M TS-0001 [2], clause 10)

The target of the request is a <container> or <mgmtObj> resource, the ResultContent and FilterCriteria parameters of the request (oneM2M TS0001 [2], clause 8.1.2) specify the nature of the information to retrieve.

Table 13: Table 6.4.2.3-1: Effect of Request Parameters on Retrieval Request

Request parameter : ResultContent	Request parameter : FilterCriteria	Effect
child-resources	resourceType = contentInstance	Retrieval of LWM2M Object Instance
attributes (default)	labels, attribute	Metadata retrieval of LWM2M Object Instance.

Specific steps of the Receiver Processing according to the interworking process shall be followed:

Step 001: Find and verify the targeted <container> or <mgmtObj> resource of the request : the resourceName corresponds to clause 6.3.2.

Step 002: Using the Hosting CSE Access Control mechanisms, check for Access Control Policy attached to the <container> or <mgmtObj> resource of the request

Step 003: On successful validation of the Access Control Policy, proceed to fetch the requested information:

a) If "ResultContent" is "attributes", jump to Step 4.

b) If "ResultContent" is "child-resources" and "FilterCriteria" is "contentInstance", the associated <contenInstance> resource of the targeted container is considered.

- **Step 003.1:** if the <contentInstance> resource is obsolete or its reference is not assigned, an event for Retrieval attempt (eventType 'E') is triggered to the Entity that subscribed to the event (i.e. IPE); as a Blocking Procedure, the Hosting CSE shall monitor the arrival of the new data or decide to report a timeout error in jumping to Step 004:
  - **Step 003.1.1:** On receiving the event of type 'E' (eventType 'E') the IPE performs a LWM2M READ request on the Object Instance of the targeted LWM2M Client.
    - **Step 003.1.2:** Once the targeted Object Instance is available to IPE, the IPE creates and populates with that data the <contentInstance> child-resource of the requested <container> resource.
- **Step 003.2:** the up-to-date information is available in the <contentInstance> resource.

Step 004: The Hosting CSE returns the appropriate response back to the Originator (e.g. Errors, or Data).

> NOTE: As an OBSERVATION has been set up on the targeted LWM2M Object Instance, the automatic synchronization between the Object Instance and its representation in the Hosting CSE is performed. Further oneM2M accesses to the resource should be simplified in minimizing impact of Step003 (up-to-date data already present from the Hosting CSE resources).

General Exceptions:

1. The targeted resource/attribute in ***To*** parameter does not exist. The Receiver shall respond with an error.
2. The Originator does not have privileges to retrieve information stored in the resource on the Receiver. The Receiver shall respond with an error.
3. A timer has expired. The Receiver shall respond with an error.

6.4.2.4 oneM2M CREATE Procedure This procedure describes the update of a LWM2M Object Instance in a LWM2M Client using the oneM2M CREATE request The information contained in the request via the Content parameter (oneM2M TS-0001 [2], clause 8.2.1) will be used to update an Object Instance in the LWM2M context.

This clause shall be treated in conformance with the CREATE Procedure specified in oneM2M TS-0001 [2], clauses 10.1.2 and 10.2.4.1.

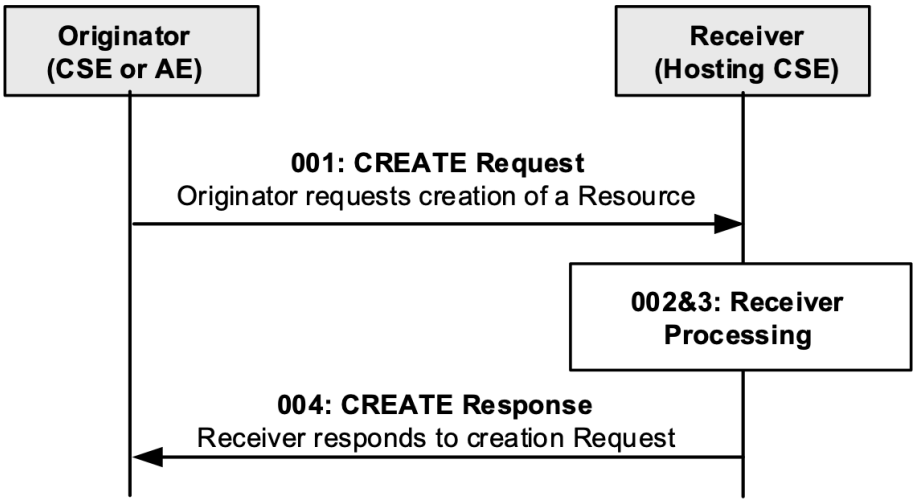

Figure 8: Figure 6.4.2.4-1: Procedure for Creating a Resource (oneM2M TS-0001 [2], clause 10)

The target of the request is a <container> or <node> resource, the ResourceType parameters of the request (oneM2M TS-0001 [2], clause 8.1.2) specifies the type of the resource to create.

Table 14: Table 6.4.2.4-1: Effect of Request Parameters on Create Request

Request parameter :ResourceType	Effect
ContentInstance	Replacement of the <latest> resource by a new one
MgmtObj	Creation of new <mgmtObj> resource or appropriate error message if one already exist

In LWM2M Interworking context, if there is already an existing <contentInstance> resource, creating a new one shall violates the policy defined in the

parent <container> resource regarding the maxNrOfInstances which shall be set to 1. Then the oldest <contentInstance> resources shall be removed from the <container> to enable the creation of the new <contentInstance> resource.

In any case, a notification is sent to the IPE, which subscribed to the parent <container> or <node> resource with the eventType 'C' (Creation of a direct child of the subscribed-to resource). The IPE shall use that notification to update the LWM2M resource model with the new data received ("Content" parameter of the request).

Specific steps of the Receiver Processing according to the interworking process shall be as followed:

Step 001: Find and verify the <container> or <node> resource of the request : the resourceName corresponds to clause 6.3.2.

Step 002: Using the Hosting CSE Access Control mechanisms, check for Access Control Policy attached to the <container> or <node> resource of the request.

Step 003: On unsuccessful validation of the Access Control Policy, jump to step 4:

- **Step 003.1:** When according to the request, a <contentInstance> or <mgmtObj> resource is created, an event for Child Creation (eventType 'C') is triggered to the Entity that subscribed to that event (i.e. IPE).
- **Step 003.2:** On receiving the event of type 'C' the IPE - via the Mca reference point - get the data from the created <contentInstance> or <mgmtObj> resource and propagates the updated data to the related Object Instance in the LWM2M Client.

Step 004: The Hosting CSE returns the appropriate response back to the Originator (e.g. Acknowledgment, Errors).

General Exceptions:

1. The Originator does not have the privileges to create a resource on the Receiver, the Receiver shall respond with an error.
2. The resource with the specified name (if provided) already exists at the Receiver, the Receiver shall response with an error.
3. The provided information in ***Content*** is not accepted by the Receiver (e.g. missing mandatory parameter), the Receiver shall respond with an error.

6.4.2.5 oneM2M UPDATE Procedure This procedure describes the update of a LWM2M Object Instance or Resource in a LWM2M Client using the oneM2M UPDATE request. The information contained in the request via the Content parameter (TS-0001 Clause 8.1.2) shall be used to update an Object Instance or Resource in the LWM2M context.

This clause shall be treated in conformance with the UPDATE Procedure specified in TS-0001 [2] clause 10.1.3 and 10.2.8.4.

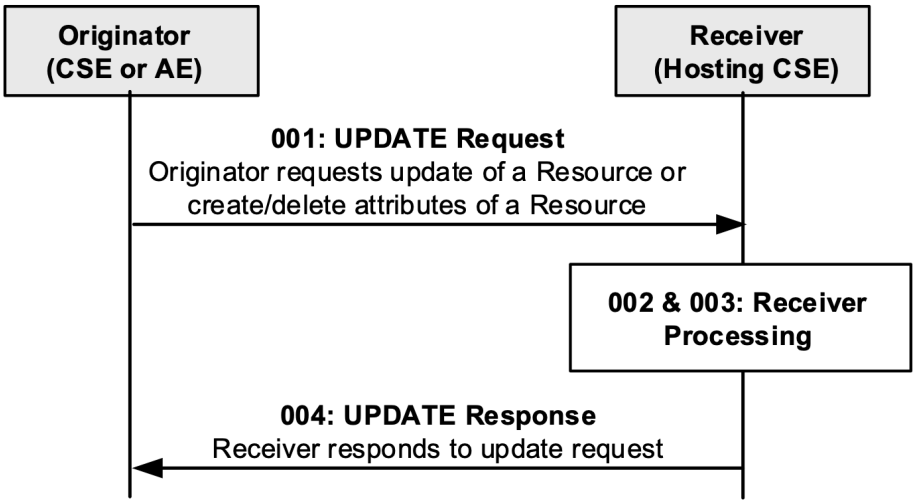

Figure 9: Figure 6.4.2.5-1: Procedure for UPDATing a Resource (TS-0001 Clause 10)

The target of the request is the <mgmtObj> resource itself.

Specific steps of the Receiver Processing according to the interworking process shall be as followed:

Step 001: Find and verify the <mgmtObj> resource of the request : the resourceName corresponds to clause 6.3.2

Step 002: Using the Hosting CSE Access Control mechanisms, check for Access Control Policy attached to the <mgmtObj> resource of the request.

Step 003: On unsuccessful validation of the Access Control Policy, jump to step 4

Step 003.1: When according to the request, a <mgmtObj> attribute is updated, an event for update to attributes (eventType 'A') is triggered to the Entity that subscribed to that event (i.e. IPE).

Step 003.2 : On receiving the event of type 'A', the IPE - via the Mca reference point - gets the data from the attribute and propagates the updated data to the related Object Instance or Resource in the LWM2M Client.

Step 004: The Hosting CSE returns the appropriate response back to the Originator (e.g. Acknowledgment, Errors)

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 modify the resource, create attributes or delete attributes on the Receiver. The Receiver responds with error.
3. The provided information in the ***Content*** is not accepted by the Receiver. The Receiver responds with error.

6.4.2.6 oneM2M DELETE Procedure This procedure describes the removal of a LWM2M Object Instance within a LWM2M Client using the oneM2M DELETE request.

