Developer Guide for oneM2M application

Any hardcopy unless authorized or a softcopy in a directory other than the central repository is an Uncontrolled Copy.

This hardcopy of the document is a Controlled Copy only when signed by an authorised signatory.
Issued to: _____ **Signature and Date**

CENTRE FOR DEVELOPMENT OF TELEMATICS
MANDI ROAD, MEHRAULI, NEW DELHI 110030, INDIA
ELECTRONICS CITY (PHASE I), HOSUR ROAD, BANGALORE 560100, INDIA

Table of Contents

List of Tables

List of Figures

1. INTRODUCTION

The developer's guide is a quick access guideline for development of oneM2M applications, where the various resource mappings, naming conventions, messages etc have been provided to guide the application developers.

1.1 References

[1]. TS-0001 Functional Architecture
[2]. TS-0009 HTTP Protocol Binding
[3]. TS-0008 CoAP Protocol Binding
[4]. TS-0004 Service Layer Core Protocol Specification
[5]. TS-0003 Security Solutions

1.2 Definitions, acronyms and terminology

1.2.1 Definitions

Resource: uniquely addressable entity in oneM2M architecture. Resources are denoted as <>

<ACP>	: Represents Access Control Policies for authorization
<AE>	: Represents application instance
<Container>	: Represents containers for data instances created by application to maintain the hierarchy
<ContentInstance>	: Represents data instance created by application
<Subscription>	: Represents subscription information for its subscribed-to resource.
<Group>	: Represents a group of resources
<fanout>	: Represents a virtual resource which is used to fan out request to each member of <group> resource

1.2.2 Acronyms

ADN-AE	Application Dedicated Node - Application Entity
ADN	Application Dedicated Node
AE	Application Entity
CSE	Common Service Entity
CoAP	Constrained Application Protocol.
HTTP	Hyper Text Transfer Protocol
IN	Infrastructure Node
M2M	Machine to Machine
REST	REpresentational State Transfer

2. USE CASE

This developer's guide is based on a street lighting use case involving lights that can be remotely controlled by a LMC (Light monitoring and control) application leveraging the capabilities of oneM2M. An overview of the use case is shown below.

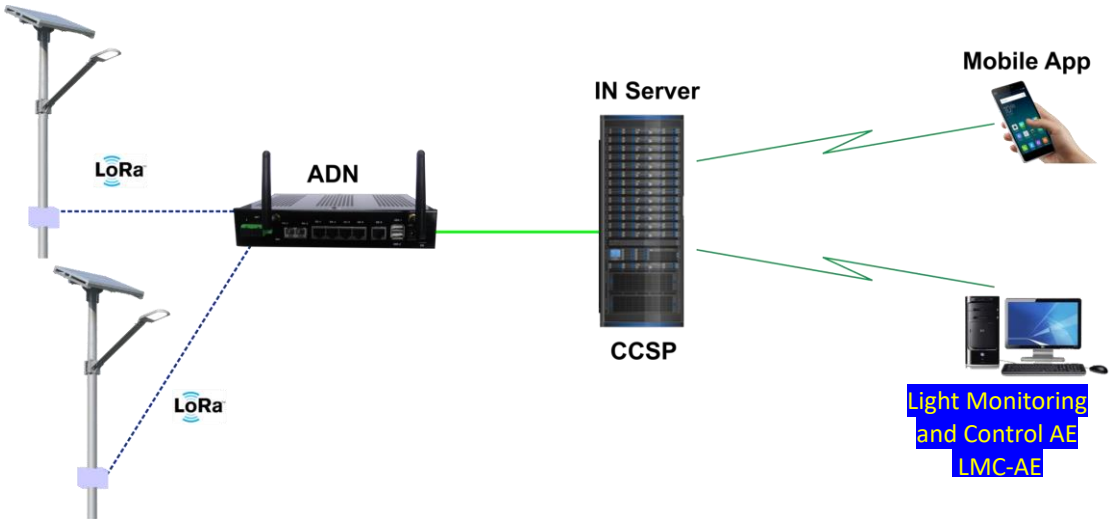


Figure 1: Use Case for Developer event

The main components are introduced as follows:

1. The lights are deployed in a Street and are attached to an ADN.
2. The ADN communicates with a platform allowing the lights to be controlled remotely by the LMC application.
3. The platform supports a set of services to enable the LMC application to more easily control the lights. Some examples of services include registration, discovery, data management, group management, subscription/notification etc.
4. The LMC hosts an application used to remotely control the lights and supports the following capabilities:
    - Discovery of lights deployed in the street.
    - Sending commands to change light state i.e. ON and OFF.
    - Retrieval of light state.

3. ARCHITECTURE

This clause describes how the different components of this use case can be represented by corresponding oneM2M architectural entities as shown below.