This clause shall be treated in conformance with the DELETE Procedure specified in oneM2M TS-0001 [2], clauses 10.1.4 and 10.2.4.4.

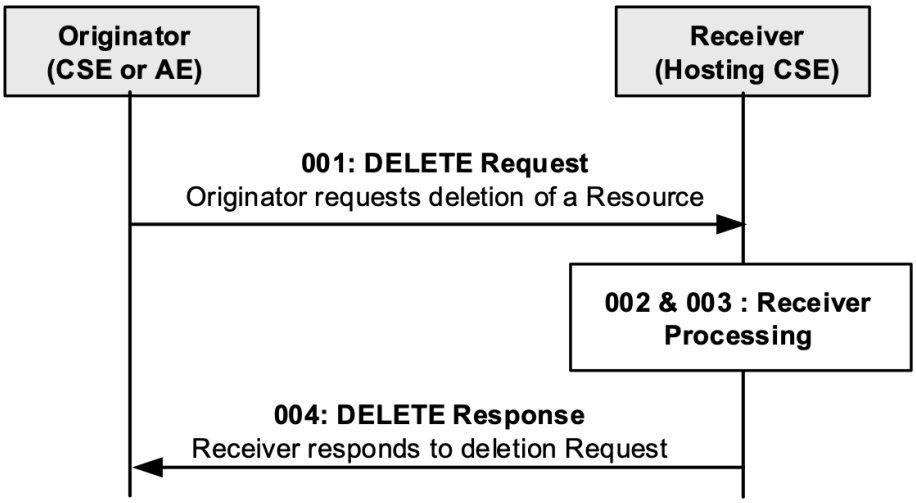

Figure 10: Figure 6.4.2.6-1: Procedure for Deleting a resource (oneM2M TS-0001 [2], clause 10)

Specific steps of the Receiver Processing according to the interworking process shall be as followed:

Step 001: Find and verify the <container> or <mgmtObj> resource of the request : the resourceName corresponds to clause 6.3.2.

Step 002: Using the Hosting CSE Access Control mechanisms, check for Access Control Policy attached to the <container> or <mgmtObj> resource of the request.

Step 003: On unsuccessful validation of the Access Control Policy, jump to step 4:

- **Step 003.1:** When the <container> or <mgmtObj> resource is deleted an event (eventType 'B': Deletion of subscribed-to resource) is triggered to the Entity that subscribed to that event (i.e. IPE).
- **Step 003.2:** On receiving the event type 'B', the IPE requests the LWM2M Client to delete the Object Instance related to the <container> or <mgmtObj> resource.
- **Step 003.3:** On Object Instance deletion, the LWM2M Client performs a De-Registration operation to the IPE.
- **Step 003.4:** Via the Mca reference point, the IPE communicates to the Hosting CSE the status of this DeRegistration.

Step 004: The Hosting CSE returns the appropriate response back to the Originator (e.g. Success, Errors).

6.4.3 oneM2M Resource Operation Responses

Table 15: Table 6.4.3-1: LWM2M Response Codes to oneM2M Resource Operation Response Codes

LWM2M Client Response Codes on Device Management & Service Enablement interface	oneM2M Resource Operation Response
Create	
2.01 Created:	2001 Created
4.00 Bad Request	4000
4.01 Unauthorized	4103
4.04 Not Found	4004
Read	
2.05 Content:	2000 OK
4.01 Unauthorized	4103
4.04 Not Found	4004
4.05 Method Not Allowed	4005
Write	
2.04 Changed	2004 Changed
4.00 Bad Request	4000
4.01 Unauthorized	4103
4.04 Not Found	4004
4.05 Method Not Allowed	4005
	4102

Delete	
2.02 Deleted	2002 deleted
4.00 Bad Request	4103
4.01 Unauthorized	4103
4.04 Not Found	4004
4.05 Method Not Allowed	4005

6.5 LWM2M Object Subscription and Notification

6.5.1 Introduction

The LWM2M Server uses the Information Reporting Interface to provide the capabilities for a LWM2M Server to subscribe to changes to the LWM2M Objects, LWM2M Object instances and the associated LWM2M Object's resources. Likewise the LWM2M Client uses the Information Reporting Interface to notify subscribed LWM2M Server's when the LWM2M Object, LWM2M Object instance and/or LWM2M Object's resources change and to cancel the subscription on LWM2M Objects, LWM2M Object instances and the associated LWM2M Object's resources.

The LWM2M Server uses the Device Management & Service Enablement Interface to set the notification criteria for a subscription.

The oneM2M Subscription capabilities permit subscription changes to an oneM2M resource's attributes and its direct child resources. Likewise, the oneM2M Notification capabilities include a rich set of criteria for when a subscribed-to oneM2M resource is notified of a change.

6.5.2 LWM2M Subscription Procedure

The LWM2M IPE interworks the oneM2M resource's <subscription> child resource with the corresponding LWM2M Object using the oneM2M <subscription> resource's attributes and the corresponding LWM2M Object resource's Notification class Attributes.

> Note: Each LWM2M Object resource has an associated set of Notification class Attributes that are used for defining the applicable subscription and notification criteria.

When the LWM2M IPE creates a oneM2M Content Sharing Resource, the LWM2M IPE creates a subscription on the Content Sharing Resource to be notified whenever the oneM2M resource's subscription attribute is changed by setting the <subscription> resource's attributes as follows.

Table 16: Table 6.5.2-1: LWM2M Subscription Procedure - <subscription> resource

Attributes of *<subscription>*	Description
accessControlPolicyIDs	Link a <accessControlPolicy> resource with the privileges: accessControlOriginator originatorID set to the LWM2M IPE AE's AE-ID accessControlOperations: Set to RETRIEVE, CREATE, UPDATE, DELETE, DISCOVER, NOTIFY
pendingNotification	Set to "sendLatest"
latestNotify	Set to "latest".
notificationContentType	Set to "resource"
<schedule>	Set to immediate notification

Whenever another AE or CSE creates or deletes a subscription to the <container> resource, the LWM2M IPE shall be notified of the change and shall perform the following steps:

Step 001: Find the associated LWM2M Object or Object Instance for notification's subscriptionReference.

Step 002: If the oneM2M notification indicates a subscription deletion:

- **Step 002a:** If the associated LWM2M Object or Object Instance has an outstanding Observation request from the LWM2M IPE then issue the LWM2M Cancel Observation operation.

Step 003: If the oneM2M notification indicates a subscription creation:

- **Step 003a:** If the associated LWM2M Object or Object Instance does not have an outstanding Observation request from the LWM2M IPE then:
  - **Step 003a001:** Retrieve the Parent resource of the <subscription> resource from the notification's subscriptionReference.
  - **Step 003a002:** Determine the LWM2M Notification class Attributes to set from the set of subscriptions of the Parent resource using the <schedule> resource associated with each of the Parent resource's subscriptions.
  - **Step 003a003:** Issue the LWM2M Observe operation with the LWM2M Notification class attributes.

General Exceptions: The processing for recovery of a failed LWM2M Cancel Observation or Observation operation is vendor specific.

Table 17: Table 6.5.2-2: LWM2M Subscription Procedure - Information Reporting Interface

LWM2M Operation Information Reporting Interface	oneM2M Resource and Operation
Observe	NOTIFY (m2m:notification subscriptionDeletion=false)
Cancel Observation	NOTIFY (m2m:notification subscriptionDeletion=true)

Table 18: Table 6.5.2-3: LWM2M Subscription Procedure Attribute Translation

LWM2M Operation DM and SE Interface Notification class Attributes	oneM2M <subscription> Attribute
Minimum Period	<schedule> resource
Maximum Period	<schedule> resource
Greater Than	Not Applicable
Less Than	Not Applicable
Step	Not Applicable

6.5.3 LWM2M Notification Procedure

When the LWM2M IPE is notified by the LWM2M Client of a change in a LWM2M Object or Object Instance the LWM2M IPE creates a new <contentInstance> for the associated <container> resource according to the procedures for the type of interworking (e.g. Transparent, Semantic) as described in clause 7 or 8.

For the case where LWM2M Objects are represented as <mgmtObj> resources in the M2M Service Layer and when the LWM2M IPE is notified by the LWM2M Client of a change in a LWM2M Object or Object Instance, the LWM2M IPE updates the corresponding <mgmtObj> resource according to the procedures described in clause 9.

6.6 LWM2M Object Security

6.6.1 Introduction

OMA-LWM2M and oneM2M Access Control Policies shall collaborate in order to assure the interworked resources are accessible according to the oneM2M Authorisation Procedure specified in clause 11.3.4 (M2M Authorization Procedure) of oneM2M TS-0001 [2] and clause 7 (Authorization) of oneM2M TS-0003 [5].

6.6.2 LWM2M Interworking Access Control Policy

The oneM2M Access Control Policy mechanisms specified in clause 7 of oneM2M TS-0003 [5], shall be used to check and validate the parameters of a request message against the ACPs (<accessControlPolicy> resources) which have been assigned to the accessed resource.

In order to assure a proper LWM2M Interworking with oneM2M, the IPE shall setup the hosting CSE by:

1. providing a mandatory set of <accessControlPolicy> (ACPs) resources
2. assigning a proper set of ACPs to the accessControlPolicyIDs attribute of each <container> resource allocated during the CSE registration phase (clause 6.3 LWM2M Object Discovery)

The process for provisioning the IPE in order to perform such a setup is described in clause 6.6.3 "IPE and Object Security provisioning" of the present document.

In addition, the Access Control Policy mechanisms specified in clause 7 of oneM2M TS-0003 [5] are fully applicable in this LWM2M interworking context.

6.6.3 IPE and Object Security provisioning

In order to provide oneM2M information specified in the clause 6.6.2 (set of <accessControlPolicy> (ACPs) resources, assignment of accessControlPolicyIDs), the LWM2M IPE shall be supplied by information such as:

- a list of oneM2M originators and their associated Access Control Rules likely to be exercised on the Hosting CSE resources
- a list of oneM2M originators likely to contact the LWM2M Clients with the associated set of authorized operations

In combining such an information with the Access Control Policy specified in a given LWM2M Client (clause 6.8 LWM2M Client Provisioning) the LWM2M IPE shall be able to provide to the Hosting CSE, the oneM2M Access Control Policy materials needed for properly registering LWM2M Objects representation. In the current release of this Specification, this procedure of how the Access Control Policy materials are provided is implementation specific.

6.7 LWM2M IPE Administration and Maintenance

6.7.1 Introduction

The IPE described in clause 5.4 is comprised functionality that includes the LWM2M Server and the IPE's AE. This clause describes the administration and maintenance of these functional elements.

6.7.2 Administration and Maintenance of the LWM2M Server Functionality

6.7.2.1 Introduction The LWM2M IPE provides the functionality that plays the role of the LWM2M Server in order to communicate with LWM2M Clients.

In order for communication to be established information needs to be provisioned into the LWM2M Client and LWM2M Server where the following artefacts are necessary to be established for the LWM2M Server:

- LWM2M Server and Client Credentials.
- LWM2M Access control lists.

In addition, the LWM2M Server maintains information about each LWM2M Client and has actions that are used to maintain the LWM2M Server.

These aspects of the LWM2M Server are further described in clause E.2 of the LWM2M Server resource [3].

The mechanisms used to administer and maintain the LWM2M Server functionality within the LWM2M IPE is out of scope of the present document.

6.7.2.2 LWM2M Server Maintenance The LWM2M Server maintains a set of LWM2M Server account information for each LWM2M Client that allows the LWM2M Server to access and communicate with LWM2M Client. These are:

- LWM2M Server identifier associated with and assigned by the LWM2M Client (LWM2M Server identifier, registration lifetime).
- Policies for default observation behavior.

The following actions can also be performed that affects the state of the LWM2M Server's interaction with the LWM2M Client:

- On-demand request for the LWM2M Client to update its registration.

6.7.3 Maintenance of the LWM2M IPE AE Context

6.7.3.1 Introduction The LWM2M IPE AE maintains information related to the operational context of the LWM2M IPE AE. Specifically the following elements are maintained for the LWM2M IPE AE:

- List of currently registered LWM2M Endpoints.
- Configuration of the Interworking Functions to be used for the LWM2M Objects and Object Instances.

6.7.3.2 LWM2M Endpoint List Whenever an LWM2M Endpoint <AE> resource is created, updated or deleted as described in clause 6.2, the LWM2M IPE also manages the list of LWM2M Endpoint <AE> resources using a oneM2M <group> resource.

The oneM2M <group> resource's lifecycle is linked to the LWM2M IPE <AE> resource's lifecycle.

Table 19: Table 6.7.3.2-1: LWM2M IPE <AE> resource - Group Lifecycle

LWM2M IPE <AE> resource Operation	oneM2M Resource and Operation
create	create <group>. The group resourceName is the AE-ID of the LWM2M IPE <AE>.resource
update	update <group>
delete	delete <group>

The LWM2M Endpoint <AE> resources' lifecycle operation maps to the following operations on the oneM2M <group> resource.

Table 20: Table 6.7.3.2-2: LWM2M Endpoint <AE> resource - Group member Lifecycle

LWM2M Endpoint <AE> resource Operation	oneM2M Resource and Operation
Create	update <group> (add member)
Delete	update <group> (delete member)

6.7.3.3 Configuration of Interworking Functions Clause 5.3 describes the types of interworking functions as Transparent Interworking Function and Semantically Enabled Interworking Function. An IPE provides the capability to perform one or both types of interworking functions. The granularity (e.g. Object/Object Instance, IPE) that is used to define which interworking function(s) to use is implementation specific.

6.8 LWM2M Client Provisioning (Bootstrap)

This present document makes assumptions that the LWM2M Clients and LWM2M Servers functionality of the IPE have been provisioned with the proper LWM2M credential materials in accordance to the LWM2M specification (A.5.3.1. Bootstrap) in order to securely communicate between the LWM2M Client and LWM2M Server.

Additionally, a LWM2M Client connected to a LWM2M IPE, should be provisioned with the LWM2M Access Control Policy information associated to the Object Instances contained in the LWM2M Client as described in clause 6.6 LWM2M Object Security. Additionally, in current release of this present

document, the LWM2M Server role of the LWM2M IPE does not contain the LWM2M Bootstrap Server capability, consequently the LWM2M Client provisioning operations shall be part of an out-of-band process.

7 Transparent Interworking Function

7.1 Introduction

Clause 5.3 introduced the Transparent Interworking function as depicted in Figure 5.3-1. This clause specifies the mappings of the attributes of the <contentInstance> resource for a <container> resource in order to allow an AE that uses the Content Sharing Resource to understand that the Content Sharing Resource has an encapsulated LWM2M Object or Object Instance.

7.2 Attribute Mapping for the <contentInstance> Resources

When an AE accesses a <contentInstance> resource, the AE needs to know that the <contentInstance> resource encapsulates a LWM2M Object or Object Instance as well as how the LWM2M Object or Object Instance is encoded.

Table 21: Table 7.2-1: Transparent Interworking Function Mapping

Interworking Function Mapping	oneM2M Resource Attribute
Indication that a LWM2M Object or Object Instance is encapsulated in the <contentInstance> resource with the content type and encoding of the LWM2M Object or Object Instance.	<contentInstance> resource: labels. Value is "LWM2M-Object-Encapsulation"
The content type of the LWM2M Object or Object Instance based on the Content-Type option NOTE: The LWM2M Technical Specification [3] defines the value to be used for the [encoding] if the Content-Type option is not present.	<contentInstance>: contentInfo. Possible contentInfo values are translated from the LWM2M Content-Type option (see note).

8 Semantically Enabled Interworking Function (Informative)

8.1 Introduction

Clause 5.3 introduced the Semantically Enabled Interworking function as depicted in Figure 5.3-2. This clause specifies how LWM2M Objects and their associated LWM2M Resources are organized as <container> resources in order for values associated with the LWM2M Resources be translated into <contentInstance> resources. In addition, this clause specifies the mapping of Content Sharing Resources the oneM2M Base Ontology [i.5].

8.2 Organization of Semantically Enabled Content Sharing Resources

8.2.1 Introduction

Semantically enabled Content Sharing Resources represent the structure and content of LWM2M Objects and Object Instances by translating LWM2M Objects, Object Instances and their LWM2M Resources and LWM2M Resource Instances into a hierarchy of Content Sharing Resources using the Content Sharing Resource's parent-child relationship described in oneM2M TS-0001 [2]. In addition, the LWM2M Resources values are contained within the <contentInstance> resource for <container> resources.

When the LWM2M Resource is of type LWM2M Object Link, the <contentInstance> resource that represents the LWM2M Resource is used to represent the LWM2M Object Link by assigning the destination of the LWM2M Object Link reference to another LWM2M Object's Content Sharing Resource. The reference is assigned using the <contentInstance> resource's contentRef attribute where the name of the attribute is "ObjectLink" and the value of the attribute is the URI of the destinaton Content Sharing Resource.

8.2.2 Lifecycle of Semantically Enabled Content Sharing Resources

Clauses 6.3 and 6.4 describe how LWM2M Objects and Object Instances are discovered and instantiated. The Semantic Interworking function uses these procedures for instantiation of the Content Sharing Resource for the LWM2M Objects and Object Instances.

The Content Sharing Resources for LWM2M Resources and Resource Instances are created as child resources of the parent Content Sharing Resource when the LWM2M Object and Object Instance are created. Likewise these child Content Sharing Resources are deleted when the parent Content Sharing Resource is deleted.

Creation, update or deletion of one or more <contentInstance> resources for the LWM2M Resource or Resource Instances that are not caused by the creation

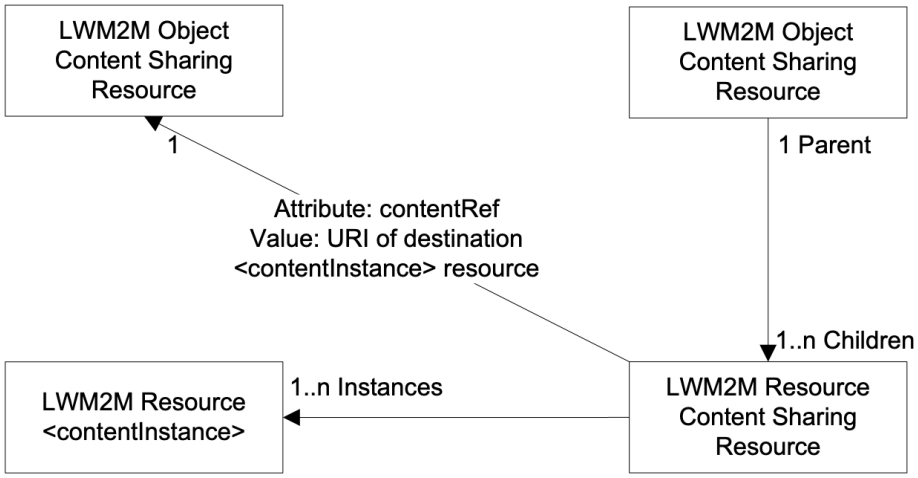

Figure 11: Figure 8.2.1-1: Relationships of LWM2M Semantically Enabled Content Sharing Resources

or deletion of the parent LWM2M Object or Object Instance Content Sharing Resource maps to the following operations on the LWM2M Client.

Table 22: Table 8.2.2-1: LWM2M Resource Content Sharing Resource Lifecycle Translation

LWM2M Operation Device Management & Service Enablement Interface	oneM2M Resource and Operation LWM2M Resource or Resource Instance Child Content Sharing Resource
Write	create child Content Sharing Resource for new Resource or Resource Instance. The name of the Content Sharing Resource shall be the Resource Id for a singleton LWM2M Resource. The name of the Content Sharing Resource shall be the LWM2M Resource ID and LWM2M Resource Instance Id. When the LWM2M Resource is of type LWM2M Object Link, the <contentInstance> resource that represents the LWM2M Resource is used to represent the LWM2M Object Link by assigning the destination of of the LWM2M Object Link reference to another LWM2M Object's Content Sharing Resource. The reference is assigned using the <contentInstance> resource's contentRef attribute where the name of the attribute is "ObjectLink" and the value of the attribute is the URI of the destination <contentInstance> resource.
Not applicable	update Content Sharing Resource
Write	delete Content Sharing Resource for Resource or Resource Instance
Not Applicable	read Content Sharing Resource

Table 23: Table 8.2.2-2: LWM2M Resource <contentInstance> Lifecycle Translation

LWM2M Operation Device Management & Service Enablement Interface	oneM2M Resource and Operation LWM2M Resource or Resource Instance <contentInstance> resource
Write	create <contentInstance>

LWM2M Operation Device Management & Service Enablement Interface	oneM2M Resource and Operation LWM2M Resource or Resource Instance <contentInstance> resource
Write - Sets the Resource to default value	delete <contentInstance>
Read	read <contentInstance>

8.2.3 Mapping for the Encoding of the <contentInstance> Resource

When an AE accesses a <contentInstance> resource, the AE needs to know how the value of the Resource or Resource Instance is encoded.