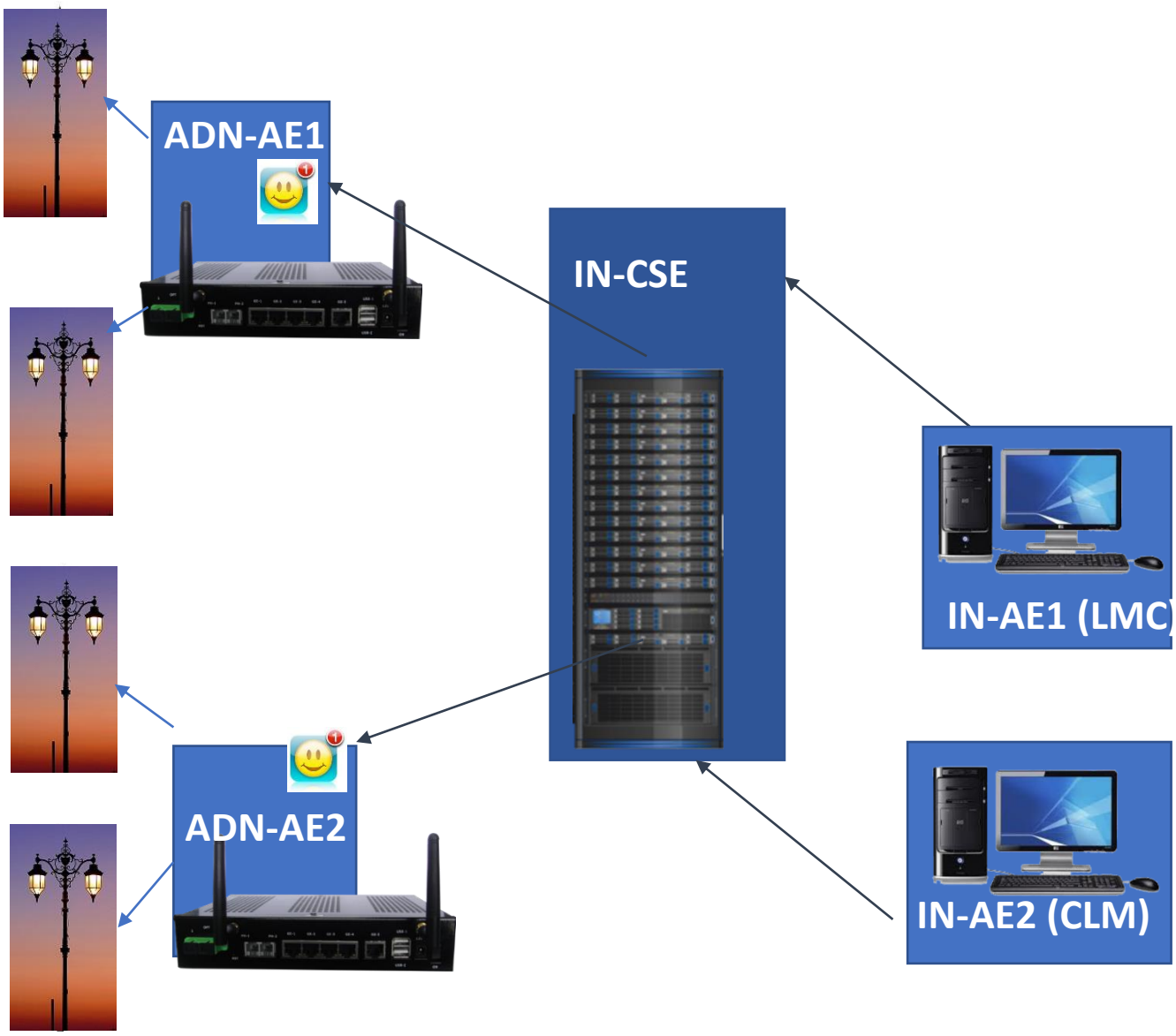


Figure 2: Architecture for Developer event

In the oneM2M functional architecture two basic types of entities are defined. One is an **AE** (Application Entity) and the other is a **CSE** (Common Services Entity). In this use case, the lights and LMC each host an AE. Also, an **IN-CSE** (Infrastructure Node CSE) is hosted by the oneM2M Service Provider.

The oneM2M defined **Mca** reference point is used to interface an AE and CSE. In this use case, the Mca reference point is used between the Light ADN-AEs and platform IN-CSE and

between the LMC AE and IN-CSE. Details of Application entities used in the current case are given below.

3.1 oneM2M Compliant Application 1 – ADN-AE

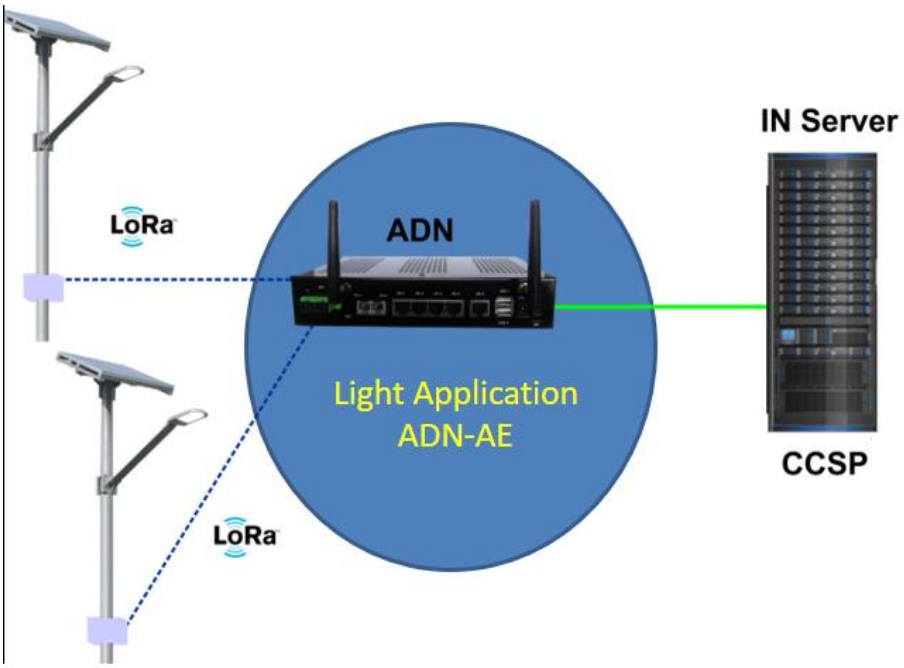


Figure 3: ADN-AE for Light

1. ADN-AE application will be collecting data from 2 lights in our development.
2. ADN-AE will send the data from each light to CCSP over HTTP/CoAP protocol in oneM2M compliant form .
3. Data will be like Voltage reading , current reading , Intensity reading.
4. ADN-AE will be sending this data periodically.
5. It will also send the initial state of each light (OFF)

3.2 oneM2M Compliant Application 2 –IN-AE 1(LMC-AE –Light Monitoring and Control AE)