Table 24: Table 8.2.3-1: Mapping of Resource or Resource Instance Encoding

Interworking Function Mapping	oneM2M Resource Attribute
The encoding of the LWM2M Resource or Resource Instance based on the Content-Type option	<contentInstance>: contentInfo. Possible contentInfo values are translated from the LWM2M Content-Type option. Note: The LWM2M Technical Specification [3] defines the value to be used for the [encoding] if the Content-Type option is not present.

8.3 Guidelines for Mapping to the Base Ontology

8.3.1 Introduction

Clause 8.2 describes the structure and relationships of the LWM2M Objects and Object Instances along with their associated Resources and Resource Instances. Using that structure this clause provides guidance on mapping the Base Ontology described by [i.5] onto that resource structure. As ontologies are created for specific applications of LWM2M Objects (e.g. Device Management, Home Automation), this clause can only be used for a basis of creating the application specific ontology because certain elements of base ontology (e.g. Aspects, Functionality, Services) cannot be inferred by the LWM2M definitions of LWM2M Objects, Object Instances, Resources or Resource Instances.

8.3.2 Mapping of the LWM2M Client

LWM2M Clients are represented as <AE> resources and are mapped to an InterworkedDevice. <AE> resources exposed by the LWM2M Server associated with the IPE are mapped to the same Area Network.

8.3.4 Mapping of the LWM2M Object, Object Instance. Resource and Resource Instance

8.3.4.1 Introduction Editor's note: Is a single sub-clause allowed?

Mapping the LWM2M Object, Object Instance. Resource and Resource Instance to the Base Ontology is based on the following guidelines:

- LWM2M Clients are mapped to InterworkedDevices
- LWM2M Objects are mapped to Services and Functionality
- LWM2M Resources that represent static (configured) properties of LWM2M Objects are mapped to ThingProperties.
- LWM2M operation of Execute map to Operation and Command
- LWM2M Create, Update, Retrieve and Delete operations permitted for the LWM2M Object or Object Instances map to Operation and Command
- LWM2M Resources (including those of type Object Link) are mapped to Input- / OutputDataPoints.
  - Sub-structures of LWM2M Resources are mapped to Variables that are sub-structures of Input- / OutputDataPoints (via the hasSubStructure relation).
    - Read-only LWM2M Resources map to Output DataPoint.
    - Write-only LWM2M Resources map to Input DataPoint.
    - Read-write LWM2M Resources map to Input Datapoint and Output Datapoint with the same instance of a Variable
  - If the LWM2M Object doesn't have a command state, the IPE will instantiate and maintain a <container> resource for command's state. In both instances the LWM2M Resource that represents the command state maps to Output DataPoint.

9 oneM2M Management Object-based Interworking Function

9.1 Introduction

Clause 5.3 introduced the Management Object-based Interworking function as depicted in Figure 5.3-3. This clause specifies the mappings of the attributes of oneM2M resources to LWM2M objects in order to allow an AE to use the oneM2M resource without needing to understand the underlying LWM2M Object, Object Instance or Resource syntax or semantics.

9.2 Translation of oneM2M Management Resource Types

9.2.1 Introduction

These mappings and procedures are used to translate between LWM2M Objects, Object Instances and Resources and the applicable management-related oneM2M resource types.

9.2.2 Translation to <mgmtObj> Resource Types

9.2.2.1 Mapping to <mgmtObj> Resource Types TS-0005 [4] provides the procedures and mappings needed to translate LWM2M Objects, Object Instances and Resources into <mgmtObj> resource types. Relevant clauses include:

- Clause 6.1 and 6.3 of TS-0005 [4] provides mapping LWM2M Objects, Object Instances and Resources into <mgmtObj> resource types.
- Clause 6.2.1 of TS-0005 [4] provides a mapping of the LWM2M Endpoint Client Name to the M2M-Node-ID.
- Clauses 6.2.2 and 6.2.3 of TS-0005 [4] provides mapping of the LWM2M Object and Instance Identifiers to the <mgmtObj> resource's objectId and objectPath attributes.
- Clause 6.7 of TS-0005 [4] provides a set of guidelines that shall be followed for one-to-one mapping of a a LWM2M Object and its resources to a corresponding oneM2M <mgmtObj> and its [objectAttribute]s.

LWM2M IPEs shall implement clause 6.1 and 6.3 or clause 6.7 of TS-0005 [4] when translating between LWM2M Objects, Object Instances and Resources and <mgmtObj> resource types.

Clause 6.2.2.2 of this present document provides the normative language that addresses the mapping of the LWM2M Endpoint Client Name used for naming the <node> resource type associated with the LWM2M Client. This is because Clause 6.2.1 of TS-0005[4] does not allow for multiple LWM2M Clients to be hosted on the same device.

Clause 6.3 of this present document provides normative language that addresses how LWM2M Object and and Instance identifiers are mapped for oneM2M resource naming and discovery. For <mgmtObj> resources, LWM2M IPEs shall implement clause 6.2.2 and 6.2.3 of TS-0005 [4] along with clause 6.3 of this present document.

9.2.2.2 Interworking of <mgmtObj> Resources

9.2.2.2.1 Introduction Clause 10.2.8 of TS-0001 [2] discusses the procedures for the *<mgmtObj>* resources as well as where the resources are hosted and the *<mgmtObj>* resource's relationship with its parent *<node>* resource. As the *<node>* resource is the parent resource of the *<mgmtObj>* resources for the node, clause 6.4.2, 6.5.2 and 6.5.3 of this present document is unable to address the relevant interworked resource settings and procedures that allows the LWM2M IPE to be notified of changes to *<mgmt O bj>* resources or to secure the *<mgmtObj>* resource. This clause provides the mapping to the settings and procedures necessary to accomplish this interworking.

9.2.2.2.2 Interworking of <mgmtObj> Resource Settings For supporting the LWM2M interworking process, a few attributes for the <node> resource,

<*mgmtObj*> resource and the <*subscription*> resource shall have a specified set of parameters.

a) Attributes of <*node*> resource

Table 25: Table 9.2.2.1.2-1: <*node*> resource - Relevant Interworked Attributes

Attributes of <*mgmtObj*> resource	Value
accessControlPolicyIDs	ACP set (see Clause 6.6)
nodeID	The M2M-Node-ID of the node which is represented by this <*node*> resource.
hostedCSELink	The resource ID of a resource where all of the following applies: - The resource is a <*CSEBase*> resource or a <*remoteCSE*> resource. - The resource is hosted on the same CSE as the present <*node*> resource. The resource represents the CSE which resides on the specific node that is represented by the current <*node*> resource.
mgmtClientAddress	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.

b) Attributes of <*mgmtObj*> resource

Table 26: Table 9.2.2.1.2-2: <*mgmtObj*> resource - Relevant Interworked Attributes

Attributes of <*mgmtObj*> resource	Value
accessControlPolicyIDs	ACP set (see Clause 6.6)
mgmtDefinition	Examples are software, firmware, memory. The list of the value of the attribute can be seen in annex D of TS-0001.

Attributes of *<mgmtObj>* resource	Value
mgmtSchema	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.
objectIDs	Contains the list of 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.
objectPaths	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. An example is: /5/0 The combination of the *objectPaths* and the *objectIDs* attribute, allows to address the technology specific data model.
mgmtLink	This attribute contains reference to a list of other *<mgmtObj>* resources in case a hierarchy of *<mgmtObj>* is needed.
[objectAttribute]	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.
description	Text format description of *<mgmtObj>*.

9.2.2.2.3 Synchronization *<mgmtObj>* resources *<mgmtObj>* resources can be maintained by AEs and/or the LWM2M IPE on behalf of the LWM2M client. Since AEs can maintain *<mgmtObj>* resources, the request originator

(i.e., AE, LWM2M IPE AE) shall create a subscription to notify the LWM2M IPE when the *<mgmtObj>* resource is created, deleted or updated using the setting as described in Table 9.2.2.1.3-1.

Table 27: Table 9.2.2.1.3-1: Subscription Procedure - <subscription> resource

Attributes of *<subscription>*	Description
accessControlPolicyIDs	Link a <accessControlPolicy> resource with the privileges: accessControlOriginator originatorID set to the LWM2M IPE AE's AE-ID accessControlOperations: Set to RETRIEVE, CREATE, UPDATE, DELETE, DISCOVER, NOTIFY
pendingNotification	Set to "sendLatest"
latestNotify	Set to "latest".
notificationContentType	Set to "resource"
<schedule>	Set to immediate notification

9.2.2.3 Example of creating new specialized <mgmtObj> resources Using the generic guidelines outlined in Clause 6.7 of TS-0005 [4], new <mgmtObj> specialization resources may be created on the CSE. Figure 9.2.2.3-1 shows the procedure a Hosting CSE executes to create a new <mgmtObj> specialization resource using the *mgmtSchema* attribute. The URI of the schema file for the <mgmtObj> specialization resource is provided in the *mgmtSchema* attribute of the request to create the <mgmtObj> resource. The Hosting CSE then retrieves the schema file using the URI and process the request as outlined below.

Step 001: The Originator shall send mandatory parameters and may send optional paramters in the Request message for a CREATE operation of a <mgmtObj> specialization resource. The specialized <mgmtObj> resource contains a full mapping of the underlying LWM2M Object and includes the URI of the XSD for the new specialized resource in the *mgmtSchema* attribute.

Step 002: The Receiver shall:

1. Check if the Originator has the appropriate privileges for performing the request.
2. Verify that the name for the created resource as suggested by the *resourceName* parameter does not already exist among child resources of the targeted resource.

Step 003: The Receiver shall check if the type <mgmtObj> specialization is present in the supportedResourceType attribute of the CSEBase. If found in the *supportedResourceType* attribute, go to Step 8; otherwise, go to Step 4.

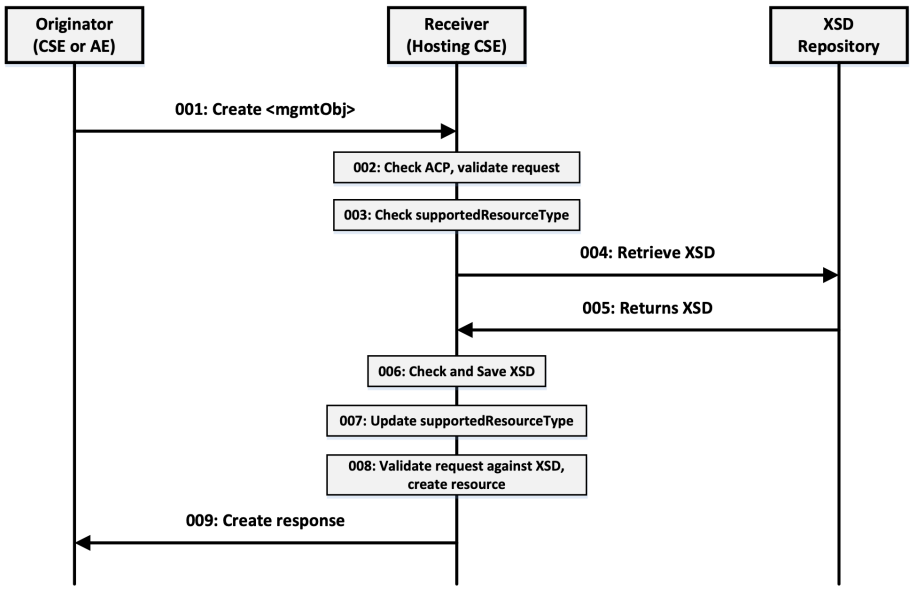

Figure 12: Figure 9.2.2.3-1: Procedure for CSE to Support New Specialized <mgmtObj> Resources

Step 004: The Receiver extracts the contents of the *mgmtSchema* attribute and retrieves an XSD from the XSD Repository.

Step 005: The XSD Repository returns the XSD for the specialized <mgmtObj> resource.

Step 006: The Receiver checks the received XSD is well formed and if it is, saves the XSD to the Receiver's local XSD repository.

Step 007: The Receiver updates the *supportedResourceType* attribute with the type of the specialized <mgmtObj>.

Step 008: The Receiver completes processing the CREATE request.

1. Assign a Resource-ID to the resource to be created.
2. Assign values for mandatory RO mode attributes of the resource and override values provided for other mandatory attributes and where allowed by the resource type definition and if not provided by the Originator itself.
3. The Receiver shall assign a value to the following common attributes:
   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.

4. 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. 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.
5. On successful validation of the Create Request, the Receiver shall create the requested resource.
6. 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.

Step 009: The Receiver shall respond with mandatory parameters and may send optional parameters in Response message for CREATE operation.

General Exceptions:

Editor's note: Is the following numbering correct? Does it refer to the previous steps?

4. The Originator does not have the privileges to create a resource on the Receiver. The Receiver responds with an error.
5. The resource with the specified name (if provided) already exists at the Receiver. The Receiver responds with an error.

- The provided information in *Content* is not accepted by the Receiver (e.g. missing mandatory parameter). The Receiver responds with an error.

9.2.2.4 LWM2M Interworking Procedure Figure 9.2.2.4-1 shows an example end-to-end procedure demonstrating how the procedures shown in Figure 9.2.2.3-1 can be utilized as part of LWM2M Interworking. The LWM2M Server and the IPE are co-located, and together they perform oneM2M procedures on behalf of the LWM2M Device. A LWM2M Device will initially register to the LWM2M Server and provides a list of all the LWM2M objects it supports. The LWM2M Server/IPE will then perform a oneM2M registration on behalf of the device and requests to create an <AE> resource. Once the <AE> resource is

created, the IPE will then create a <node> resource to host all the <mgmtObj> resources for the device. As part of this step, the nodeLink attribute of the <AE> resource will be updated to point to the newly created <node> resource. The IPE then proceeds to create a specialized <mgmtObj> resource for each LWM2M objects the device supports. The specialized <mgmtObj> resource will directly map to the corresponding LWM2M Object using the *objectAttribute* attribute of the <mgmtObj>. This will allow for one-to-one mapping of LWM2M resources to oneM2M attributes. Once all the <mgmtObj> specializations are created, the IPE/LWM2M Server returns an appropriate response to the LWM2M Device.

NOTE: The following procedure shows only a high level call flow of the interactions among the LWM2M Device, the LWM2M Server/IPE, and the Hosting CSE. It does not detail all the steps required to perform the indicated operations.

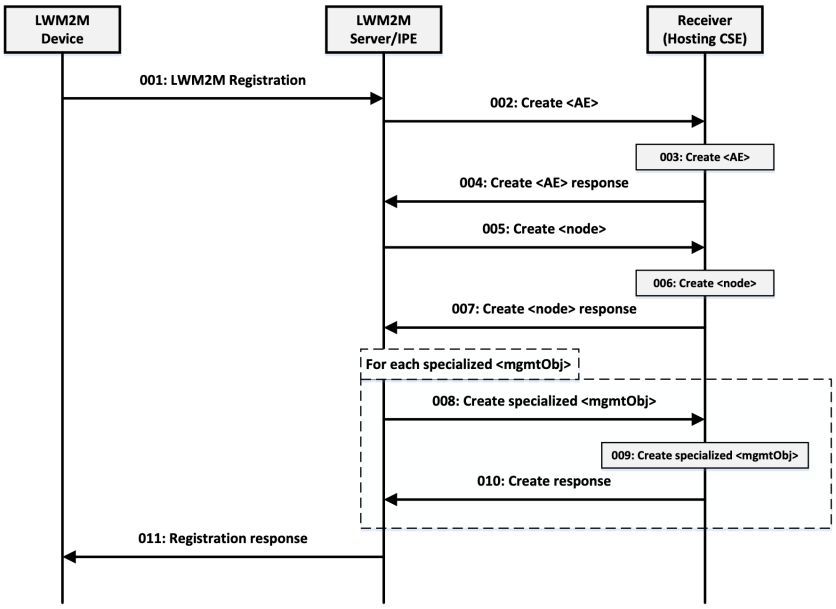

Figure 13: Figure 9.2.2.4-1: End-to-end LWM2M Interworking Procedure

Step 001: A LWM2M device registers to the LWM2M Server and provides a list of supported LWM2M Objects. Co-located with the LWM2M Server is the IPE.

Step 002: In response to the LWM2M registration, the IPE requests to create an <AE> resource on the Hosting CSE on behalf of the LWM2M Device.

Step 003: The Hosting CSE evaluates the request, performs the appropriate checks, and creates the <AE> resource.

Step 004: A response is sent to the IPE indicating the <AE> resource was

created.

Step 005: The IPE proceeds to create a <node> resource for the LWM2M Device so <mgmtObj> specialization resources can be created for AE's to manage the device. As part of this multi-step procedure, the *nodeLink* attribute of the <AE> resource created in Step 003 is updated to point to the newly created <node> resource.

Step 006: The Hosting CSE creates the <node> resource and updates the <AE>'s *nodeLink* attribute as well.

Step 007: The Hosting CSE returns an appropriate response for creating the <node> resource.

Step 008: For each of the LWM2M Objects supported by the LWM2M Device, the IPE creates an appropriate specialized <mgmtObj> resource as a child of the <node> resource. This <mgmtObj> specialization maps one-for-one with the corresponding LWM2M Object.

Step 009 : .The Hosting CSE creates the <mgmtObj> specialization resource.

Step 010: An appropriate response is returned to the IPE for the creation of the <mgmtObj> specialization resource.

Step 011: The IPE/LWM2M Server completes the LWM2M registration procedure by sending the LWM2M Device an appropriate response.

9.2.2.5 Use of oneM2M attribute level subscription in LWM2M Interworking Once the <mgmtObj> resources are created, attribute level subscriptions can be created to get notifications for certain desired operation such as a firmware update process. With the one-to-one mapping relationship between LWM2M resources and oneM2M [objectAttribute], attribute level subscriptions can be made. Table 9.2.2.5-1 shows a mapping of the LWM2M Firmware Update object to a new oneM2M Firmware <mgmtObj> with all the LWM2M resources mapped one for one to oneM2M attributes.

Table 28: Table 9.2.2.5-1: LWM2M Firmware Update Object 1:1 Mapping to oneM2M <mgmtObj> Resource

LWM2M Resource Name	LWM2M Resource #	[objectAttribute]
Package	0	package
Package URI	1	pkgURI
Update	2	update
State	3	state
Update Result	5	updateResult
Pkg Name	6	pkgName
Pkg Version	7	pkgVersion
Firmware Update Protocol Support	8	firmwareUpdateProtocolSupport

LWM2M Resource Name	LWM2M Resource #	[objectAttribute]
Firmware Update Delivery Method	9	firmwareUpdateDeliveryMethod

The call flow below uses the updated Firmware Update object mapping in Table 9.2.2.5-1 to allow an AE to monitor the state of a firmware update on a LWM2M device. Prior to initiating a firmware update, the AE can subscribe to the ***state*** attribute of the 1:1 mapped ***firmware*** <mgmtObj> to get notifications of the firmware update process. Figure 9.2.2.5-1 shows the end-to-end use case of using oneM2M's attribute level subscription to get notifications for the state of the firmware update.