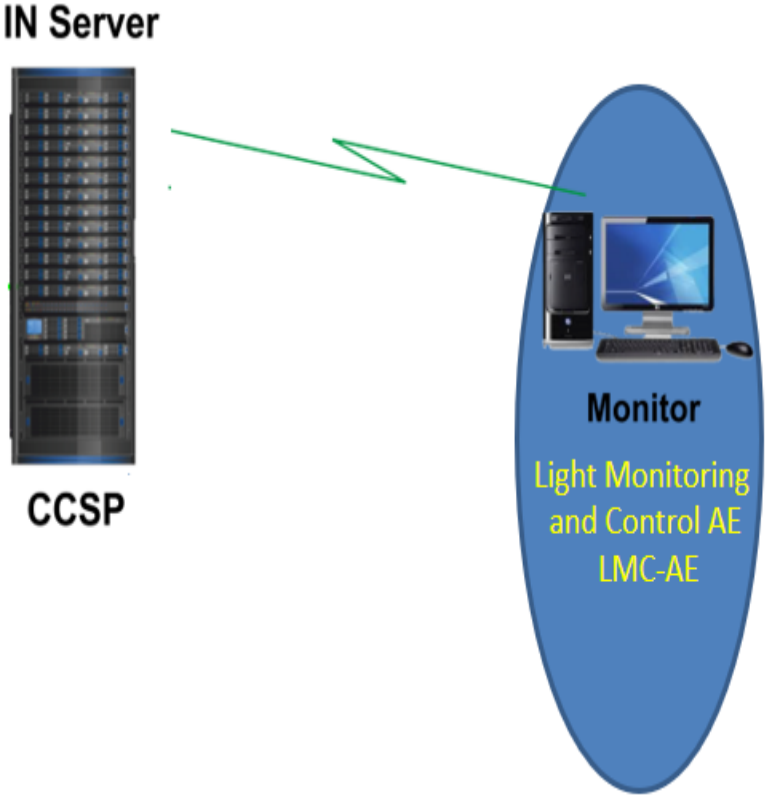


Figure 4 : IN-AE2 (LMC- AE)

1. LMC-AE will get the data created by ADN-AE, in form of notifications from IN.
2. LMC-AE will also control the state of the light. State can be ON or OFF.
3. IN-CSE on getting State change Request from LMC-AE ,shall send notification to the ADN-AE

3.3 oneM2M Compliant Application 3 -IN-AE 2 (CLM-AE -Central Light Monitoring GUI)

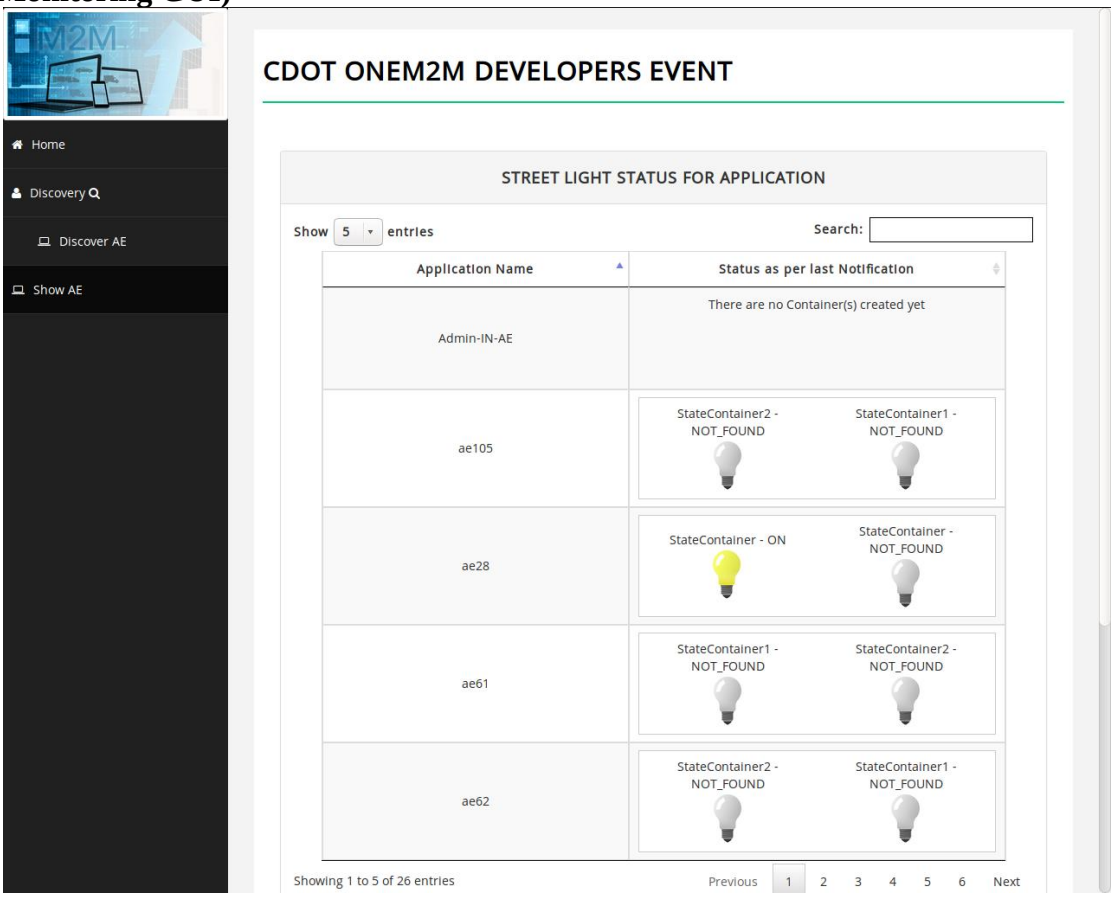


Figure 5 : IN- AE2 (CLM)

- All ADN-AE applications should set the notification URI to above portal in order to demonstrate the state of their light change from LMC-AE
- Also, one notificationURI can be set to the light hardware actuator uri.