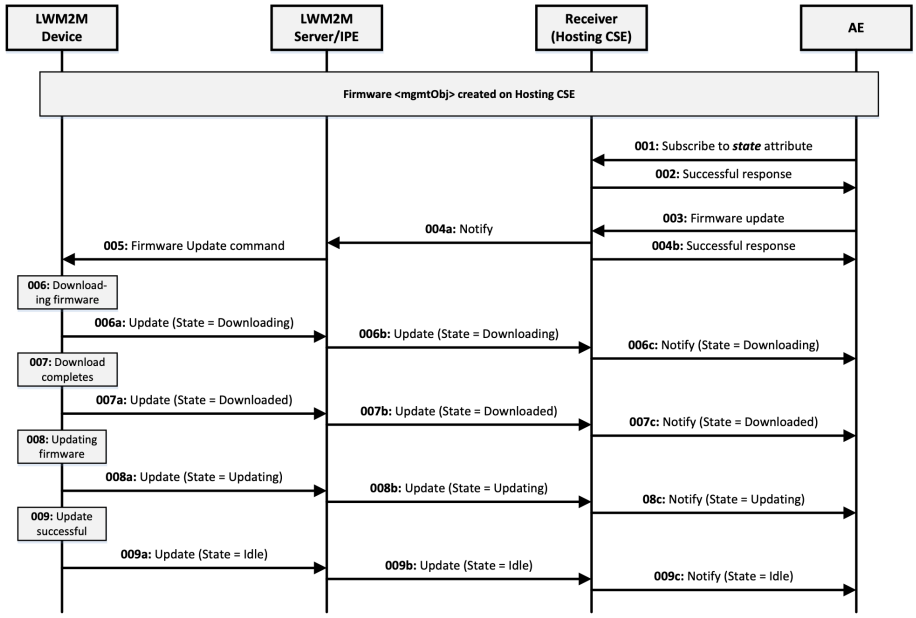

Figure 14: Figure 9.2.2.5-1: oneM2M Attribute Level Subscription Use Case

Step 001: AE subscribes to the ***state*** attribute of the ***firmware*** <mgmtObj> associated with LWM2M Device.

Step 002: Hosting CSE grants the subscription and sends a successful response.

Step 003: AE sends a firmware update request to update the firmware on the LWM2M Device.

Step 004: Hosting CSE processes the request and performs the following:

a) Hosting CSE sends a Notify message to the IPE of the firmware update

request.

b) Hosting CSE sends a successful response to the AE.

Step 005: The LWM2M Server/IPE sends a firmware update command to the LWM2M Device.

Step 006: The LWM2M Device receives the command and performs the download of the firmware image.

a) The LWM2M Device sends an update to the LWM2M Server with status that ***state = Downloading***

b) The LWM2M Server/IPE sends an update to the Hosting CSE with status that ***state = Downloading***.

c) The Hosting CSE sends a notify to the AE with status that ***state = Downloading***.

Step 007: The LWM2M Device completes the download and updates its state resource.

a) The LWM2M Device sends an update to the LWM2M Server with status that ***state = Downloaded***.

b) The LWM2M Server/IPE sends an update to the Hosting CSE with status that ***state = Downloaded***.

c) The Hosting CSE sends a notify to the AE with status that ***state = Downloaded***.

Step 008: The LWM2M Device begins updating the firmware and updates its state resource.

a) The LWM2M Device sends an update to the LWM2M Server with status that ***state = Updating***.

b) The LWM2M Server/IPE sends an update to the Hosting CSE with status that ***state = Updating***.

c) The Hosting CSE sends a notify to the AE with status that ***state = Updating***.

Step 009: The LWM2M Device completes updating the firmware successfully and updates its state resource.

a) The LWM2M Device sends an update to the LWM2M Server with status that ***state = Idle***.

b) The LWM2M Server/IPE sends an update to the Hosting CSE with status that ***state = Idle***.

c) The Hosting CSE sends a notify to the AE with status that ***state = Idle***.

Annex A (Informative): Introduction to OMA LightweightM2M (LWM2M)

A.1 Introduction

OMA Lightweight M2M is a protocol for device and service management for M2M. The main purpose of this technology is to address service and management needs for constrained M2M devices, over UDP and SMS bearers.

> NOTE: This annex provides an overview of the LWM2M protocol. The authoritative source for the protocol is provided by the LWM2M Technical Specification [3].

The crucial aspects in this work are the:

- Target devices for this protocol are resource constraint devices (e.g. 8-16bit MCU, RAM is in tens of KB and flash is in hundreds of KB).
- Ability to perform Data collection and remote control of devices without the need for complex computing and UI operations.
- Optimization of network resources to allow a large numbers of devices may be connected to the communication network simultaneously.
- Fusion of device functionalities management and service manipulation into a single protocol.

From the implementation view LWM2M has the following features:

- Suitable for resource constraint devices.
- Usage of compact binary packets.
- Support for multiple data encoding formats that include Binary , JSON, plain text and opaque data formats.
- Support for reporting information from the Server to the Client when specified condition are met.
- Easy to be implemented though the reuse of existing implementation of IETF technologies : e.g. CoAP.
- (Constrained Application Protocol) for the Transfer Protocol, and DTLS (Datagram, Transport Layer Security) [i.3] for securing the Server/Client exchanges.

One of typical use cases of using LWM2M technology is the firmware upgrade of streetlights [i.4].

- A Streetlights supervisor is responsible for managing the streetlights system. (There are thousands of streetlights in the city and low-cost LWM2M devices embedded in the streetlights.)

- The supervisor needs to remotely upgrade of the firmware of a specific streetlight or a group of streetlights.

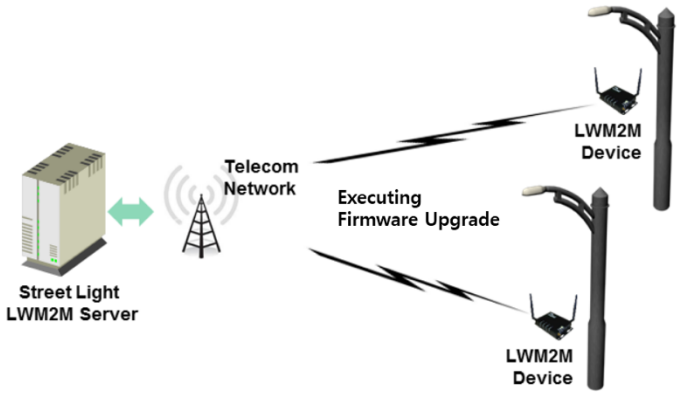

Figure 15: Figure A.1-1: Firmware Upgrade of Streetlight of Use Case using LWM2M

A.2 Architecture

As shown in the Figure A.2-1, the layout is the architecture of LWM2M [3]. The Components specified by OMA LWM2M compose the LWM2M enabler which specifies the LWM2M Server / LWM2M Client interface. The LWM2M Server and LWM2M Client are typically instantiated in a M2M Server and a M2M Device.

Based on the deployment scenario, the LWM2M Server has the bootstrapping capability itself, or the LWM2M Bootstrap Server exists separately for security reasons.

A.3 Terminology

LWM2M [3] is a RESTful protocol with concepts that are similar to oneM2M, however LWM2M uses different terms for these concepts. The following table provides a comparison of applicable LWM2M and oneM2M terminology.

Table 29: Table A.3-1: LWM2M/oneM2M Terminology Mapping

LWM2M Terminology	oneM2M Terminology
Client Endpoint	<AE> resources that reside on devices and oneM2M nodes.

LWM2M Terminology	oneM2M Terminology
Object, Object Instance	Resource in general; <contentInstance> resource when used for interworking.
Resource	Attribute for a Resource.

A.4 Reference Points

A.4.1 Introduction

This clause introduces the interfaces carried over the reference point consisting of two main components LWM2M Server and the LWM2M Client.

A.4.2 Functional Components

A.4.2.1 LWM2M Server The LWM2M Server is a logical component which serves as an endpoint of the LWM2M protocol.

A.4.2.2 LWM2M Client The LWM2M Client is a logical component. This LWM2M Client serves as an endpoint of the LWM2M protocol and communicates with the LWM2M Server to execute the device management and service enablement operations from the LWM2M Server and reporting results of the operations.

A.4.3 Interfaces

There are four interfaces supported by the reference point between LWM2M server and LWM2M Client. The logical operation of each interface is defined as follows:

- Bootstrap:
  - This interface is used to provision essential information into the LWM2M Client so that the LWM2M Client can register to the LWM2M Server(s) after bootstrap procedure has completed.
- Client Registration:
  - This interface allows the LWM2M Client register to the LWM2M Server. This procedure lets the Server know the existence and information (e.g. address, capabilities) of the LWM2M Client so that LWM2M Server can perform M2M services and device management on the LWM2M Client.
- Device Management and Service Enablement:
  - This interface allows the LWM2M Server to perform the device management and M2M service enablement operations. Over this interface, the LWM2M Server can send operations to the LWM2M Client and gets response of the operations from the LWM2M Client.
- Information Reporting:

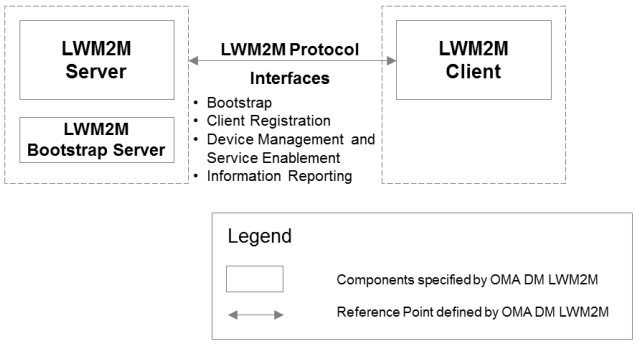

Figure 16: Figure A.2-1: LWM2M Architecture

– This interface allows the LWM2M Client to report resource information to the LWM2M Server. This Information Reporting can be triggered periodically or by events (e.g. resource information is changed and configured conditions are met).

A.5 Protocols

A.5.1 Protocol Stack

The LWM2M has the protocol stack defined as below.

- LWM2M Objects: LWM2M Objects are designed for the functionality provided by the LWM2M enabler. The LWM2M specification [i.4] defines a set of Standard Objects. Other Objects may also be added by OMA, external SDOs (e.g. the IPSO alliance) or vendors to enable certain M2M Services.
- LWM2M Protocol: LWM2M protocol defines the logical operations and mechanisms per each interface.
- CoAP: The LWM2M utilizes the IETF Constrained Application Protocol [i.2] as an underlying transfer protocol across UDP and SMS bearers. This protocol defines the message header, request/response codes, message options and retransmission mechanisms. The LWM2M only uses the subset of features defined in CoAP.
- DTLS: DTLS [i.3] is used to provide secure UDP/SMS on-device channels between the LWM2M Server and the LWM2M Client for all the messages interchanged.
- UDP Binding with CoAP (Mandatory): Reliability over the UDP transport

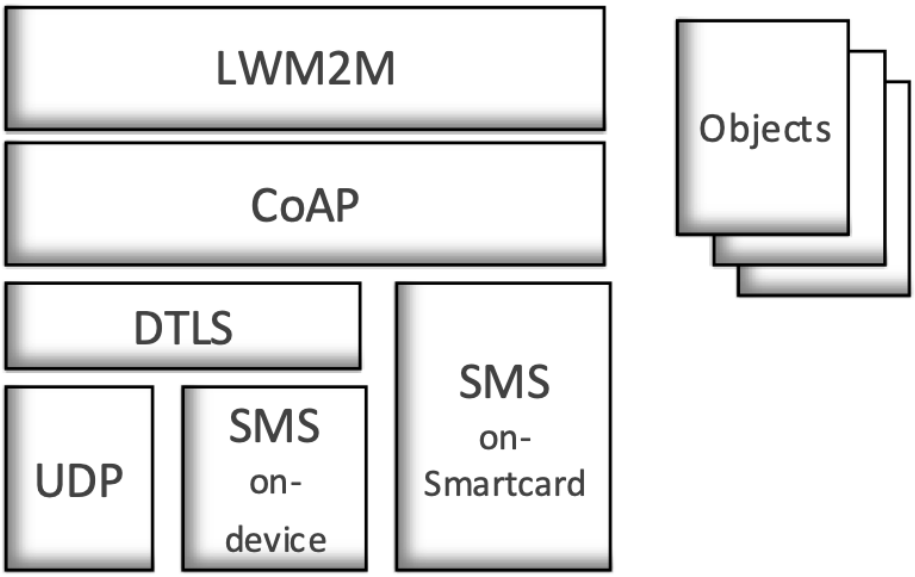

Figure 17: Figure A.5.1-1: LWM2M Protocol Stack

is provided by the built-in retransmission mechanisms of CoAP.

- SMS Binding with CoAP (Optional): CoAP is used over SMS by placing a CoAP message in the SMS payload using 8-bit encoding.

A.5.2 Data Model

In the LWM2M Enabler technical specification [i.4], a simple data model is described. Basically, a resource made available by data model of the LWM2M Client is a Resource, and Resources are logically organized into Objects. Figure A.5.2-1 illustrates this structure, and the relationship between Resources, Objects, and the LWM2M Client. The LWM2M Client may have any number of Resources, each of which belongs to an Object.

Resources are defined per Object, and each resource is given a unique identifier within that Object. Each Resource is designed to have one or more Operations that it supports. A Resource may be a single or multiple (possibility of several instantiations) one, dependent on the Resource definition in Object specification. An Object defines a grouping of Resources, for example the Firmware Object contains all the Resources used for firmware update purposes. The LWM2M enabler defines standard Objects and Resources and other Objects may be added to enable a certain M2M Services.

Object needs to be instantiated either by the LWM2M Server or the LWM2M Client, which is called Object Instance, before using the functionality of an

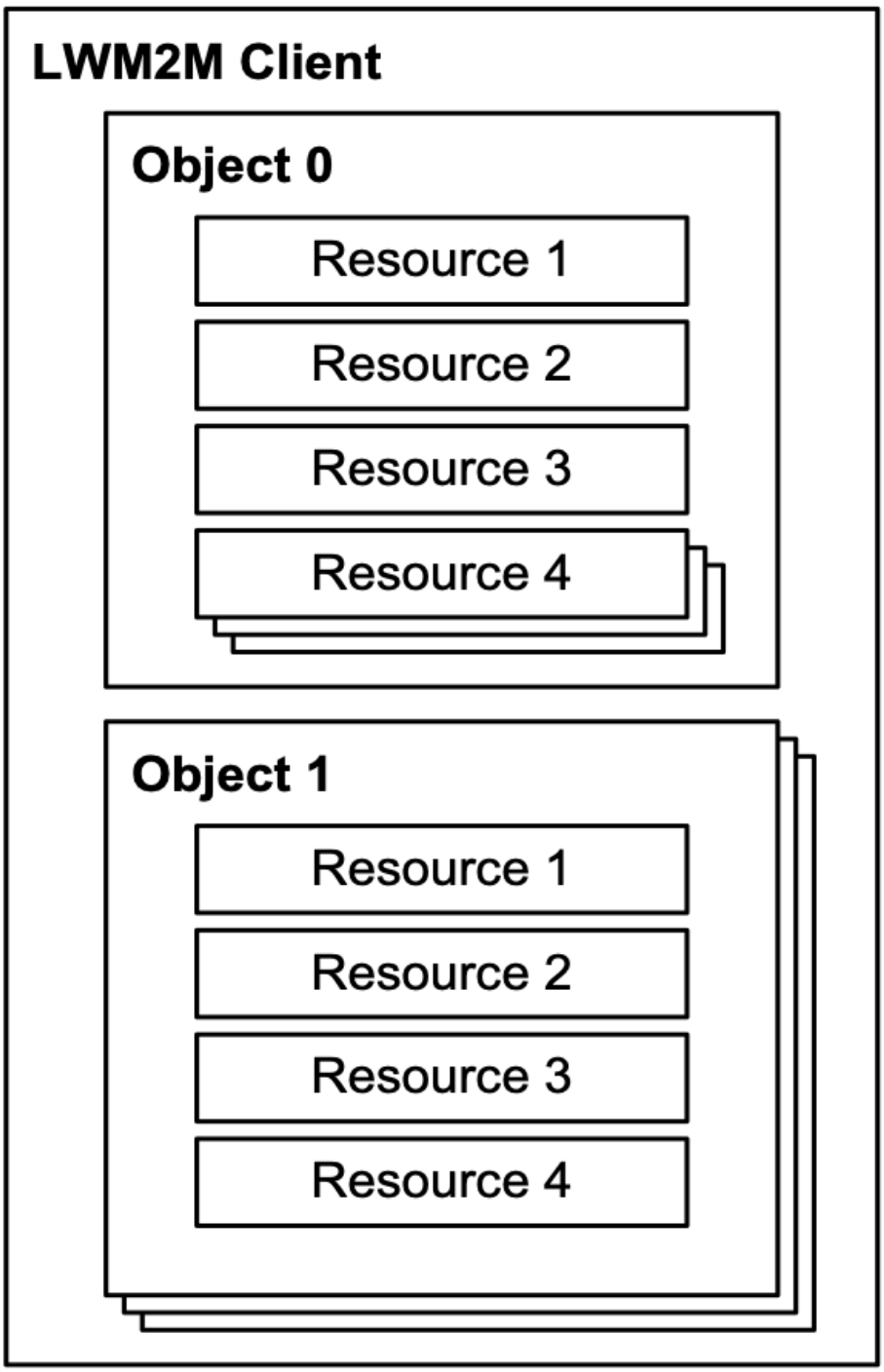

Figure 18: Figure A.5.2-1: LWM2M Data Model [3]

Object. After Object Instance is created, the LWM2M Server can access that Object Instance and Resources in the Object Instance. Furthermore a Resource can contain a simple value (e.g. sensor measure), or a reference to an Object Instance.

A.5.3 Interface Descriptions

A.5.3.1 Bootstrap The Bootstrap interface is used to provision essential information into the LWM2M Client in order to allow the LWM2M Client to be able to register to a certain LWM2M Server. There are four modes for bootstrapping:

- Factory Bootstrap: the LWM2M Client is already provisioned at the time of the device manufacture. The pre-configured data is stored in the LWM2M Client.
- Bootstrap from Smartcard: When the Device supports a Smartcard and retrieval of bootstrap message from Smartcard is successful, the LWM2M Client processes the bootstrap message from the Smartcard and applies it to the LWM2M Client.
- Client initiated Bootstrap: the LWM2M Client requests and retrieves the bootstrap message from a LWM2M Bootstrap Server. In this case the LWM2M Client needs to be pre-provisioned with the LWM2M Bootstrap Server Bootstrap Information.
- Server initiated Bootstrap: the LWM2M Bootstrap Server provisions the bootstrap message into the LWM2M Client after recognizing the existence of the LWM2M Device. In this case the LWM2M Client needs to be pre-provisioned with the LWM2M Bootstrap Server Bootstrap Information.

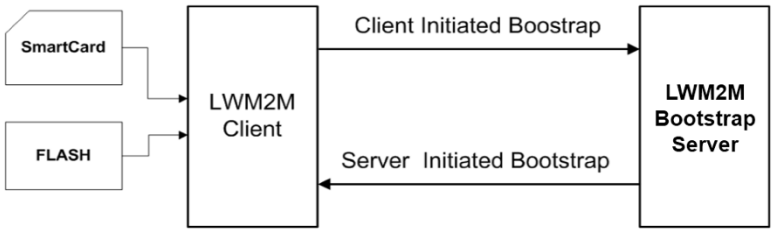

Figure 19: Figure A.5.3.1-1: Bootstrap Modes

A.5.3.2 Client Registration The Client Registration interface is used by the LWM2M Client to register with one or more LWM2M Servers, maintain each registration, and de-register from the LWM2M Server(s). When registering, the LWM2M Client indicates its Endpoint Name, MSISDN, supporting binding modes, lifetime of registration, the list of Objects the Client supports and available Object Instances. The registration is a soft state, with a lifetime

indicated by the registration lifetime. The LWM2M Client periodically performs an update of its registration information to the registered Server(s). If the lifetime of a registration expires without receiving an update from the Client, the Server removes the registration information. Finally, when shutting down or discontinuing use of a Server, the Client performs de-registration.