4. GUIDELINES FOR APPLICATION DEVELOPMENT

1.Devices and application entities are independently addressable with host names resolved by DNS network services;

2. HTTP/CoAP binding of oneM2M primitives will be used;

3. JSON/XML serializations of oneM2M primitives are used in the current use case;

4. All mandatory HTTP/CoAP headers (options in case of CoAP) are presented in the HTTP/CoAP requests while optional headers are selectively used in the current use case;

5.All mandatory resource attributes for resources presented in the current use case are presented in the HTTP/CoAP requests while optional resource attributes are selectively used in the current use case;

6. All resources created in the current use case are addressable with the oneM2M Resource Identifier form of *non-hierarchical address*;

7. Short names for the representation of the resources and attributes are used in the current use case; (mapping described in section 8.1)

8. All request originators will send *Blocking Requests* for accessing resources located in CSEs.

9. TLS PSK based Security is considered in the current use case for HTTP;

10. For HTTP/CoAP oneM2M messages in JSON/XML refer to corresponding payload documents, which will be available during Developer's event.

S. No.	Protocol	JSON/XML	Document Name
1.	HTTP	JSON	DevEvent_HTTP_JSONPayLoad_Apr2019
2.	HTTP	XML	DevEvent_HTTP_XMLPayLoad_Apr2019
3.	CoAP	JSON	DevEvent_CoAP_JSONPayLoad_Apr2019
4.	CoAP	XML	DevEvent_CoAP_XMLPayLoad_Apr2019

5. ONEM2M PROCEDURES FOR APPLICATION ENTITIES

5.1 Security Association Establishment (SAE)

5.1.1 SAE Requirements

M2M services are offered by CSEs to AEs and/or other CSEs. To be able to use M2M Services offered by one CSE, the AEs and/or CSEs need to be mutually identified and authenticated by that CSE, in order to provide protection from unauthorized access and Denial of Service attacks.

This mutual authentication enables to additionally provide encryption and integrity protection for the exchange of messages across a single Mca reference point. In addition, communicating AEs that require similar protection for their own information exchanges can be provisioned to apply the same security method to their communications. This is the purpose of the Security Association Establishment (SAE) procedure.

5.1.2 PSK based SAE

In this example it is assumed that authentication between the ADN-AE and IN-CSE is performed using provisioned keys (Kpsa) and key identifiers (KpsaID).

Configuration of ADN-AE:

- The AE is assumed to be configured with the details such as registrar information, its keys etc. This information shall be shared via mail or offline). An example of credential configuration is given in the figure below.The AE is assumed to be configured with a pair of credentials (psk, psk_identity) associated with the CSE-ID. The length of the keys Kpsa is not mandated by TS-0003 and left to implementation. The key identifiers comply with the format specified in clause 10.5 of TS-0003.

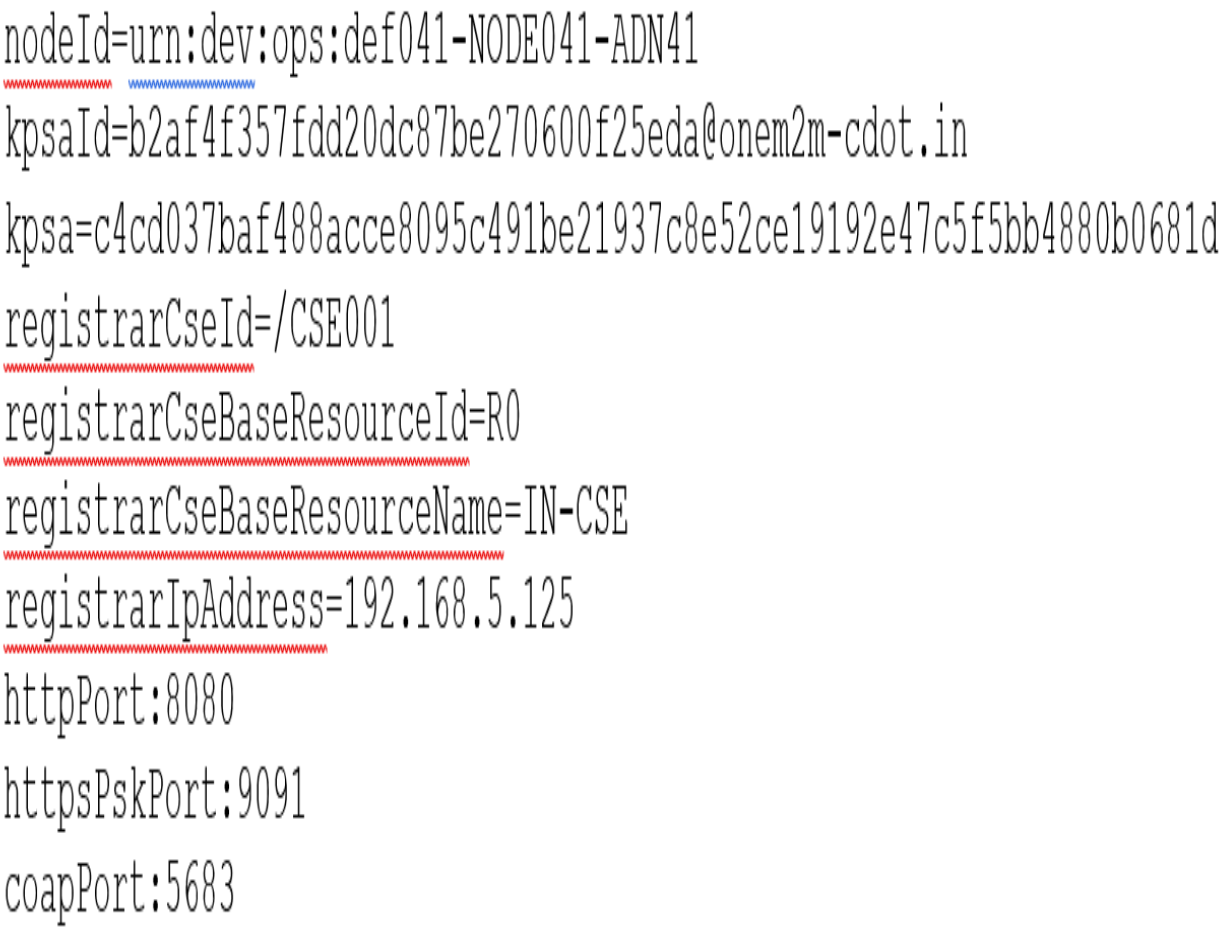


Figure 6 Example Configuration Details for ADN-AE

Operation of ADN-AE1 and ADN-AE2

When the AE is triggered to establish a TLS-PSK session with the IN-CSE using some security credentials pair (Kpsa, KpsaID), the following should occur automatically based on the AE's configuration:

- AE's TLS Client is triggered to perform a TLS-PSK handshake with the TLS values (psk, psk_identity) set to the values of (Kpsa, KpsaID), and with the configured list of TLS ciphersuites.
- On completion of the TLS handshake, the AE associates the established TLS session with the IN-CSE's CSE-ID.

5.2 Application Registration (after successful SAE)

1. An AE which want to resgister , send a "Create <*AE*>" request primitive via Mca interface to its Registrar CSE .
2. When an ADN-AE or an LMC-AE registers, the registrar CSE needs to retrieve and check the service subscription information which is defined in a <*m2mServiceSubscriptionProfile*> instance on the IN-CSE (see clause 10.2.2.2 of TS-0001)
3. The instance of a <*m2mServiceSubscriptionProfile*> has children <serviceSubscribedNode> which includes information of ADN node (for ADN-AE and LMC AE each) configured for a subscriber and another <serviceSubscriberNode> which includes information of its registrar CSE, this <*serviceSubscribedNode*> reveals *nodeID* and *CSE-ID* of the IN-CSE and it is assumed to have a *ruleLink* attribute which includes the resource identifier of a <*serviceSubscribedAppRule*> resource which includes information related to ADN-AE, LMC-AE.
4. The <*serviceSubscribedAppRule*> resource can have 3 specific attributes: *allowedCredIDs*, *allowedAppIDs* and *allowedAEs*. Each of these attributes generally can include a list of elements. If a <*serviceSubscribedAppRule*> relates to a single AE only, the *allowedAppIDs* and *allowedAEs* attributes contain a single element only.
5. At <AE> registration, information included in applicable <*serviceSubscribedAppRule*> resources are examined by the registrar CSE and compared if it matches with security credentials employed for Security Association Establishment (SAE), and App-ID and AE-ID indicated in the registration request message.
6. In case the information used by the registree does not match with the information given in applicable <*serviceSubscribedAppRule*> resources, the registration request needs to be rejected by the registrar CSE.

5.3 Resource Creation (<accessControlPolicy>)

1. The resource controls "who" is allowed to do "what" and the context in which it can be used for accessing resources. It stores a representation of privileges.
2. ADN-AE will create an <accessControlPolicy> to control the access of its resource to itself only.
3. Once LMC-AE registers, ADN-AE will discover LMC-AE and update its <accessControlPolicy> to give access to LMC-AE too.

5.4 Resource Creation (<container>, <CI>)

1. Concept of creating resources for keeping application data
2. ADN-AE will create <container> for each light's data and will create corresponding <contentInstance> to store the light's data
3. ADN-AE will create <container> for each light's state
4. LMC-AE will create <contentInstance> for light's state in ADN light's state <container>.

5.5 Resource Creation (<subscription>)

1. ADN-AE will create <subscription> resource on light's state container to get the notifications for light's state change.
2. LMC-AE will create <subscription> resource on light's data to get the notifications for light's data change.

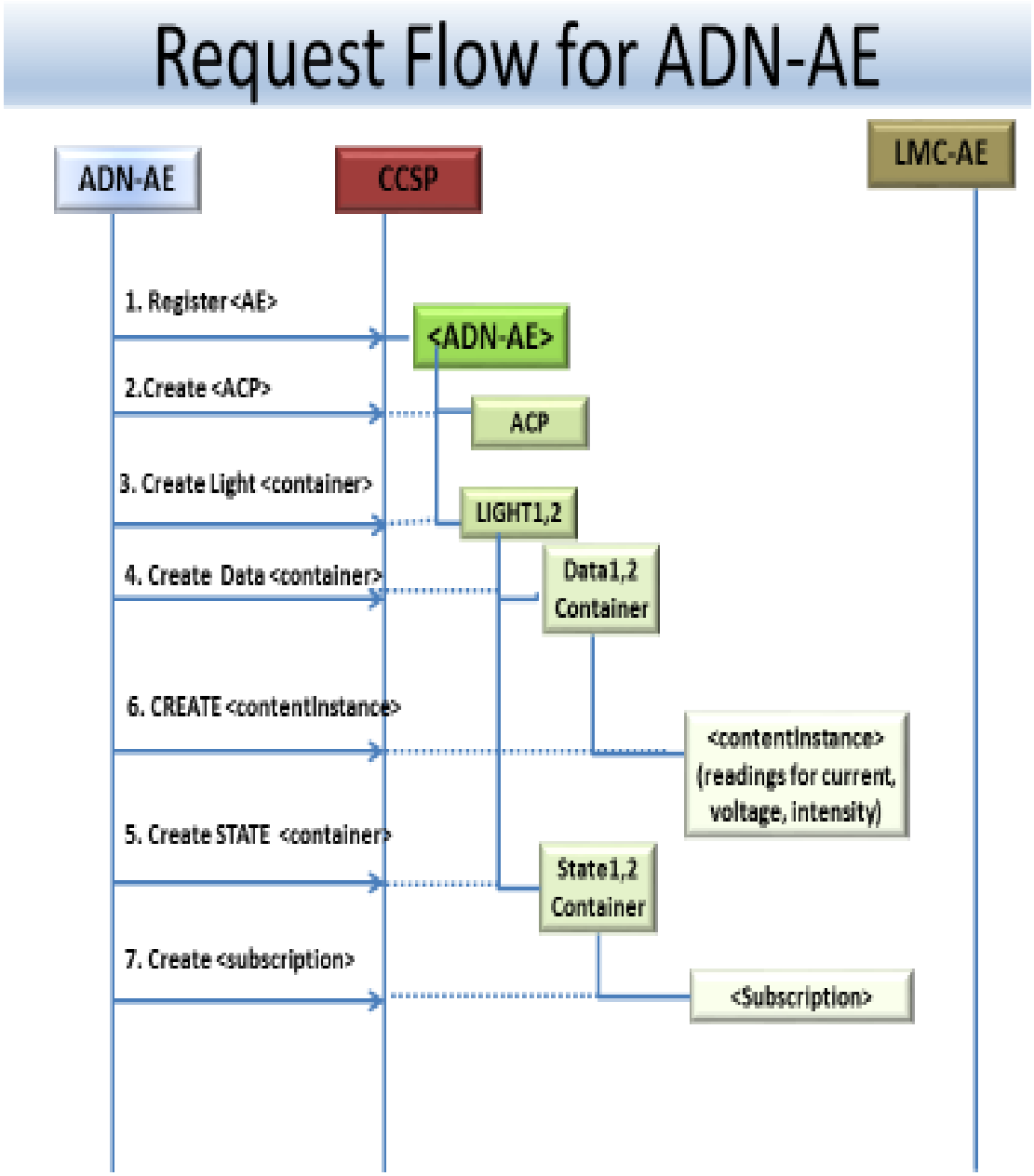
5.6 Resource Creation (<group>)