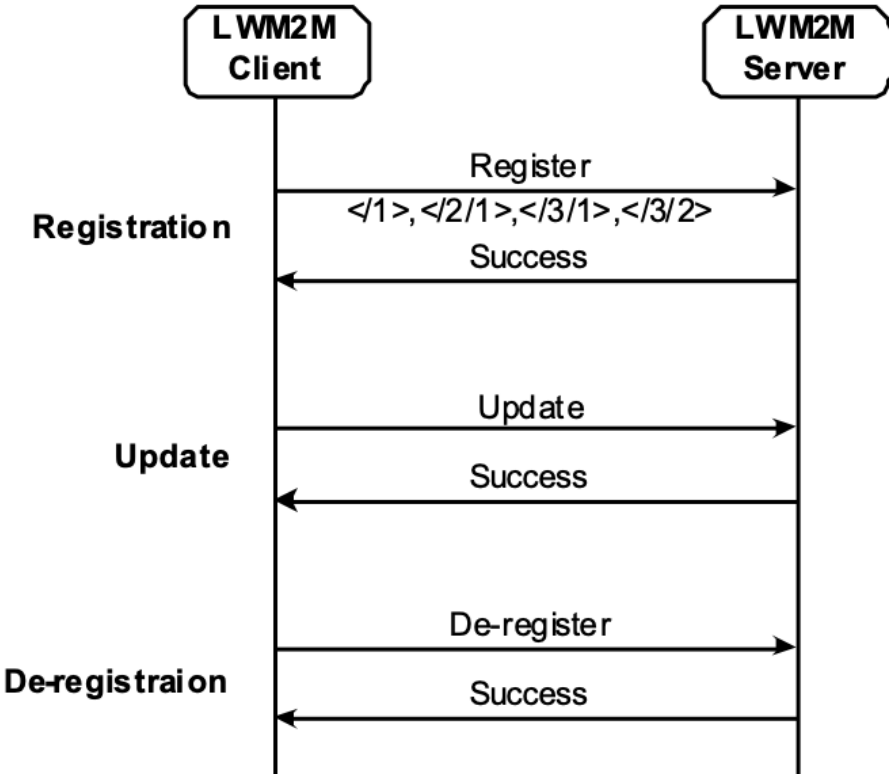

Figure 20: Figure A.5.3.2-1: Example of Registration Procedure

A.5.3.3 Device Management and Service Enablement This interface is used by the LWM2M Server to access Resources available from a LWM2M Client using Create, Read, Write, Delete, or Execute operations. The operations that a Resource supports are defined in the definition of its Object.

A.5.3.4 Information Reporting This interface is used by the LWM2M Server to observe any changes in a Resource on the LWM2M Client, receiving notifications when new values are available. The LWM2M Server needs to configure observation related parameters by sending "Write Attribute" operation before observing Resources in the LWM2M Client. This observation relationship is initiated by sending an "Observe" operation to the L2M2M Client for an

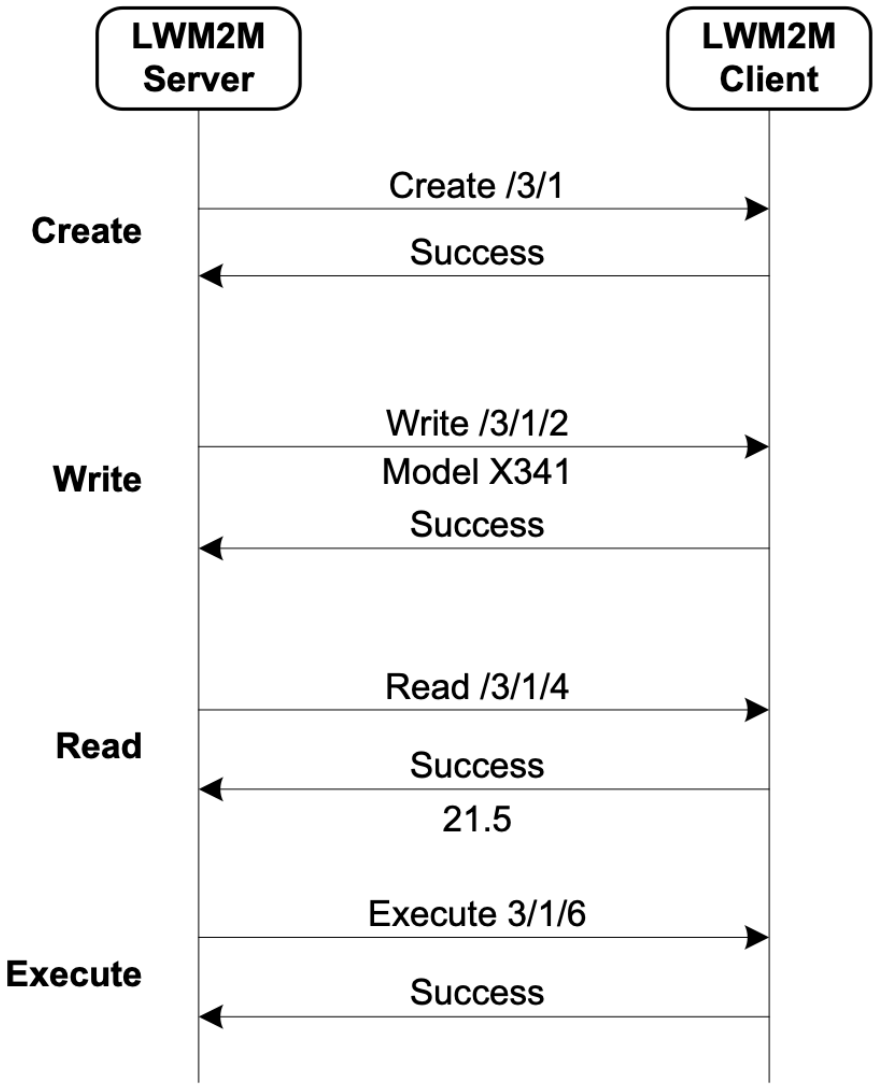

Figure 21: Figure A.5.3.3-1: Example of Device Management and Service Enablement Interface

Object Instance or Resource. An observation ends when a “Cancel Observation” operation is performed

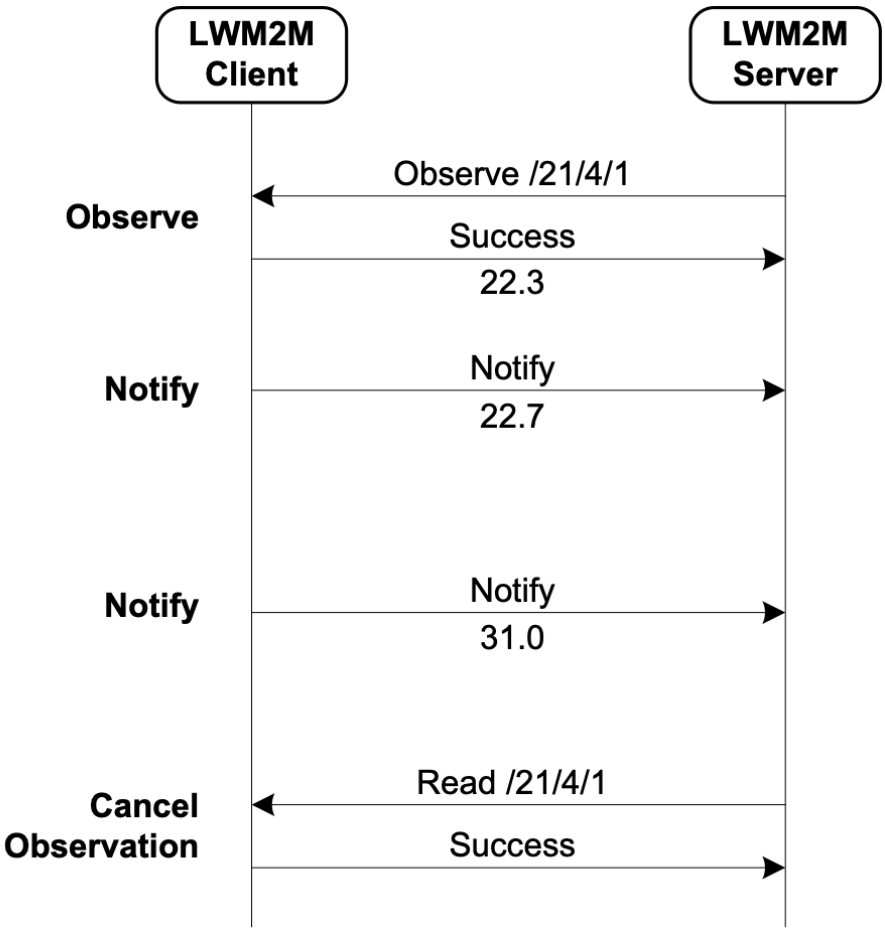

Figure 22: Figure A.5.3.4-1: Example of Information Reporting Interface

A.6 Functions

A first set of standard Objects for the LWM2M 1.0 enabler have been developed:

- Server Security: security data related to the LWM2M server(s) and/or the LWM2M Bootstrap Server.
- Server: data, configuration, functions related to the LWM2M Server.
- Access Control: to check whether the LWM2M server has access right for performing an operation on Resources in the LWM2M Client.
- Device: provision of a range of device related information, device reboot and factory reset function.

- Connectivity Monitoring: to monitor parameters related to underlying network connectivity.
- Firmware: provision of firmware management, installing and updating new firmware.
- Location: provides location information of the LWM2M Devices.
- Connectivity Statistics: to statistical information of network connection (e.g. SMS counter, UDP data size).

These Standard Objects are intended to support a variety of functionalities to manage LWM2M Devices. OMA has already specified others LWM2M objects (e.g. as Software Management, Device Capability Management,) and may create further objects in future. Furthermore, other organizations and companies may define additional LWM2M Objects for their own M2M services using according to LWM2M Object Template and Guideline Annex in [3]: e.g. oneM2M has specified the set of LWM2M Objects around CMDH Policy functionality; IPSO Alliance has developed LWM2M Objects for various sensors.

History

Publication history

Version	Date	Description
V3.1.1	February 2019	Release 3 - Publication
V4.0.0	February 2023	Release 4 - Publication

Draft history (to be removed on publication)

Version	Date	Description
V4.0.1	2024-11-09	SDS-2024-0136-TS-0014_initial_conversion_to_markdown