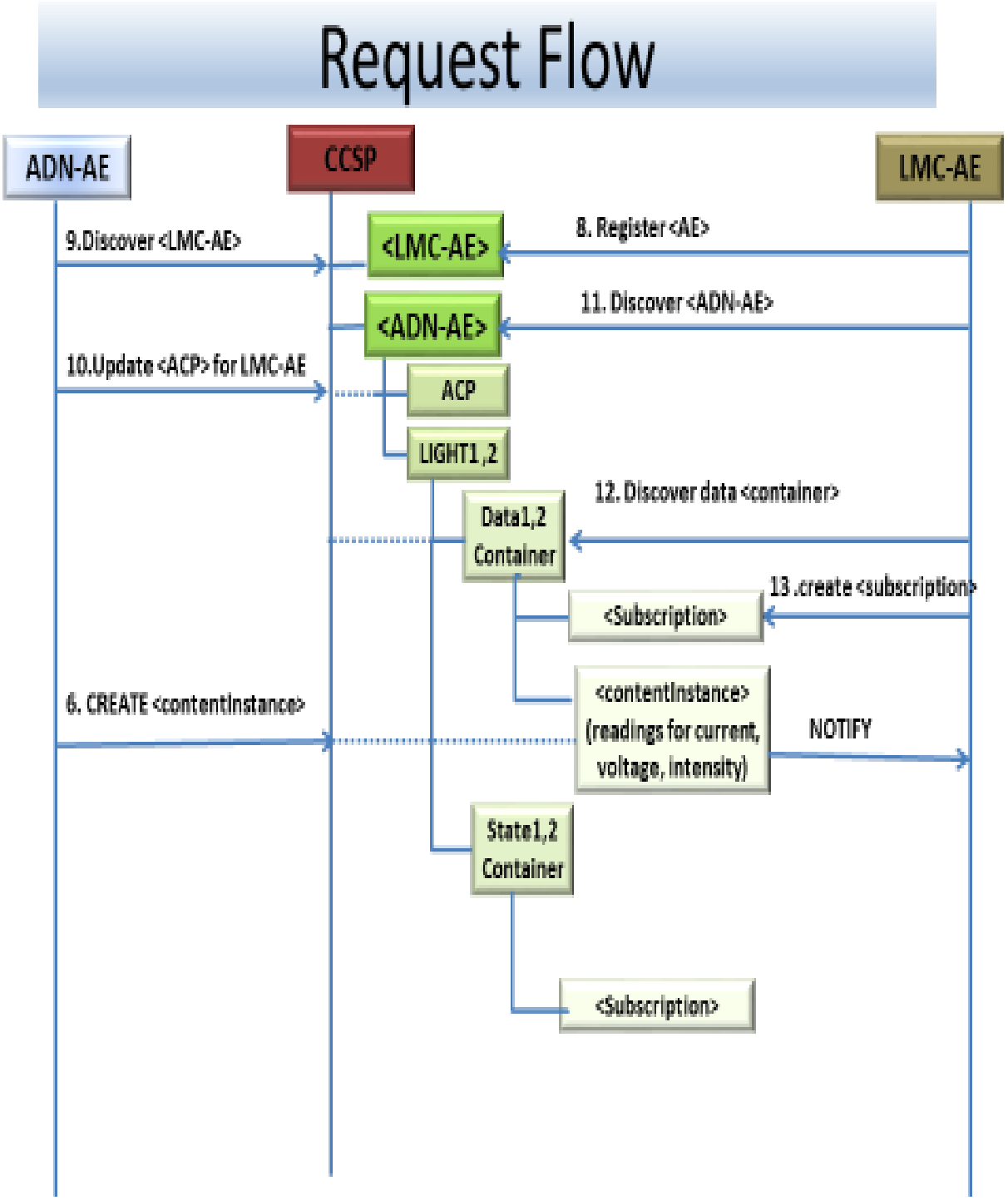
1. LMC-AE will create <group> resource for light's state containers to control them in group.

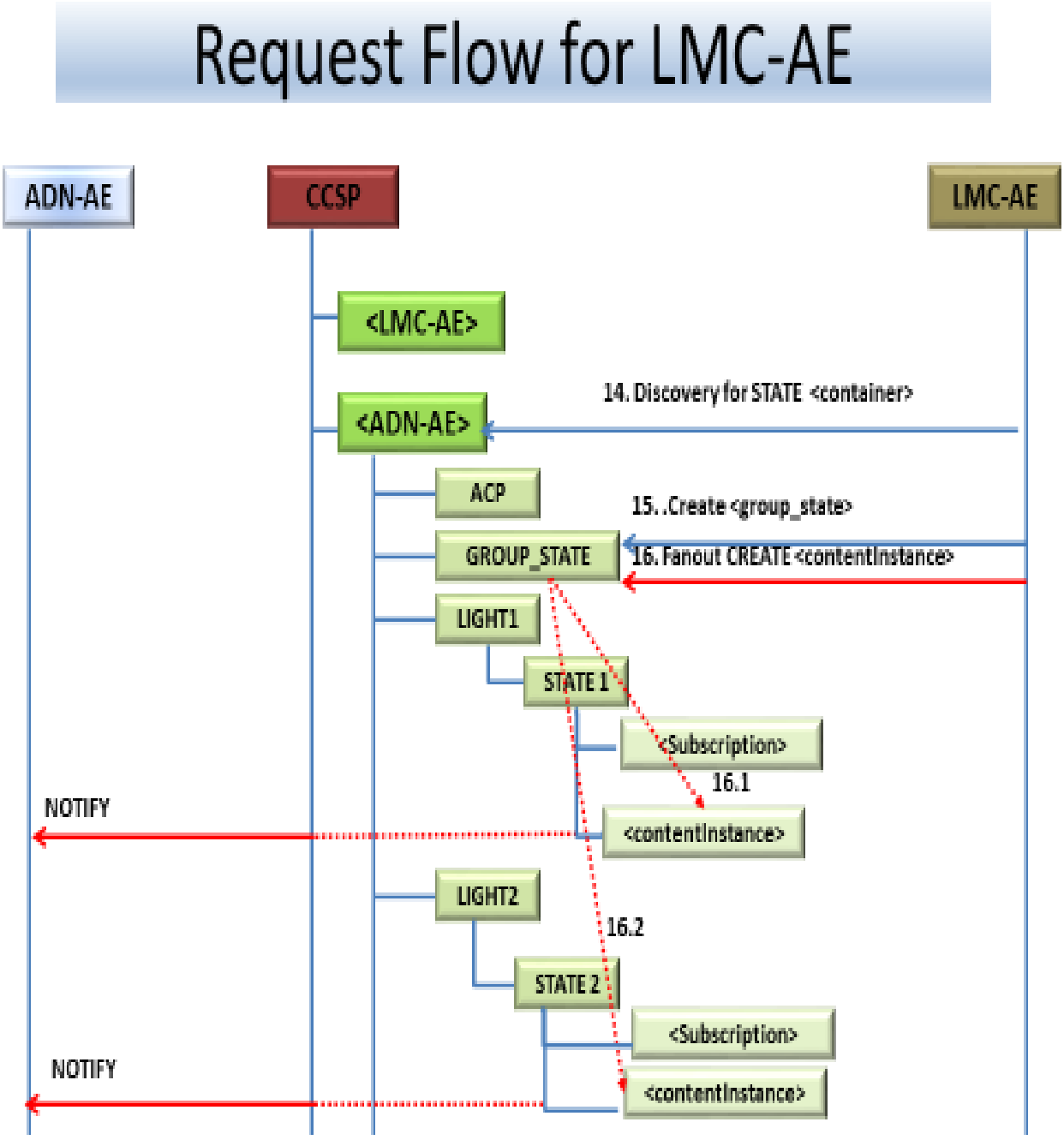
5.7 Application and container Discovery

1. Concept of searching other application and its resources on the basis of some pre-defined information e.g. labels
2. ADN-AE and LMC-AE: These 2 applications have to know each other , know their resources . For this we shall use the concept of "labels" in Discovery .
3. Light Application (ADN-AE) will associate label with each resource created , while LMC-AE shall be using these labels to discover the resources.

6. APPLICATION MESSAGE SEQUENCE







ADN-AE

1. Create AE
2. Create ACP (Define AccessControlPolicy to access the resource<AE>)
3. Create Light Containers (Step a: LIGHT1, Step b: LIGHT2) with label as "company_light" under ADN-AE
4. Create Data Containers (Step a: DATA1, Step b: DATA2) with label as "company_data" under LIGHT1 and LIGHT2 respectively
5. Create State Containers (Step a: STATE1, Step b: STATE2) with label as "company_state" under LIGHT1 and LIGHT2 respectively

6. Create CI in DATA1, DATA2 container (data corresponding to the voltage, current is stored) and CI in STATE1 and STATE2 containers for initial state for respective lights.
7. Create Subscription on STATE1, STATE2 (with notification URI set as ADN-AE and CDOT HES-AE) in order to get the notification for any change in current state of lights.

Light Monitoring and Control LMC-AE

8. Create AE

ADN-AE to set access permissions for LMC-AE

9. Discover the LMC-AE with label field set in filter criteria
10. Now UPDATE ACP (created in step2) in ADN-AE to give privileges to LMC AE

Light Monitoring and Control LMC-AE

11. Discover the ADN-AE label field set in filter criteria
12. Discover the DATA containers with label = company_data set in filter criteria
13. Create Subscription on DATA Containers (Step a: DATA1, Step b: DATA2), discovered in step 12, to get notification on CI creation in DATA containers
14. Discover the STATE containers with label = company_state set in filter criteria
15. Create GROUP1 for all STATE containers for creation of CI in STATE containers
16. Fanout contentInstance CREATE request on GROUP1

7. MAPPING AND NAMING CONVENTION

7.1 Short Name and Long Name Mapping

7.1.1 Resource Mapping

Short Name	Long Name	Resource Type (ty)
ae	applicationEntity	2
acp	accessControlPolicy	1
cnt	container	3
cin	contentInstance	4
sub	subscription	23
grp	group	9
fopt	fanoutpoint	Not Applicable
ggn	Single Notification	Not Applicable

Table 1: Resource Mapping (Short Name – Long Name)

7.1.2 Resource Attributes Mapping (used in Request)

Short Name	Long Name
ri	*resourceID*
pi	*parentID*
rn	*resourceName*
lbl	*labels*
acpi	*accessControlPolicyIDs*
api	*appID*
rr	*requestReachability*
pv	*privileges*

pvs	*selfPrivileges*
acr	*accessControlRule*
acor	*accessControlOriginators*
acop	*accessControlOperations*
nu	*notificationURI*
enc	*eventNotificationCriteria*
ent	*eventNotificationType*
nct	*notificationContentType*
con	*contentInfo*
mnm	*maxNrOfMembers*
mid	*memberIds*
poa	*pointOfAccess*
srv	*supportedReleaseVersion*

Table 2 : Resource Attributes Mapping (used in Request)

7.2 Resource Naming Convention

Resource Name to be set in request (for the readability purpose)

- <> denotes variable and [] denotes optional, x is used for sequence of many individuals from one company.
- For naming of any resource, resource short name and company name are to be used (used for the viewing purposes in the monitoring GUI) along with other details if required e.g.

For AE resource:	ae_<companyName><x>
For Container resource:	cnt_<companyName><x>
For Contentinstance resource:	cin_<companyName><x>
For AccessControlPolicy resource:	acp_<companyName><x>
For subscription resource:	sub_<companyName><x>
For group resource:	grp_<companyName><x>

Table 3 : Resource Name Convention

7.3 Example naming for <AE> resource: (detailed in payloads)

For Participant 1: **adn-ae_cdot1, lmc-ae_cdot1 etc.**

For Participant 2: **adn-ae_cdot2, lmc-ae_cdot2 etc.**
(From same company)

7.4 Status Code Mapping

oneM2M Response Status Codes	HTTP Status Codes [10]	CoAP Status Codes
2000 (OK)	200 (OK)	2.05 (OK)
2002 (DELETED)		2.02 (DELETED)
2004 (UPDATED)		2.04 (CHANGED)
2001 (CREATED)	201 (Created)	2.01 (CREATED)
1000 (ACCEPTED)	202 (Accepted)	2.03 (ACK)
4000 (BAD_REQUEST)	400 (Bad Request)	4.00(Bad Request)
4001(RELEASE_VERSION_NOT_SUPPORTED)		
4102 (CONTENTS_UNACCEPTABLE)		
4110 (GROUP_MEMBER_TYPE_INCONSISTENT)		
6010 (MAX_NUMBER_OF_MEMBER_EXCEEDED)		
4101 (SUBSCRIPTION_CREATOR_HAS_NO_PRIVILEGE)	403 (Forbidden)	4.03 (Forbidden)
4103 (ORIGINATOR_HAS_NO_PRIVILEGE)		
5105 (RECEIVER_HAS_NO_PRIVILEGE)		
5106 (ALREADY_EXISTS)		4.00(Bad Request)
5203 (TARGET_NOT_SUBSCRIBABLE)		4.03 (Forbidden)
5205 (SUBSCRIPTION_HOST_HAS_NO_PRIVILEGE)		
4106 (ORIGINATOR_HAS_NOT_REGISTERED)		
4107 (SECURITY_ASSOCIATION_REQUIRED)		
4108 (INVALID_CHILD_RESOURCE_TYPE)		
4109 (NO_MEMBERS)		
4117 (ORIGINATOR_HAS_ALREADY_REGISTERED)		
5214 (TARGET_HAS_NO_SESSION_CAPABILITY)		
5215 (SESSION_IS_ONLINE)		
4004 (NOT_FOUND)	404 (Not Found)	4.04 (Not Found)
5103 (TARGET_NOT_REACHABLE)		
4005 (OPERATION_NOT_ALLOWED)	405 (Method Not Allowed)	4.05 (Method Not Allowed)
5207 (NOT_ACCEPTABLE)	406 (Not Acceptable)	4.06 (Not Acceptable)
4008 (REQUEST_TIMEOUT)	408 (Request Timeout)	4.04 (Not Found)
4104 (GROUP_REQUEST_IDENTIFIER_EXISTS)	409 (Conflict)	4.00(Bad Request)
4105 (CONFLICT)		4.03 (Forbidden)
4015 (UNSUPPORTED_MEDIA_TYPE)	415 (Unsupported Media Type)	4.15 (Unsupported Media Type)
5000 (INTERNAL_SERVER_ERROR)	500 (Internal Server Error)	5.00 (Internal Server Error)
5204 (SUBSCRIPTION_VERIFICATION_INITIATION_FAILED)		

oneM2M Response Status Codes	HTTP Status Codes [10]	CoAP Status Codes
5209 (GROUP_MEMBERS_NOT_RESPONDED)		
5001 (NOT_IMPLEMENTED)	501 (Not Implemented)	5.01 (Not Implemented)

Table 4 : Status Code Mapping

7.5 HTTP protocol specific Mapping

7.5.1 Mandatory oneM2M Request and Response headers

Mandatory HTTP request headers (as per oneM2M):

- X-M2M-RI (Request Identifier)
- X-M2M-ORIGIN (except AE registration) (Request Originator)

Mandatory HTTP Response headers (as per oneM2M):

- X-M2M-RSC (Response Status Code)
- X-M2M-RI (Request Identifier corresponding to which this response is sent)

7.5.2 oneM2M operation to HTTP Method Mapping

oneM2M Operation	HTTP Method
Create	POST
Retrieve	GET
Update	PUT
Delete	DELETE
Notify	POST

Table 5 : OneM2M operation to HTTP method Mapping

7.6 CoAP protocol specific Mapping

7.6.1 Mandatory oneM2M Request and Response headers

Mandatory oneM2M request options (as per oneM2M):
oneM2M-FR (Request Originator–256, except AE Registration)
oneM2M-RQI (Request Identifier-257)
oneM2M-TY (Resource Type – 267 only for Create resource requests)

Mandatory CoAP Response headers (as per oneM2M):
oneM2M-RSC (Response Status Code-265)
oneM2M-RQI (Request Identifier-257, corresponding to which this response is sent)

7.6.2 oneM2M operation to CoAP Method Mapping

oneM2M Operation Parameter	CoAP Method (Corresponding Integer Value)
CREATE	POST(2)
RETRIEVE	GET(1)
UPDATE	PUT(3)
DELETE	DELETE(4)
NOTIFY	POST(2)

Table 6 : OneM2M operation to CoAP method Mapping

7.6.3 oneM2M Options for CoAP protocol

No	Name	Format
256	oneM2M-FR	string
257	oneM2M-RQI	string
264	oneM2M-EC	uint
265	oneM2M-RSC	uint
267	oneM2M-TY	uint
271	oneM2M-RVI	string

7.6.4 Media Types for CoAP Protocol

Please use following values for Content-Format and Accept

oneM2M Operation Parameter	CoAP Method (Corresponding Integer Value)
application/xml	41
application/json	50